Towards Efficient Black-Box Robustness Evaluation and Certifying Robustness against Adversarial Patch Attacks

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät
der Eberhard Karls Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von
Dott. Mag. Maksym Yatsura
aus Dnipro/Ukraine

Tübingen
2024

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation:	23.01.2025
Dekan:	Prof. Dr. Thilo Stehle
1. Berichterstatter/-in:	Prof. Dr. Matthias Hein
2. Berichterstatter/-in:	Prof. Dr. Seong Joon Oh

Abstract

Deep Learning is an emerging field of Artificial Intelligence that has already revolutionized countless industrial and societal aspects of the modern world. Despite its impressive results, Deep Learning field still has a lot of unsolved problems and the most worrisome ones are related to its robustness, trustworthiness and safety. Contemporary Deep Learning models are consistently demonstrated to be vulnerable to adversarially crafted perturbations of the input such as imperceptible noise or small patches. In this thesis, we address the problems of evaluating and improving Deep Learning robustness. We propose an approach that we call *Meta Square Attack* to meta-learn the search distribution of black-box random search based adversarial attacks on Deep Learning models to improve the evaluation of black-box robustness. We study certified defences against adversarial patch attacks that provide a guaranteed lower bound on the model robustness in this threat model. We propose *BagCert* that allows end-to-end training and efficient certification of the classification models. We also propose an inpainting-based approach called *Demasked Smoothing* which is the first method to certify the robustness of semantic segmentation models against adversarial patches. Demasked Smoothing can work with arbitrary segmentation models and requires no additional training.

Zusammenfassung

Deep Learning ist ein entstehender Bereich der künstlichen Intelligenz, der bereits unzählige industrielle und gesellschaftliche Aspekte der modernen Welt revolutioniert hat. Trotz seiner beeindruckenden Ergebnisse hat der Bereich des Deep Learnings noch viele ungelöste Probleme, und die besorgniserregendsten beziehen sich auf seine Robustheit, Vertrauenswürdigkeit und Sicherheit. Zeitgenössische Deep Learning Modelle haben sich durchweg als anfällig für gegnerisch gestaltete Störungen der Eingabe wie unmerkliches Rauschen oder ein kleiner Patch erwiesen. In dieser Dissertation befassen wir uns mit den Problemen der Bewertung und Verbesserung der Robustheit von Deep Learning. Wir schlagen einen Ansatz vor, den wir *Meta Square Attack* nennen, um die Suchverteilung von Black-Box-Zufallssuche-basierten gegnerischen Angriffen auf Deep-Learning-Modelle zu meta-lernen, um die Bewertung der Black-Box-Robustheit zu verbessern. Wir untersuchen zertifizierte Abwehrmaßnahmen gegen gegnerische Patch-Angriffe, die eine garantierte Untergrenze für die Robustheit des Modells in diesem Bedrohungsmodell bieten. Wir schlagen *BagCert* vor, das ein durchgängiges Training und eine effiziente Zertifizierung der Klassifikationsmodelle ermöglicht. Wir schlagen auch einen auf Inpainting basierenden Ansatz namens *Demasked Smoothing* vor, der die erste Methode ist, um die Robustheit semantischer Segmentierungsmodelle gegenüber gegnerischen Patches zu zertifizieren. Demasked Smoothing kann mit willkürlichen Segmentierungsmodellen arbeiten und erfordert kein zusätzliches Training.

Contents

Chapter 1

Introduction

Deep learning is a field of Artificial Intelligence that uses data to train complex computational structures called *deep neural networks* (Schmidhuber, 2015) that are able to achieve required predictive ability. Deep learning has achieved impressive results in different problems of interest such as computer vision (Simonyan and Zisserman, 2014) and natural language processing (Vaswani et al., 2017). Deep neural networks were demonstrated to be vulnerable to small perturbations in the input that are imperceptible to human observer (Biggio et al., 2013; Szegedy et al., 2014). Such perturbations are called *adversarial examples*. Their ”imperceptibility” was originally formalised as having a small ℓ_p norm for some p. The task of finding adversarial examples efficiently is paramount for evaluating model *robustness* which is the ability of the model to maintain the prediction in presence of adversarial examples (Madry et al., 2018b).

Studying deep learning robustness in the presence of ℓ_p-norm bounded adversarial examples is a thriving research direction (Croce et al., 2020a). Nevertheless, such results alone say little about the danger that the adversarial examples pose for the real world deep learning applications (Lu et al., 2017; Eykholt et al., 2018). Therefore, another branch of deep learning robustness has appeared that studies so called *physical world* adversarial examples (Athalye et al., 2018b; Kurakin et al., 2018). One way to make implementing adversarial examples in the real world more feasible is localizing their impact so that they only need to affect a small region of the scene. It was implemented in the threat model called *adversarial patch* (Brown et al., 2017; Karmon et al., 2018). Adversarial patch is not an imperceptible attack in the strict sense of this word because significant modification of the confined pixels can be noticed by a human being. However the size of the patch region is usually small enough (taking up to several percent of the image surface) so that the patch is not making it more difficult for a real person to analyze the image.

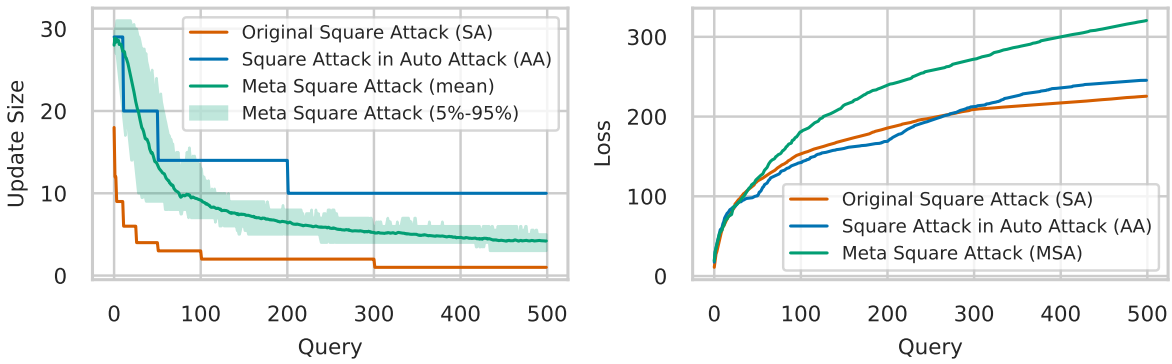
In practice, an attacker only possesses a restricted access to the model information. For example, when the source code of the model training is not publicly available and an attacker can only access the output of a model for a given input. To evaluate model robustness in this case, one considers *black-box* adversarial attacks (Chen et al., 2017a). There is a strong necessity of efficient means to evaluate and improve robustness of deep learning models. Now, we present the contributions that we have made in this work.

The rest of this section is organized as follows. First, we discuss *Meta Square Attack* (Yatsura et al., 2021). We propose a framework for learning to optimize adversarial attacks and apply it to the black-box Square Attack (Andriushchenko et al., 2020). Automating adversarial attacks is a crucial step towards making robustness evaluation more reliable and universal (Jia et al., 2022). Evaluating black-box model robustness is crucial for the real world applications and can be used as an additional mean of evaluation for gradient-based empirical defences (Tramer et al., 2020).

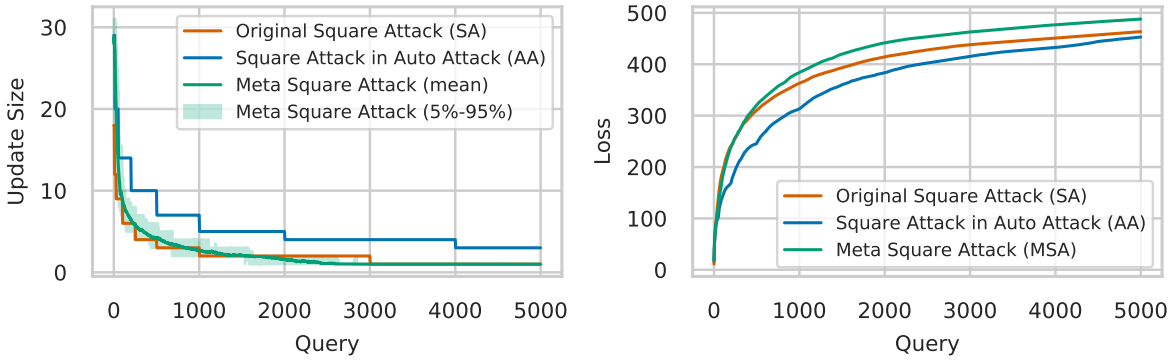
Then, we discuss *BagCert* (Metzen and Yatsura, 2021), a certified defence against adversarial patch attacks on image certification that applies end-to-end training for adversarial robustness. Finally, we propose *Demasked Smoothing* (Yatsura et al., 2023), the first certified defence against adversarial patch attacks on semantic segmentation models. Demasked Smoothing is based on image inpainting and novel masking schemes. It does not require any specific model training and can be applied with any off-the-shelf segmentation models.

Meta Square Attack. This work was published by us as a conference paper at NeurIPS 2021 (Yatsura et al., 2021). Adversarial attacks are often used to obtain an upper bound estimate on the exact robustness of the model. The stronger an attack is, the tighter is the estimate (Croce and Hein, 2020b). However, finding strong adversarial attacks usually requires significant manual design and fine-tuning (Carlini et al., 2019; Andriushchenko et al., 2020). In this work, we look at the adversarial robustness evaluation as a *meta-learning* problem (Finn et al., 2017). Meta-learning field can be described as *learning how to learn* and is closely related to *learning to optimize* (Chen et al., 2021).

A very promising direction in the field of black-box adversarial attacks are randomized search schemes for crafting adversarial examples (Guo et al., 2019; Andriushchenko et al., 2020; Croce et al., 2020b). Combining random search with specific update proposal distributions allows to achieve state-of-the-art black-box efficiency for different threat models such as ℓ_∞ and ℓ_2 (Andriushchenko et al., 2020), ℓ_1 (Croce and Hein, 2021), ℓ_0, adversarial patches, and adversarial frames (Croce et al., 2020b). Despite the conceptual simplicity of these methods, the main



(a) $T = 500$ queries



(b) $T = 5000$ queries

Figure 1.1: (Left) The schedules for SA (Andriushchenko et al., 2020) (which scales accordingly for different query budgets) and AA (Croce and Hein, 2020b) compared to Meta Square Attack (MSA) proposed in this work. MSA adapts the update size during the course of the attack for each image. We illustrate the mean and the percentile range for a given set of attacked images. (Right) The maximization of the loss for the model of Ding et al. (2020) on 100 CIFAR10 images. MSA outperforms SA and AA significantly in terms of the achieved loss.

disadvantage of random search based methods is that the construction of a suitable proposal distribution requires significant manual design and is crucial for competitive performance. Lord et al. (2022) observed, that the parameter choice plays a big role in the efficiency of black-box adversarial attacks, in particular, for the Prior-guided Random Gradient-Free attack (P-RGF) (Cheng et al., 2019).

In this work, we propose a method that allows to circumvent the fine-tuning and reduce the amount of manual design in random search based attacks. We use gradient-based meta-learning to automatically optimize controllers for schedules and proposal distribution on models with white-box access. After meta-training, the controllers can be plugged into a random search

attack substituting manually designed schedules and proposal distributions. Importantly, once meta-trained, the controllers do not require any gradient access and can thus be used in a fully black-box setting and without being affected by gradient obfuscation. We consider the proposed methodology for the case of Square Attack for the ℓ_∞ and ℓ_2 threat models (Andriushchenko et al., 2020). We meta-train controllers for update size and color on an adversarially trained (Madry et al., 2018a) ResNet18 (He et al., 2016a) model with white-box access and apply them to many different models from the RobustBench model zoo (Croce et al., 2020a) that we treat as black-boxes. Black-box attack setting is defined by the query budget i. e. the number of queries to the model that the attack is allowed to make (Guo et al., 2019). Since query efficiency is of crucial importance in black-box adversarial attacks, we also study the method for different query budgets ranging from several hundreds to several thousands. Depending on the query budget and the attacked model we obtain up to 20% improvement with respect to the baseline schedules proposed by Andriushchenko et al. (2020) and Croce and Hein (2020b).

We describe our work in detail in Chapter 3. In short, we make the following contributions:

- We frame adversarial attack optimization as a meta-learning problem.
- We formalize the gradient-based meta-learning for the Square Attack (Andriushchenko et al., 2020) and propose Meta Square Attack.
- We meta-train Meta Square Attack (MSA) on a CIFAR10 (Krizhevsky and Hinton, 2009) model with white-box access and show that MSA improves robust accuracy estimate by up to 5.6% on a vast range of CIFAR10 models with black-box access with respect to the hand-designed search distributions proposed in previous work (Andriushchenko et al., 2020; Croce and Hein, 2020b) for the ℓ_∞ and ℓ_2 threat models.
- We show that Meta Square Attack generalizes well to different datasets and to the targeted attack setting. It achieves up to 20% better robust accuracy compared to the state-of-the art baseline (Andriushchenko et al., 2020) for attacking models on CIFAR100 and ImageNet.

BagCert. This work was published by us as a conference paper at ICLR 2021 (Metzen and Yatsura, 2021). Previously, we have formulated the notion of *adversarial patch*. In this threat model, an attacker can freely control a small subregion of the input (the “patch”) but needs to leave the rest of the input unchanged. It is relevant because adversarial patch is a physically realizable attack (Lee and Kolter, 2019b). Moreover, once an attacker has generated a successful patch pattern, this pattern can be easily shared, will be effective against all systems using the

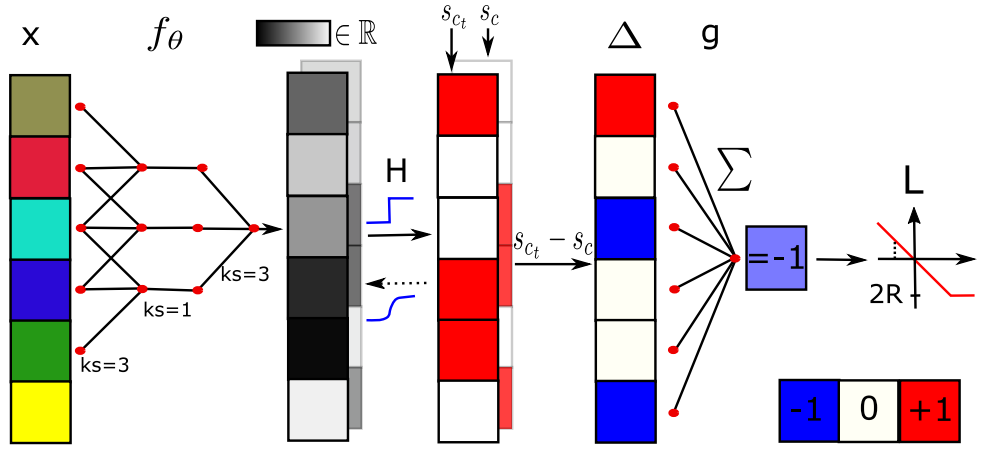


Figure 1.2: Illustration of BagCert training for a 1D input and two classes. An input X is processed by region scorer f_θ, consisting of a 3-layer CNN with kernel sizes (ks) 3, 1, and 3. The resulting continuous region scores are passed through a Heaviside step function (replaced by a sigmoid in the backward pass) to obtain binary region scores s for every class. The differences Δ between true and non-true class scores are then processed by spatial aggregation g, in this case simply summing them via $g = g_{\sum}$. The resulting value is maximized by passing it into margin loss L.

same perception component, and an attack can be conducted without requiring access to the individual system. For instance, this makes attacking an entire fleet of cars of the same vendor feasible.

Empirical defences against adversarial patches can be broken by newly developed attacks, thus the need for certified defences arises. Ideally, a certified defense should combine high certified robustness with efficient inference while maintaining strong performance on clean inputs. Moreover, the training objective should be based on the certification problem to avoid post-hoc calibration of the model for certification (Levine and Feizi, 2020).

Previously proposed defenses do not satisfy all of these conditions. In this work, we propose BagCert, which combines high certified accuracy (60% on CIFAR10 for 5×5 patches occupying approximately 2.4% of the image surface) and clean performance (86% on CIFAR10), efficient inference (43 seconds on a single GPU for the 10.000 CIFAR10 test samples), and end-to-end training for robustness against patches of varying size, aspect ratio, and location. BagCert (see Chapter 4 for a detailed discussion and experiments) is based on the following contributions:

- We propose three different conditions that can be checked for certifying robustness. One of these corresponds to the condition proposed by Levine and Feizi (2020). However, we show that an alternative condition improves certified accuracy of the same model typically

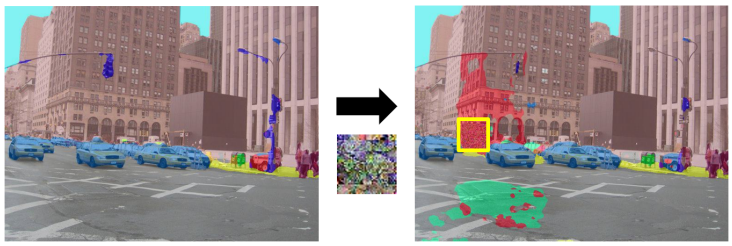


Figure 1.3: A simple patch attack on the ViT-based Swin Liu et al. (2021) manages to switch the prediction for a part of the image.

(a) input with a patch (b) patch not masked (c) patch masked

Figure 1.4: Illustration of masking an adversarial patch in the image.

by roughly 3 percent points while remaining broadly applicable.

- We derive a loss function that directly optimizes for certified accuracy against a uniform distribution of patch sizes at arbitrary positions. This loss corresponds to a specific type of the well known class of margin losses.
- Similarly to Levine and Feizi (2020), we classify images via a majority voting over a large number of predictions that are based on small local regions of a single input. However, the proposed model achieves this via a single forward-pass on the unmodified input, by utilizing a neural network architecture with very small receptive fields, similar to BagNets (Brendel and Bethge, 2019). This enables efficient inference with surprisingly high clean accuracy and was concurrently proposed by Zhang et al. (2020b) and Xiang et al. (2021).

Demasked Smoothing. This work was published by us as a conference paper at the Eleventh International Conference on Learning Representations, ICLR 2023 (Yatsura et al., 2023). As can be seen in Section 2.3 of the overview, certified defences against patch attacks were mostly proposed for the image classification problems. While several defences were proposed for object detection (Xiang and Mittal, 2021a; Xiang et al., 2022b) as well, the ways to certify robustness of semantic segmentation models (Zhao et al., 2017) were not available before our work.

Previous work shows that end-to-end training of the model for certified robustness can be highly

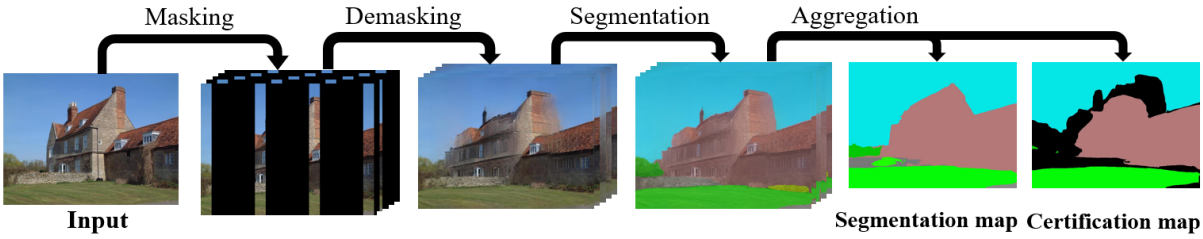


Figure 1.5: A sketch of Demasked Smoothing for certified image segmentation. First, we generate a set of masked versions of the image such that each possible patch can only affect a certain number of masked images. Then we use image inpainting to partially recover the information lost during masking and then apply an arbitrary segmentation method. The output is obtained by aggregating the segmentations pixelwise. The masking strategy and aggregation method depend on the certification mode (certified detection or certified recovery).

beneficial for performance, for example, in *BagCert* (Metzen and Yatsura, 2021). However, in some cases, specific training or fine-tuning of the model may be too expensive or undesirable. It may especially be the case for the semantic segmentation task which has to provide a much richer dense output than image classification or object detection and, thus, requires more computational resources to train a model.

We propose a way to resolve the two abovementioned issues by introducing *Demasked Smoothing*, the first method to certify the robustness of semantic segmentation models that relies on off-the-shelf inpainting and segmentation models and does not require any specific training. Similarly to previous work (Levine and Feizi, 2020), we mask different parts of the input and provide guarantees with respect to every possible patch that is not larger than a certain predefined size. While prior work required the classification model to deal with such masked inputs, we leverage recent progress in image inpainting (Dong et al., 2022) to reconstruct the input *before* passing it to the downstream model. This decoupling of image demasking from the segmentation task allows us to support arbitrary downstream models. Moreover, we can leverage state of the art methods for image inpainting. We also propose different masking schemes tailored for the segmentation task that provide the dense input allowing the demasking model to understand the scene but still satisfy the guarantees with respect to the adversarial patch. See detailed discussion and experimental results in Chapter 5.

We summarize our contributions as follows:

- We propose Demasked Smoothing which is the first certified recovery or certified detection based defence against adversarial patch attacks on semantic segmentation models.
- Demasked Smoothing can do certified detection and certified recovery with any off-the-

shelf segmentation model without requiring finetuning or any other adaptation.

- We implement Demasked Smoothing, evaluate it for different certification objectives and masking schemes. We can certify 65% of all pixels in certified detection for a 1% patch and 47% in certified recovery for a 0.5% patch for the BEiT-L (Bao et al., 2022) segmentation model on the ADE20K Zhou et al. (2017) dataset.

Chapter 2

Background and Related Work

2.1 Deep Learning and its Applications

Machine learning is a research field studying the models based upon sample data that can make predictions or decisions without being explicitly programmed to do so (Samuel, 1959). Being a part of a broader artificial intelligence field, machine learning is tightly connected to the disciplines of computational statistics and optimization. Numerous approaches to build and train machine learning models were proposed in the past decades such as decision trees (Quinlan, 1987) or support vector machines (Hearst et al., 1998). However, thanks to the breakthrough in computational technologies, *deep learning* became the most thriving and fruitful machine learning branch of the last years (LeCun et al., 2015).

Deep learning uses sample data to train complex computational structures called *deep neural networks* (Schmidhuber, 2015) to achieve required predictive ability. *Training* neural networks is the optimization process that adjusts their parameters to perform well on unseen data using methods like gradient descent and its variations. The ability to perform well on data not seen during training is called *generalization* (Goodfellow et al., 2016).

Deep learning has achieved impressive results in different problems of interest such as computer vision (Simonyan and Zisserman, 2014) and natural language processing (Vaswani et al., 2017). Deep learning based methods even managed to beat human players in complex intellectual games (Silver et al., 2016) which seemed impossible with more primitive manually designed approaches.

2.2 Evaluating Deep Learning Robustness

2.2.1 Adversarial Examples

Despite their great success in different tasks, deep neural networks were demonstrated to be vulnerable to small perturbations in the input that are imperceptible to human observer (Biggio et al., 2013; Szegedy et al., 2014). Such perturbations are called *adversarial examples*.

Let us define adversarial examples for the image classification problem. Let $\mathcal{X} \subset [0,1]^{H \times W \times C}$ be a set of images with height H, width W and the number of channels C. Usually in practice $C = 1$ (grayscale images) or $C = 3$ (RGB images). Let $\mathcal{Y}$ be a set of classes such that each image $x \in \mathcal{X}$ belongs to one and only one of these classes. For example, these could be handwritten digits (LeCun et al., 1998) or common objects (Krizhevsky, 2009). We assume that there are K classes and each of them is encoded as a number from 0 to $K-1$: $\mathcal{Y} = \{0, 1, \dots, K-1\}$. We define $F : \mathcal{X} \to \mathcal{Y}$ to be a *classification model* (or simply *classifier*) which assigns a class to an input image.

Consider a tensor $\delta \in \mathbb{R}^{H \times W \times C}$. Let $\mathcal{S}$ be a set of *imperceptible* additive image perturbations. "Imperceptible" means that if $\delta \in \mathcal{S}$, then for an image $x \in \mathcal{X}$ a human observer cannot easily distinguish between x and a perturbed image $x' := x + \delta$. One choice of a set $\mathcal{S}$ is a ball

$$\mathcal{B}(p, \epsilon) := \{\delta \mid \|\delta\|_p < \epsilon\} \tag{2.1}$$

for some ℓ_p norm and some radius $\epsilon \in \mathbb{R}^+$. Then $\forall\ \delta \in \mathcal{B}(p, \epsilon)\ \|x' - x\|_p = \|\delta\|_p < \epsilon$ and for a small enough ϵ the perturbation δ is imperceptible.

Let $x \in \mathcal{X}$ be an image. We say that δ is an *adversarial example* if δ is imperceptible and $F(x) \neq F(x + \delta)$. It means that although the perturbation δ is practically invisible for a human observer, it drastically changes the classification model output.

Such behavior raises significant concerns for the real world deployment of deep learning approaches in safety-critical domains such as autonomous driving (Ranjan et al., 2019a). Artificial neural networks were inspired by the real neuronal structures in human brain. Human perception was not observed to be susceptible to adversarial examples. Therefore, apart from obvious security issues, such examples serve as a tool to better understand the true nature of artificial neural networks as well as to capture their inherent properties and differences from their biological counterparts (Ilyas et al., 2019).

Albeit the existence of adversarial examples was confirmed (Biggio et al., 2013; Szegedy et al.,

2014), establishing the procedure to find them efficiently remains an open problem (Carlini and Wagner, 2017b). However, this task is paramount for evaluating model *robustness* which is the ability of the model to maintain the prediction in presence of adversarial examples (Madry et al., 2018b). Only by measuring this ability we can establish whether a model can be deployed in safety-critical domains that require reliable and trustworthy performance such as medical imaging or autonomous driving. The process of finding an adversarial example is often called an *adversarial attack*.

A straightforward way to check the existence of adversarial examples for a given model is to apply brute-force and simply check all possible imperceptible perturbations. However for high-dimensional data such as visual images, the dimension of a possible additive pixel-wise perturbation is also high. Thus, directly checking all the elements of the perturbation set $\mathcal{S}$ becomes prohibitively expensive and leads to a combinatorial explosion for natural images (Katz et al., 2017).

Complete verifiers. Nonetheless, a line of work exists that studies the procedures to evaluate worst-case adversarial robustness of a model *exactly*. These are the so called *complete verifiers* (Ehlers, 2017).Katz et al. (2017) propose Reluplex, a robustness verification method based on Satisfiability Module Theory. They consider deep neural networks with several hundreds ReLU activations proposed for the aircraft collision avoidance systems. Tjeng et al. (2018) use Mixed Integer Linear Programming to achieve the verification speed several orders of magnitude faster than Reluplex and were able to verify residual networks (He et al., 2016c) with over 100,000 neurons on MNIST (LeCun et al., 1998). However, contemporary ImageNet (Krizhevsky et al., 2017) classification models often have more than one billion parameters (Zhai et al., 2022). Therefore, evaluating them with fundamentally combinatorial methods is not feasible and one requires robustness evaluation methods that scale to larger neural networks.

One way is to use approximate methods and find an *upper bound* on the true robust accuracy of the model. These adversarial attacks are usually based on computational optimization methods (Madry et al., 2018b).

We define $f : \mathcal{X} \to \Delta^{K-1}$ where

$$\Delta^{K-1} = \Big\{(p_0, ..., p_{K-1}) \in \mathbb{R}^K \Big| \sum_{i=0}^{K-1} p_i = 1, \text{ and } p_i \geq 0 \text{ for } i = 0, ..., K-1\Big\}, \tag{2.2}$$

denotes the set of probability distributions over the K classes of $\mathcal{Y}$. Then we can define

$$F(x) := \underset{c \in \{0,\dots,K-1\}}{\arg\max} f_c(x) \tag{2.3}$$

i. e. our classifier predicts the class with the highest score. Let the model $f(\theta, \cdot)$ be parameterized by a vector θ. Let us define the loss function $L(f,\ \theta,\ x,\ y)$ that is a measure of performance of a classification model f with parameters θ on an input image x with the ground truth label y. The loss function is usually used for the gradient-based model training and serves as a differential proxy for a non-differential indicator function

$$L_I(f,\ \theta,\ x,\ y) := [\underset{c \in \{0,\dots,K-1\}}{\arg\max} f_c(\theta, x) = y] \tag{2.4}$$

This function is discrete and thus does not have a meaningful gradient that can be used in training. One popular choice for the loss is the cross-entropy function which is smooth

$$L_{CE}(f,\ \theta, x,\ y) := -\log(f(\theta, x))_y \tag{2.5}$$

However there are other possibilities. The goal of the model training is to find a vector θ with which the model achieves good classification performance. Let $\mathcal{D}$ denote a probability distribution defined on the set of image-label pairs $\mathcal{X} \times \mathcal{Y}$. Then we need to solve

$$\min_{\theta} \underset{(x,\ y) \sim D}{\mathbb{E}} L(f, \theta,\ x,\ y) \tag{2.6}$$

Due to the highly non-linear nature of modern neural architectures (He et al., 2016c), solving this problem exactly is typically not feasible. Nevertheless, an approximate solution can be found using numerical optimization methods such as gradient descent or its variants

$$\theta := \theta - \alpha \nabla_\theta \underset{x \sim D}{\mathbb{E}} L(f, \theta,\ x,\ y) \tag{2.7}$$

with some optimization step α.

When the model is already trained we assume the vector θ to be constant. We use the notation $f(x)$ and $L(f,\ x,\ y)$ instead of $f(\theta, x)$ and $L(f,\ \theta,\ x,\ y)$ implicitly assuming the parameter θ. To find an adversarial example we want to find a perturbation δ such that $F(x + \delta) \neq F(x)$. Optimizing an indicator function with gradient-based methods is still unfeasible, thus we again consider the loss function optimization instead. But now we are interesting in *maximizing* the

loss instead of minimizing it as we did for the model training

$$\max_{\delta} L(f,\ \theta,\ x+\delta,\ y) \tag{2.8}$$

For this, we can use $\nabla_\delta L(f,\ x+\delta,\ y)$. This is the core idea of gradient-based adversarial attacks (Goodfellow et al., 2015; Madry et al., 2018b).

Originally adversarial perturbations were proposed for the ℓ_∞ norm (Szegedy et al., 2014) and even now this threat model is considered to be one of the principal ones in the field of adversarial robustness (Croce et al., 2020a). It assumes the pixel-wise constraint on the perturbation δ:

$$\max_{1\leq i\leq H,\ 1\leq j\leq W,\ 1\leq k\leq C} |\delta_{i,j,k}| \leq \epsilon \tag{2.9}$$

Fast Gradient Sign Method (FGSM) proposed by Goodfellow et al. (2015) for the ℓ_∞ adversarial attacks constructs an adversarial example x' as

$$x' = x + \epsilon \cdot \text{sign}(\nabla_x L(f,\ x,\ y)) \tag{2.10}$$

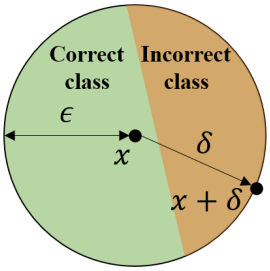
See a 2-D schematic illustration in Figure 2.1a. This single-step procedure allows to create adversarial examples without significant computational overhead. One must ensure that after this modification x' still remains an image and has its pixel values in range $[0,\ 1]$. Thus, we consider the projection operator Π_R that projects each pixel to a given range R. The final adversarial example is $\Pi_{[0,\ 1]}(x')$.

Madry et al. (2018b) propose an iterative optimization procedure for finding adversarial examples called projected gradient descent (PGD). PGD is parameterized by the number of iterations T and the step size α:

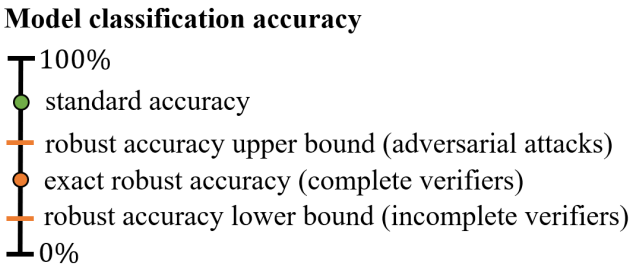
$$\begin{aligned} x^0 &= x \\ x^{t+1} &= \Pi_{(x+\mathcal{B}(p,\epsilon))\cap[0,1]^{H\times W\times C}}(x^t + \alpha \cdot \text{sign}(\nabla_{x^t} L(f,\ x^t,\ y))), \end{aligned} \tag{2.11}$$

where $x + \mathcal{B}(p,\epsilon) = \{x+\delta \mid \delta \in \mathcal{B}(p,\epsilon)\}$.

Croce and Hein (2020b) propose Auto-PGD (APGD) modification of PGD that allows budget-aware dynamic choice of the step size. They also proposed a shift and rescaling invariant loss called difference of logits ratio (DLR). They combine APGD with Fast Adaptive Boundary attack (Croce and Hein, 2020a) and Square Attack (Andriushchenko et al., 2020) to obtain *AutoAttack*, a parameter-free ensemble of adversarial attacks for reliable evaluation of model robustness.



(a) FGSM (Goodfellow et al., 2015)



(b) Robustness evaluation methodologies

Figure 2.1

Carlini and Wagner (2017b) suggest a different approach to robustness evaluation. Instead of looking for an adversarial example in a given ball $\mathcal{B}(p, \epsilon)$ they look for an adversarial example which is *as close as possible* to the image x. They introduce a distance metric $D(u, v)$ which can be e. g. a distance in some ℓ_p norm: $D(u, v) = \|u - v\|_p$. They also denote C to be an objective function such that $C(u) = t$ means that u is an adversarial example. Thus their problem is formulated as

$$\begin{aligned} \text{minimize} \quad & \mathcal{D}(x, x + \delta) \\ \text{such that} \quad & C(x + \delta) = t, \\ & x + \delta \in [0, 1]^{H \times W \times C} \end{aligned} \tag{2.12}$$

Incomplete verifiers. We have already discussed that complete verifiers provide the true value of robust accuracy for a given model, and adversarial attacks provide an approximate upper bound on it. There is a third option of deep learning robustness evaluation, namely estimating the *lower bound* on the model robustness. A trivial lower bound of 0% accuracy (which can actually be tight in some cases (Andriushchenko et al., 2020)) always exists. Hence, one is interested in having a lower bound estimate which is as close to the true model robustness as possible. The methods to achieve this are called *incomplete verifiers* (Wong and Kolter, 2018; Gowal et al., 2018). Having a non-trivial lower bound estimate usually requires training the model with a specific procedure, thus we discuss these methods in detail in Section 2.3. We summarize the schematic relation between different approaches to deep learning robustness evaluation in Figure 2.1b.

An important notion of adversarial robustness field are the so called *targeted* adversarial attacks (Szegedy et al., 2014). Instead of finding $\delta \in \mathcal{S}$ such that $F(x + \delta) \neq F(x)$, we are interested in having $F(x + \delta) = y_t$ for some target class y_t. Therefore, instead of solving 2.8, we consider

$$\min_{\delta} L(f,\ \theta,\ x + \delta,\ y_t) \tag{2.13}$$

Targeted adversarial attacks can pose a significant threat since they allow an attacker to target the most critical classification label. For example, by forcing an autonomous driving system to always classify the speed limit shown on a traffic sign as the highest possible value.

Given the popularity of deep convolutional neural networks (CNN) in the recent years (Krizhevsky et al., 2017), their robustness was evaluated extensively (Croce et al., 2020a). Nevertheless, other neural architectures such as Vision Transformers (Dosovitskiy et al., 2021) were also found vulnerable to adversarial examples (Bai et al., 2021). Although in this discussion we were mostly concentrated on the classification problem, adversarial examples exist for other computer vision tasks such as object detection (Xie et al., 2017) or semantic segmentation (Arnab et al., 2018).

2.2.2 Physical World Robustness

Studying deep learning robustness in the presence of ℓ_p-norm bounded adversarial examples is a thriving research direction (Croce et al., 2020a). Nevertheless, such results alone say little about the danger that the adversarial examples pose for the real world deep learning applications (Lu et al., 2017; Eykholt et al., 2018). Modifying image pixels can be easily done if one has access to the digital version of the image provided to the model. However, such scenarios are rare in the real world and potentially require access to the software running on deployed devices. Therefore, another branch of deep learning robustness has appeared that studies so called *physical world* adversarial examples (Athalye et al., 2018b; Kurakin et al., 2018). These are the perturbations made in the real world rather than in the digital world. For example, it can be a modification of the scene captured by the perceptual component of the deep learning system (Lee and Kolter, 2019a) or modification of the perceptual component itself such as placing a transparent film on the surface of a camera lens (Li et al., 2019).

2.2.3 Expectation over transformation

It was shown that adversarial examples optimized for the digital world lose their efficiency when the real world factors such as different viewpoints or angles come into play (Lu et al., 2017). To study physical adversarial examples Athalye et al. (2018b) propose *Expectation Over Transformation* (EOT) method. They consider a distribution T of expected transformations $t(\cdot)$ such as rotation, translation or brightness change. For a given distance function $d(\cdot,\ \cdot)$ they perform a

targeted adversarial attack

$$\begin{aligned} \arg\max_{x'} \quad & \mathop{\mathbb{E}}_{t\sim T}[\log P(y_t|t(x'))] \\ \text{such that} \quad & \mathop{\mathbb{E}}_{t\sim T}[d(t(x'),\ t(x))] < \epsilon \\ & x' \in [0,1]^{H\times W\times C} \end{aligned} \tag{2.14}$$

They study 2-D and 3-D adversarial examples and demonstrate that they pose a threat in the real world even when captured from different viewpoints and angles.

2.2.4 Universal adversarial perturbations

One assumption made in the Section 2.2.1 is that we create an adversarial example to affect the prediction of a classifier f on a single image $x \in \mathcal{X}$. A model f deployed in the real world usually has to process many different images and providing a specified adversarial modification for each of them drastically reduces the feasibility of adversarial attacks in this setting. Moosavi-Dezfooli et al. (2017) proposed *universal* adversarial perturbations (UAP). In case of UAP, the same perturbation has to be able to fool a given model on as many different images as possible. Let $\mathcal{D}$ be a probability distribution defined on the set of images $\mathcal{X}$. For a fixed $\alpha \in [0,\ 1]$ the goal is to find δ for which:

$$\delta \in \mathcal{S} \quad \text{and} \quad \mathop{\mathbb{P}}_{x\sim D}(F(x+\delta) \neq F(x)) \geq 1-\alpha \tag{2.15}$$

Moosavi-Dezfooli et al. (2017) propose an iterative algorithm over the images in a given set for crafting UAPs for the classification task. If δ does not shift the prediction on a current image, they compute the minimal perturbation Δv that achieves the desired shift and add it to δ. Other approaches exploit singular vectors (Khrulkov and Oseledets, 2018) or generative networks (Mopuri et al., 2018). When the universal perturbation is obtained, it can be realized in the form of a semi-transparent sticker and placed on the camera lens to be applied to all of the camera inputs (Li et al., 2019). Metzen et al. (2017a) have gone beyond the classification task and proposed UAP for semantic segmentation problem. They studied two objectives for an attacker: making the model always output a static target segmentation regardless of the input and blending certain classes (such as pedestrians in the autonomous driving task). Xu and Singh (2022) demonstrated that standard UAP fail to be adversarial under the real world transformations. They applied a specific form of EOT for optimizing them to make UAP robuster.

2.2.5 Black-box adversarial attacks

Standard adversarial perturbations assume to have full knowledge of the model architecture and access to the model weights (Szegedy et al., 2014; Madry et al., 2018b). This is required for computing exact model gradients 2.10 for the adversarial optimization problem Goodfellow et al. (2014b). When internal characteristics of the target model are known to the attacker, we call it a *white-box attack*. However, in practice an attacker often does not have such a broad access to the model information. For example, when the source code of the model training is not publicly available and an attacker can only access the output of a model for a given input (Zhou et al., 2022). To evaluate model robustness in this case, one considers *black-box* adversarial attacks (Chen et al., 2017a).

Transfer-based black-box attacks. Even though full access to the target model is not possible, the limitations on this access may vary. For example, an attacker might be able to obtain a model with the white-box access which is similar to the target one. We call it a *surrogate model*. In this case, the attacker can generate adversarial examples for a surrogate model with white-box approaches and apply them to the target one. This approach is called *transfer attack* (Papernot et al., 2016; Cheng et al., 2019; Yuan et al., 2021). Please note that the word *"transfer"* here is meant in the sense of transferring an image-specific adversarial example for some image between different models. In the previous section, universal adversarial perturbations were a way to transfer the same adversarial perturbation for one model between different input images.

A downside of transfer attacks is that the similarity between the target and the surrogate model is a critical factor. Without that the efficiency of transfer attacks is drastically reduced (Huang and Zhang, 2020). Papernot et al. (2017) train a substitute model for performing transfer attacks on the data, annotated by querying the target model. Liu et al. (2017) perform large-scale analysis of adversarial examples transferability between different datasets and models. In particular, they study the transfer of targeted attacks for the first time. They propose an approach to facilitate the transfer of targeted adversarial examples. Zhang et al. (2022a) argue that the standard Attack Success Rate (ASR) is not a suitable metric to evaluate the transferability of adversarial examples and suggest to use the Top-k predictions instead.

To perform a transfer attack one does not need to perform many queries to the target model. However the requirement of having a surrogate model with sufficient level of similarity is rather strong and cannot be fulfilled in many cases. If we only have a target model, then our only option is use it as a black box to construct an adversarial example. As said above, it means that we can provide input to the model and observe its output. However, depending on the form of

this output, black-box attacks are further divided into decision-based and score-based.

Decision-based attacks (Brendel et al., 2018; Chen et al., 2019; Li et al., 2020b) for image classifiers assume that one can submit query images x to the model and receive only the predicted class label $F(x)$. This is assumed to be a harder case for an attacker since the information available to it is very limited. Brendel et al. (2018) propose Boundary Attack which starts with a perturbation with big magnitude that fools the model and gradually reduce this magnitude while keeping the property of fooling the model. Pointwise attack (Schott et al., 2018) considers the L_0 perturbation norm. This type of attack also called "sparse" allows modifying only a limited number of the image pixels. PGD_0 (Croce and Hein, 2019) proposes a PGD modification for a black-box sparse attack. Shi et al. (2022) propose a decision-based attack on Vision Transformers (ViT) (Dosovitskiy et al., 2021) based on Patch-wise adversarial removal (PAR) to reduce the magnitude of the generated adversarial noise while keeping the number of queries to the model. SparseEvo (Vo et al., 2022) applies evolutionary strategies to achieve a query-efficient sparse decision-based attack.

Score-based attacks (Chen et al., 2017a; Ilyas et al., 2018; Alzantot et al., 2019) consider a richer form of the model output. For a query image x one obtains a vector of probabilities (or scores) $f(x)$ assigned to each class. Vector $f(x)$ can be accessed e.g. in Google Cloud Vision API. One approach is to use the scores to evaluate the model gradient using numerical methods and apply gradient ascent 2.11 similarly to the white-box case (Chen et al., 2017a; Uesato et al., 2018). Alzantot et al. (2019) propose a derivative-free approach based on genetic algorithms. Guo et al. (2019) use a simple search procedure based on iteratively adding vectors from an orthonormal vector set to the perturbation and checking whether it improves the objective. Several works use an empirical observation that successful ℓ_∞ perturbations are located at the corners of the color cube, therefore instead of continuous optimization one can reduce the problem to a discrete one (Meunier et al., 2019; Al-Dujaili and O'Reilly, 2020).

Combining transfer-based and query-based black-box attacks. Another sub-field of the black-box adversarial attacks considers a hybrid of the two abovementioned approaches. It suggests to enhance the query-based attacks by using the prior from the surrogate models (Cheng et al., 2019; Huang and Zhang, 2020). Our method Meta-Square Attack (MSA) described in the Chapter 3 has the highest similarity with this group of hybrid methods. However, unlike most of them it uses a model with the white-box access not during each particular attack episode but rather during the attack pre-training making the dependence on the white-box model less prominent as will be discussed in the Chapter 3.

Cheng et al. (2019) propose *Prior Random Gradient-free* (P-RGF) attack that uses prior from the surrogate model to sample directions for the RGF gradient estimation approach instead of using random samples. Lord et al. (2022) show how gradient transfer can be combined with the query-based methods to achieve an efficient black-box attack. They try to craft an adversarial perturbation by using the gradient of the surrogate models and, when it fails, use the random sampling based on the Output Diversified Sampling (Tashiro et al., 2020). They also argue that the choice of attack parameters can have a huge effect on the attack efficiency. This supports our hypothesis that query-based black-box attacks can benefit a lot from meta-learning their parameters. Black-box Attacks via Surrogate Ensemble Search (BASES) approach (Cai et al., 2022a) performs the search in the model ensemble space by optimizing the weights of the different models in the loss and updating one weight at a time. Since the search happens in the low-dimensional model weights space instead of the high-dimensional image space or the image-embedding space, on average only a few queries are enough to fool the target model. However, a diverse ensemble is still a key prerequisite for the attack success.

Black-box attacks on dense prediction tasks. Cai et al. (2023) consider the problem of black-box attacks on object detection and semantic segmentation. They argue that the losses, optimized during the training of the dense prediction models, are more complex than the ones applied in a more studied classification setting. Therefore, they propose weight balancing and weight optimization approaches to account for the different scale of the losses and improved the results of the attacks based on an ensemble of surrogate models. Gu et al. (2021) study the transfer of adversarial examples between segmentation models.

Cai et al. (2022b) study how the context of the objects can be explored to craft strong transfer adversarial attacks on object-detection. Instead of focusing on affecting the model prediction for the target object, their attack aims to affect all the objects present in the image. Or even introduce it's own helper objects to assist the final goal. To grasp the connection between different object classes, the method is constructing context graphs based on some available dataset. Using the context allows to significantly improve the performance over the baseline. Although building the context graph might constitute a significant overhead, the authors demonstrate that, surprisingly, trying to perturb other objects to random classes still provides a significant improvement.

Parallel Rectangle Flip Attack (Liang et al., 2022b) splits the image into rectangles and generates an adversarial perturbation in each of them using random search to affect multiple bounding boxes in the object detection task. Their results emphasize the importance of random search-

based methods in the black-box robustness evaluation. GARSDC (Liang et al., 2022a) uses genetic algorithms for the black-box attacks on object detection. It formulates the attack objective to not only minimize True Positive but also maximise False Positive.

Xie et al. (2021) considered universal 3-D black-box attacks on video recognition systems. The research done on black-box attacks for the dense prediction tasks demonstrates interesting aspects of the problems and suggests further work on exploring the effects of the image context. Li et al. (2021) study black-box adversarial attacks for the video classification problem. Given the temporal dependence in the gradient, they reduce the search space by exploiting geometric transformations.

Novel directions in black-box adversarial attacks. Hong and Hong (2023) propose a novel line of research allowing to certify the success of black-box attacks. Their method is based upon the randomized smoothing methodology (Cohen et al., 2019) applied not to the clean images but to the adversarial examples. Certified black-box attacks may be a promising direction for future work allowing to make the performance of black-box attacks more predictable. Dynamic Substitute Training (DST)(Wang et al., 2022) considers the *data-free* black-box attack setting in which the attacker has no information not only on the model architecture and weights, but also on the training data and even the number of classification categories. It suggests to train a substitute model with dynamic structure to better handle attacking different target models.

Sun et al. (2022) propose a novel setting of black-box attacks, namely the *lightweight* attacks that operate in the context of no-box threat model proposed by Chen et al. (2017a). In this setting the attacker only has access to a limited number of samples for which the target model makes high-confidence predictions. For example, one sample per classification category. In the unrestricted attacks the image can be significantly modified in a semantically meaningful way (Laidlaw and Feizi, 2019). Natural Color Fool (NCF) (Yuan et al., 2022) is a novel unrestricted color attack. The authors show how the transferability of this type of attacks can be improved by exploiting color distributions of different semantic classes in the image.

The works mentioned in this secion mostly do the evaluation on the locally stored models for which they assume black-box access, only some works consider truly black-box scenarios such as cloud services. Wu et al. (2022) consider this specific case and make extensive evaluation of black-box attacks on cloud APIs. Further investigation of the black-box attacks in the real-world setting would be a promising direction.

2.2.6 Square Attack

Andriushchenko et al. (2020) propose *Square Attack* (SA), a random-search based method with carefully crafted update rule that efficiently finds adversarial examples in the black-box setting. It even achieves a better robustness estimate on the MNIST (LeCun et al., 1998) dataset than white-box approaches (Madry et al., 2018b). Previously, Guo et al. (2019) used a simple random search scheme for the ℓ_2 threat model that uses a set of update directions based on the discrete cosine transform. SA combines classical random search with heuristic design of the update rule that allows to significantly improve the performance. This design depends on the geometry of the perturbation set and therefore differs for ℓ_∞- and ℓ_2-attacks and was later extended also to ℓ_1 (Croce and Hein, 2021).

As the adversarial attack problem is highly non-convex, a good initialization can significantly improve query efficiency. SA uses a stripe initialization motivated by empirical findings (Yin et al., 2019). The design principle of the attack is based on the observation that the strongest perturbations are usually found on the boundary of the feasible set (Seungyong et al., 2019). For the ℓ_∞-case, updates are sampled as squares parametrized by square size, square color, and position of the square. Performance of SA depends heavily on how color, position, and square size are chosen, which requires manual design.

The square size schedules employed by previous work are relatively sophisticated (indicating non-trivial manual design): Andriushchenko et al. (2020) proposed a schedule parametrized by $p^0 \in [0, 1]$ (the fraction of image pixels to be modified by a square in the first query) and the total query budget T, where p^t is halved at $\{0.1, 0.5, 2, 10, 20, 40, 60, 80\}$% of the total query budget. For evaluation on CIFAR10, Andriushchenko et al. (2020) suggest different values of p^0 and report $p^0 = 0.3$ as a default choice. Croce and Hein (2020b) proposed a different schedule for SA to be used in AutoAttack: they use $p^0 = 0.8$ and $T = 5000$ but fix the halving points of p^t as if $T = 10000$. An illustration of the two schedules for $T = 500$ and $T = 5000$ can be seen in Figure 1.1.

The second aspect that characterizes SA is the distribution from which position and color of the next square are sampled. For positions, the distribution is uniform over all positions for which a square of a given size would be fully contained in the input image. For colors, SA always generates points on the boundary of the perturbation set (corners of the color cube) and uses a uniform distribution over the 2^c different colors with c being number of channels. Typically we consider RGB images for which $c = 3$, so there are 8 different options.

Thus SA either relies on extensive manual design (for the square size schedules) or resorts

to simple baseline choices (such as the uniform distributions over colors and positions) which might be suboptimal. In this work, we show that meta-learning SA consistently improves the already strong performance of SA (see Section 3.2.2) with little manual design and identify non-trivial patterns that increase attack efficiency (see Section 3.2.3).

2.2.7 Meta-learning and adversarial robustness

The closely related fields of meta-learning (Hospedales et al., 2021) and learning to optimize (Chen et al., 2021) have been employed in the field of adversarial robustness. The idea of learned optimizers (Andrychowicz et al., 2016; Chen et al., 2017b) was applied to finding adversarial examples in white-box (Xiong and Hsieh, 2020) and black-box (Ruan et al., 2020) settings. Meta-learning (Finn et al., 2017; Nichol et al., 2018) was also used to improve zeroth-order gradient estimation and allow better query efficiency of black-box attacks (Du et al., 2020). Besides that there is the recent work (Yao et al., 2021) on automating the existing AutoAttack (Croce and Hein, 2020b) framework for robustness evaluation. Meta-learning has also been used as a part of adversarial training (Xiong and Hsieh, 2020) for increasing adversarial robustness.

Meta-learning was applied in the black-box attack setting. Yuan et al. (2021) proposed Meta Gradient Adversarial Attack (MGAA) concurrently with our MSA work (Chapter 3]). MGAA is a black-box attack inspired by the idea of meta-learning in which a perturbation is crafted as a sequence of tasks. In each of them, first meta-training phase is performed by attacking the models from a pre-defined model zoo with white-box attacks. Then in the meta-testing phase the black-box attack is used. Yin et al. (2022) introduce meta generator that produces the perturbations based on the history of attacks as well as the feedback from the current attack task. They train their meta generator on a surrogate model with white-box access.

2.2.8 Adversarial Patch

Previously we assumed $\delta \in \mathbb{R}^{H \times W \times C}$ i. e. the perturbation δ can modify every pixel of the image as long as $\delta \in \mathcal{S}$. If we do not have the access to the camera and cannot e.g. put a sticker on it, doing this in the real world is difficult. Modifying all the objects in the scene including background ones to reproduce a digital adversarial example can be impossible even with the Expectation Over Transformation (Athalye et al., 2018b) method. One way to make implementing adversarial examples in the real world more feasible is localizing their impact so that they only need to affect a small region of the scene. It was implemented in the threat model called *adversarial patch* (Brown et al., 2017; Karmon et al., 2018). We assume that the

attacker can modify the pixels only within a confined region but the range of the modification is usually unbounded (as long as pixels values remain valid e. g. in range $[0,\ 1]$). An adversarial perturbation constructed this way can potentially be printed out and put directly in the physical environment e. g. by attaching a poster to some vertical surface.

Adversarial patch is not an imperceptible attack in the strict sense of this word because significant modification of the confined pixels can be noticed by a human being. However the size of the patch region is usually small enough (taking up to several percent of the image surface) so that the patch is not making it more difficult for a real person to analyze the image. Or it can even be perceived as a natural element of the scene such as QR code (Chindaudom et al., 2022). However, the neural networks were observed to be vulnerable to adversarial patch modifications (Karmon et al., 2018). Moreover, it was demonstrated to be the case when the adversarial patch was printed out and placed in the physical world (Brown et al., 2017; Lee and Kolter, 2019b). Recent work of Sato et al. (2021) has demonstrated further possible threat models such as dirty road patches that pose a significant threat for widely used Automatic Lane Centering systems in automated driving. They show how catastrophic car deviation from the lane can be achieved in under a second which is much shorter than reaction time of a driver. Such examples raise significant concerns for applying deep learning in automated or autonomous driving systems.

Black-box adversarial patches. Wei et al. (2022) propose to simultaneously optimize patch position and perturbation using reinforcement learning. They take advantage of the mutual correlation between the position and content of a patch. The results are provided for the face recognition as well as traffic sign recognition tasks. The attack is performed in the black-box setting (Section 2.2.5) which emphasizes its practicality. Lapid and Sipper (2023) produce realistic black-box adversarial patches using a Generative Adversarial Network (GAN) (Goodfellow et al., 2014a) and evaluate their efficiency both digitally and physically. Jiang et al. (2023) propose decision-based black-box adversarial patches for the video recognition task.

Adversarial patches for various computer vision tasks. Originally, Brown et al. (2017) suggested an optimization procedure to produce highly robust and practical adversarial patches for attacking classification models on real world objects. These patches were *universal* in the sense that adding the same patch to different inputs resulted in shifting the classification output of the deep learning model. It was attributed to the fact that the patch, although looking unnaturally to a human observer, managed to fool the neural network into believing that it is an object of a target class e. g. a toaster. Croce et al. (2022) proposed Sparse-RS framework that used a random search procedure similar to the one in Square Attack (Andriushchenko et al., 2020)

to produce black-box adversarial attacks for the threat models such as ℓ_0, adversarial patches and adversarial frames.

Patch attacks were also proposed for object detection systems (Pavlitskaya et al., 2022; Hartnett et al., 2022). Lovisotto et al. (2022) observed that certain types of self-attention (Vaswani et al., 2017) used in the recently emerged Vision Transformer (Dosovitskiy et al., 2021) architectures can be particularly susceptible to adversarial patches. Including a physically printed patch in the image has lead to a failure to detect objects (Lee and Kolter, 2019b) or people (Thys et al., 2019). Nesti et al. (2022) studied the effect of adversarial patch on semantic segmentation for autonomous driving. They observed that patches optimized with Expectation over Transformation (Athalye et al., 2018b) technique were able to significantly modify the segmentation model prediction in the real world. Kontár and Horváth (2022) studied patch attacks on semantic segmentation task as well. Ranjan et al. (2019b) have demonstrated the efficiency of patches applied to the optical flow estimation which is a crucial problem in autonomous driving. Adversarial patch attacks were also proposed for the task of monocular depth estimation (Cheng et al., 2022).

2.2.9 Conclusion

Adversarial examples constitute an intriguing, yet worrisome, aspect of deep learning. Apart from having significant theoretical value, studying adversarial examples has a clear practical goal of making deep learning applications robust, trustworthy and safe. This goal is emphasized by the existence of threat models that were demonstrated to be efficient in the physical world. Therefore, evaluating model robustness via adversarial attacks (including real world attacks) has to become an integral step of the future validation pipelines. But apart from finding out how vulnerable the models are to adversarial examples, another paramount task is to improve their robustness, make them less fragile. We discuss this aspect in the following section.

2.3 Improving Deep Learning Robustness

2.3.1 Empirical defences

There were numerous empirical approaches to make deep learning models robust to adversarial examples (Schott et al., 2018; Croce et al., 2020a). Dziugaite et al. (2016) study how JPG compressing affects the effectiveness of adversarial examples. Mosbach et al. (2018) propose adversarial logit pairing method to counteract gradient-based adversarial attacks. Up to now the most promising direction for obtaining robust models is *adversarial training* (AT) (Madry

et al., 2018b). It is a data augmentation technique based on using adversarial perturbations during training. Thus, model training becomes a minmax problem where an internal attacker is trying to maximize the loss while en external model training objective is to minimize it. For the image-label pair distribution $\mathcal{D}$ and a set of allowed perturbations $\mathcal{S}$ the AT problem formulation becomes:

$$\min_{\theta} \mathop{\mathbb{E}}_{(x,\, y)\sim D} \max_{\delta\in\mathcal{S}} L(f,\ \theta,\ x+\delta,\ y) \tag{2.16}$$

Rice et al. (2020) demonstrate that AT is prone to overfitting thus early stopping is required to achieve better model robustness. AT depends on the internal attack optimization procedure for which PGD 2.11 is often used (Madry et al., 2018b). Xiong and Hsieh (2020) suggest to use learned optimizers to improve the AT performance and reduce the amount of manual design. Jia et al. (2022) enhance AT with learnable attack strategies for generating adversarial examples and consistently improve the resulting model robustness of different AT modifications.

There is a novel line of work dedicated to empirical defences against black-box attacks. Aithal and Li (2022) propose Boundary Defence designed specifically against black-box attacks 2.2.5. Zhang et al. (2022b) consider different methods to defend machine learning model with only black-box access. They introduce a novel scenario in which not the *attacker* but the defender lacks the model information and can only query it. Chen et al. (2022) consider score-based attacks and propose an "Adversarial Attack on Attacker" (AAA) approach that slightly disturbs the output logits to prevent the attacks from fooling the model.

Training models to be robust in presence of adversarial examples usually causes a drop in clean accuracy i. e. when no adversarial examples are present (Madry et al., 2018b). Zhang et al. (2019b) propose TRADES defence framework which allows to control the trade-off between clean and robust accuracy.

Keeping the model prediction unchanged in presence of adversarial examples is not the only way to combat them. Another line of work is dedicated to *detecting* adversarial examples. Metzen et al. (2017b) suggest to use a detector subnetwork specifically trained for solving the binary classification task of distinguishing between benign and adversarial images. However, Carlini and Wagner (2017a) demonstrate that numerous detection methods can be broken by applying specific loss functions. Tramer (2022) further demonstrate that the problems of defending against adversarial examples and detecting them are to some extent equivalent.

In general, a shared problem of different empirical defences against adversarial examples is that evaluating the efficiency of such defences in a proper way is a highly non-trivial problem (Carlini et al., 2019). Uesato et al. (2018) observe that using weak adversarial attacks for robustness

evaluation may lead to false sense of security when deploying seemingly robust models. For example, gradient-obfuscation defences were shown to be unreliable when gradient-free attacks (such as black-box attacks from Section 2.2.2) are used (Athalye et al., 2018a). Croce and Hein (2020b) demonstrate how numerous empirical defence methods can be ”broken” by applying a stronger evaluation procedure. Tramer et al. (2020) systematize the known weaknesses in existing broken defences and advocate for using adaptive attacks for future evaluation.

2.3.2 Certified defences

One way to overcome the hurdle of unreliable robustness evaluation is to use defences that allow obtaining a non-trivial *provable lower bound* on robust accuracy of a model. Such result guarantees that although the model performance may degrade in presence of adversarial examples, it will certainly not go below some threshold. This can be done with *incomplete verifiers* (Wong and Kolter, 2018; Gowal et al., 2018) that we have mentioned in Section 2.2.1. Hein and Andriushchenko (2017) propose the Cross-Lipschitz regularization functional and provide provable ℓ_2-robustness guarantees for networks with one hidden layer.

The general idea of many robustness evaluation methods with incomplete verification is to consider the set S of allowed perturbations of the image x in the input space and propagate it through the classifier f to its logit space. If the whole propagated set lies within the same class as $f(x)$, we certify that no adversarial perturbation from the set S can change the prediction. The problem with this approach is that neural networks f are highly non-linear which makes exact propagation of the set S prohibitively expensive in terms of computational time. Thus one usually considers a convex relaxation f^R of the network f (Ehlers, 2017) which makes the propagation faster but less precise: $f(S) \subset f^R(S)$. And if we cannot verify that $f^R(S)$ lies completely within one class, it does not mean that the same holds for $f(S)$. In other words, by using a relaxation we never miss the existence of an adversarial example, but if we detect its existence with our relaxed propagation, it may be a false alarm. The bound for standardly trained networks is usually loose. Thus in order to make our robust accuracy estimate tighter one usually has to train networks specifically so that the difference $f^R(S) \setminus f(S)$ is small.

Wong and Kolter (2018) formulate a ReLU (rectified linear unit) network propagation as a linear program and consider its dual to evaluate the lower bound of the maximum difference between the true class logit and all other logits. If it is positive, then there is provably no adversarial example in the given neighborhood of the image. They consider a convex relaxation of ReLU activation function (Ehlers, 2017) and formulate the dual computation as a neural network propagation. The method required to evaluate the lower and upper bounds on each pre-activation

value which made the certification complexity quadratic with respect to the number of neurons in the network. Wong et al. (2018) extend this duality framework to other activation functions and skip connections (He et al., 2016c) via Fenchel duality. They also scale it to larger networks (although not to the ImageNet networks). Gowal et al. (2018) formulate the networks propagation in terms of interval arithmetic and obtain the robustness certificate without using duality approach. They call their method *Interval Bound Propagation*. Mirman et al. (2018) apply abstract interpretations machinery to propagate the adversarial set through the network. All of the above methods require specific model training procedure and achieve provable guarantees at the cost of clean model accuracy. Besides, they are hard to scale to the ImageNet classification problem. Salman et al. (2019) discuss the *convex relaxation barrier* problem that limits the certification tightness of propagation-based certified defences.

Cohen et al. (2019) proposed *Randomized Smoothing* (RS) which made a breakthrough in certified defences against ℓ_p norm adversarial examples by obtaining the first meaningful robustness certification results on ImageNet against ℓ_2 bounded attack. The defence is using the majority vote over the predictions obtained after adding random gaussian noise to the original image. The certification results are not deterministic, but can be obtained for an arbitrarily big confidence threshold by increasing the number of random noise sampling accordingly. However, the method relies on the model's ability to classify well under gaussian noise which can require additional model training. Salman et al. (2020b) address this issue by proposing *Denoised Smoothing* method that prepends a denoiser to a classification model. Carlini et al. (2022) introduce diffusion models (Sohl-Dickstein et al., 2015a) into the RS pipeline. They consider one-step image denoising that allows to perform robustness certification with arbitrary off-the-shelf models and significantly improve the certification results achieving the new state-of-the-art. Hong et al. (2022) propose UniCR, a framework that extends RS to certifying against arbitrary ℓ_p-norm bounded attacks with $p \in \mathbb{R}^+$ and different types of the random noise.

2.3.3 Defending against adversarial patches

In Section 2.2.2, we have described the adversarial patch threat model. Its key characteristic is the potential to be implemented in the physical world and efficiently fool deployed deep learning models. Hence, defences against this threat model are particularly important. Hayes (2018) discuss the difficulties of defending against adversarial patch attacks and propose *digital watermarking* defence. They emphasize the need to consider a variety of potential attacks when deploying deep learning models. Naseer et al. (2019) observe that pixel values of the optimized adversarial patch have big variation, thus the adversarial effect can be mitigated by applying

local gradient smoothing.

However, Chiang et al. (2020) demonstrate that empirical patch defences can be bypassed by using adaptive attacks specifically tailored to breaking these defences. This result is similar to the one for defences against ℓ_p-norm bounded attacks (Uesato et al., 2018). Chiang et al. (2020) propose the first certified defence against patch attacks that allows obtaining a guaranteed lower bound on model robust accuracy. They adapt the Interval Bound Propagation method, which was proposed by Gowal et al. (2018) for ℓ_∞ attacks, to the patch attacks. This approach caused significant drop in clean accuracy of the model on CIFAR10 and was difficult to scale to the ImageNet classification.

Levine and Feizi (2020) propose a certification procedure similar to the RS (Cohen et al., 2019) which they call *Derandomized Smoothing* (DRS). Classification results were obtained by majority voting over the images modified in a way that allows controlling the effect of any possible patch attack within certain boundaries. However, instead of random noise used in RS, the DRS was *masking* parts of the images. By keeping just a small region of the image visible and masking every other pixel with a special NULL symbol, Levine and Feizi (2020) were able to guarantee that a square patch with the size not-exceeding some pre-defined threshold cannot affect more than a certain number Δ of masked images. They trained a network f that was classifying the images with pixel values belonging to the range $[0,\ 1] \cup \{\text{NULL}\}$. Since some regions of the image (such as solid background) are not informative and do not provide the necessary cues to infer the image class, f was allowed to return no class prediction or several class predictions at once. By taking the majority vote and defining the winner class, they were obtaining the prediction for the image. If the difference in votes between the winner and the runner-up class is more than $2 \cdot \Delta$, then it is guaranteed that a patch cannot switch the prediction no matter where it is located or how it is optimized. An important property of DRS and other certified defences is that even if an attacker has perfect knowledge of the defence mechanism, it is still not possible to fool the model on an image which is certified. Randomized cropping (Lin et al., 2021) was using a similar methodology as DRS but instead of using a deterministic set of masks, they used a set of randomly located visible regions. They improved the inference speed significantly, but the certificate itself was not deterministic anymore as it was for DRS.

Further work studied how different model architectures could enhance certification against patch attacks. In particular, BagNet (Brendel and Bethge, 2019) was a popular choice due to its small receptive field. This neural network architecture is processing an image as a set of independent patches and applies global averaging to predictions on them. This naturally allows to limit the

effect of an adversarial patch. Zhang et al. (2020b) were using BagNet with clipping to improve the speed and performance of DRS. Xiang et al. (2021) apply a ”detect-and-mask” filter to the logits of pretrained BagNets.

Another neural architecture that inherently works with patch structure are the Vision Transformers (ViT) (Dosovitskiy et al., 2021). Salman et al. (2021) study the application of DRS to the ViT and observe significant boost in clean and robust accuracy as well as inference time. However, tailoring a defence to a specific architecture has a disadvantage of leaving other popular neural architectures undefended. This issue was addressed by Xiang et al. (2022a) who proposed *PatchCleanser*. This method applies a two-step masking procedure. In the first step, they mask all the possible patch locations in the image thus guaranteeing that at least one masked image is not affected by a patch, no matter where the patch is located. The second step applies another masking to certifiably determine what is the clean prediction of the model on this image. The only requirement to the classification model is that it needs to be able to maintain the prediction on clean images when any two patch locations are masked. It is not a very strict requirement given that DRS, for example, required a model to be able to infer the prediction from only a single small visible region. Balasubramanian and Feizi (2022) perform in-depth study of the masking procedure for the CNN input images which is a crucial component of the line of work based on DRS. They also argue that the ViT are less susceptible to the masking procedure since one can simply drop the transformer patches instead of actually masking them.

Certified detection techniques were proposed for patch attacks as well. Such defences guarantee that prediction is not modified by the patch attack or returns an alert if an attack is detected. Keeping the ratio of false alerts on clean images as low as possible is a secondary objective. Minority Reports (McCoyd et al., 2020) was the first defence of this kind. It was sliding the mask over the possible patch location to find the inconsistency in prediction. PatchGuard++ (Xiang and Mittal, 2021b) was applying the sliding mask not in the image space but in the feature space obtained by a feature extractor with small receptive field. It achieved a significant speed up since feature extraction only needed to happen once in the whole procedure on a given image. Han et al. (2021) propose to identify superficial important neurons which allows pruning the network so that the predictions are made for fewer masked inputs. Another defence using ViT was PatchVeto (Huang and Li, 2021). The authors observe that propagation in ViT can be done for a reduced set of transformer patches which allows to achieve certified attack detection without any adjustments or special training.

Defences against patch attacks were studied for the object detection task. Saha et al. (2019) in-

vestigate the role of spatial context in the object detection algorithms which makes them vulnerable to the patch attacks. Metzen et al. (2021) propose *Meta Adversarial Training* that improves model robustness against universal adversarial patch in the traffic-light detection problem. Detector Guard (Xiang and Mittal, 2021a) uses provably robust object classifiers to certify object detection in presence of patch hiding attacks. These are the attacks on object detection task which attempt to prevent the detection of certain objects i. e. *hide* them. When such an attack is detected, Detector Guard can abstain from making a prediction. Object Seeker (Xiang et al., 2022b) improves the certification results against patch hiding attacks. Moreover, in this defence we can obtain a robustness guarantee against a patch of *any* shape or size. Previous methods usually required prior knowledge of the expected patch geometry. Their novel masking scheme based on splitting image into halves allows to use off-the-shelf object detectors instead of provable object classifiers that usually suffer from a drop in clean accuracy. Xiang et al. (2023) summarize the advancements in the provable robustness against patches in different computer vision tasks and outline the main open problems. These are the improving of the trade-off between clean performance, certified robustness and computational overhead and generalizing proposed approaches to end-to-end AI systems consisting of numerous modules such as sensing, perception and planning.

2.3.4 Conclusion

We have discussed existing methods to defend deep learning models against adversarial attacks. We have considered defences against ℓ_p norm bounded attacks as well as against adversarial patches. The problems of probably evaluating empirical defences were discussed and certified defences were demonstrated to be a way to overcome some of them. In the following section, we describe the advancements that we have made in this work in the fields of both evaluating and improving the robustness of deep learning models.

Chapter 3

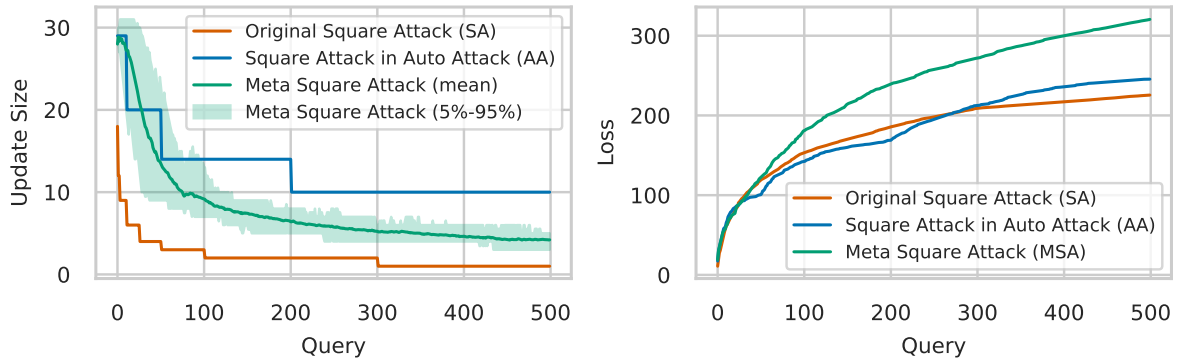
Meta Square Attack

This work was published by us as a conference paper at NeurIPS 2021 (Yatsura et al., 2021). We summarize the contributions made in this chapter as follows:

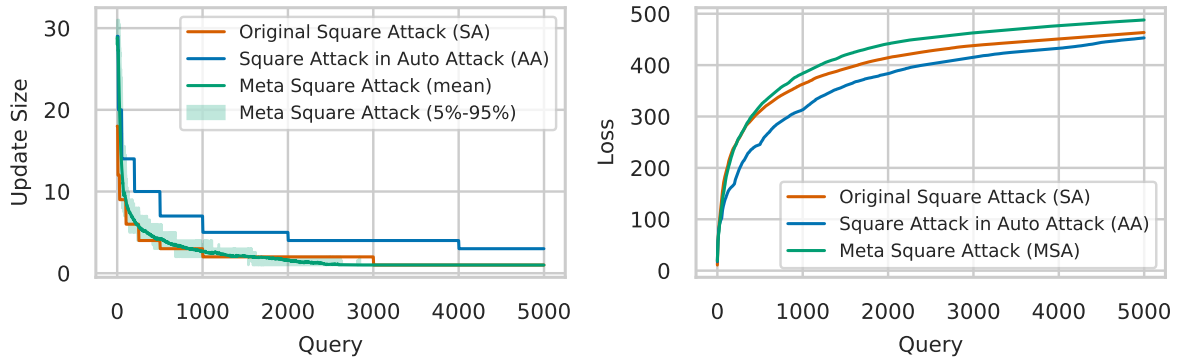
- We frame adversarial attack optimization as a meta-learning problem.
- We formalize gradient-based meta-learning for the Square Attack (Andriushchenko et al., 2020) and propose *Meta Square Attack* (MSA).
- We meta-train MSA on a CIFAR10 (Krizhevsky and Hinton, 2009) model with white-box access and show that MSA improves robust accuracy by up to 5.6% on a vast range of CIFAR10 models with black-box access with respect to the hand-designed search distributions proposed in previous work (Andriushchenko et al., 2020; Croce and Hein, 2020b) for the ℓ_∞ and ℓ_2 threat models (Figure 3.1).
- We show that Meta Square Attack generalizes well to different datasets and to the targeted attack setting. It achieves up to 20% better robust accuracy compared to the state-of-the art baseline (Andriushchenko et al., 2020) for attacking models on CIFAR100 and ImageNet.

Contributions

Please refer to the Table 3.1 for the contributions of the co-authors. Jan Hendrik Metzen and Matthias Hein have contributed to developing the method proposed in the Section 3.1. Jan Hendrik Metzen has helped with creating Figure 3.2 and implemented the code for the illustrative experiments in the Figures 3.6, 3.7.



(a) $T = 500$ queries



(b) $T = 5000$ queries

Figure 3.1: (Left) The schedules for SA (Andriushchenko et al., 2020) (which scales accordingly for different query budgets) and AA (Croce and Hein, 2020b) compared to Meta Square Attack (MSA) proposed in this work. MSA adapts the update size during the course of the attack for each image. We illustrate the mean and the percentile range for a given set of attacked images. (Right) The maximization of the loss for the model of Ding et al. (2020) on 100 CIFAR10 images. MSA outperforms SA and AA significantly in terms of the achieved loss.

Table 3.1: Contributions of the co-authors to the content of this chapter

Author	Author position	Scientific ideas %	Data generation %	Analysis & interpretation %	Paper writing %
Maksym Yatsura	1	80	90	90	85
Jan Hendrik Metzen	2	10	10	5	10
Matthias Hein	3	10	0	5	5
Paper title:		Meta-Learning the Search Distribution of Black-Box Random Search Based Adversarial Attacks			
Status in publication process:		Accepted at the Neural Information Processing Systems 2021			

3.1 Meta-learning adversarial attacks

In the following, we introduce the formulation of the optimization problem for an adversarial attack, rephrase it in the from of a meta-learning problem, and introduce our method for learning the search distribution of a specific random search based black-box adversarial attack, namely Square Attack (Andriushchenko et al., 2020). We denote this method as Meta Square Attack (MSA).

3.1.1 Adversarial Robustness Evaluation

Let K be the number of classes in a classification problem. Recall from 2.2 that we denoted

$$\Delta^{K-1} = \Big\{(p_0, ..., p_{K-1}) \in \mathbb{R}^K \Big| \sum_{i=0}^{K-1} p_i = 1, \text{ and } p_i \geq 0 \text{ for } i = 0, ..., K-1\Big\}, \tag{3.1}$$

to be the set of probability distributions over the K discrete possible outcomes. Let $f : [0,1]^d \to \Delta^{K-1}$ be a classifier which maps a d-dimensional input x to a probability vector. For a label $y \in \{0, ..., K-1\}$, a loss function $l : \Delta^{K-1} \times \{0, ..., K-1\} \to \mathbb{R}$, a perturbation set S, and an operator a, we define the robustness evaluation problem as:

$$V(f, x, y) = \max_{\delta \in S} l\big(f(a(x, \delta)), y\big) \tag{3.2}$$

For $S = \{\delta \mid ||\delta||_p \leq \epsilon\}$ and $a(x, \delta) = \Pi_{[0,1]^d}(x + \delta)$ (where $\Pi_{[0,1]^d}$ is the projection onto $[0,1]^d$), one obtains the standard ℓ_p ball threat model for images. Assuming that the loss function l and operator a are fixed, we denote $L(f, x, y, \delta) := l(f(a(x, \delta)), y)$ as a functional that one needs to maximize in robustness evaluation.

Since exact maximization of Equation 3.2 is intractable in the general case Katz et al. (2017), we consider a (potentially non-deterministic) procedure $\mathcal{A}_\omega(L, f, x, y)$ called adversarial attack. This attack $\mathcal{A}_\omega$ is parametrized by hyperparameters ω and designed with the intention that $\delta^\omega \sim \mathcal{A}_\omega(L, f, x, y)$ with $\delta^\omega \in S$ becomes an approximate solution (tight lower bound) of $V(f, x, y)$, that is $V(f, x, y) - L(f, x, y, \delta^\omega)$ should become small (in expectation). Optimizing the hyperparameters of the attack via

$$\max_{\omega} \mathop{E}_{\delta^\omega \sim \mathcal{A}_\omega(L, f, x, y)} L(f, x, y, \delta^\omega) \tag{3.3}$$

can allow a tighter lower bound of $V(f, x, y)$. Unfortunately, this maximization is still intractable typically, for instance when f is a black-box (no gradient information is available), the number of queries to f per data (x, y) is limited, or $\mathcal{A}_\omega$ has high variance.

3.1.2 Black-box Adversarial Attack Optimization as a Meta-learning Problem

We now frame optimization of the adversarial attack in the query-restricted black-box setting as a *meta-learning* problem. We follow the taxonomy proposed in the survey on meta-learning by Hospedales et al. (2021).

First, we formulate our **meta-objective** (the specification of the goal of meta-learning): we assume data to be governed by a distribution $(x, y) \sim \mathcal{D}$ and classifiers defined on this data, which need to be evaluated, by a distribution $f \sim \mathcal{F}$. Our meta-objective is to find parameters ω^* of the attack $\mathcal{A}_\omega$ that maximize the lower bound $L(f, x, y, \delta^\omega)$ of $V(f, x, y)$ in expectation across models $f \sim \mathcal{F}$, data $(x, y) \sim \mathcal{D}$, and the stochastic attack $\delta^\omega \sim \mathcal{A}_\omega$:

$$\omega^* = \underset{\omega}{argmax} \underset{f\sim\mathcal{F}}{E} \underset{(x,y)\sim\mathcal{D}}{E} \underset{\delta^\omega\sim\mathcal{A}_\omega(L,f,x,y)}{E} L(f, x, y, \delta^\omega) \tag{3.4}$$

Here, the expensive optimization of ω^* is amortized across models and data. More specifically, we assume finite sets of data $D = \{(x_i, y_i)_{i=1}^N \mid (x_i, y_i) \sim \mathcal{D}\}$ and classifiers $F = \{f_j \mid f_j \sim \mathcal{F}\}$ are available. Moreover, we assume the $f_j \in \mathcal{F}$ allow white-box access and quasi-unrestricted number of queries. The sets D and F can be used during *meta-training* of the attack. That is, the objective during meta-training is finding the parameter vector ω^*

$$\omega^* = \arg\max_{\omega} R(F, D, \omega) \tag{3.5}$$

that maximizes the sum of losses for each point $(x, y) \in D$ and each classifier $f \in F$

$$R(F, D, \omega) = \sum_{f\in F} \sum_{(x,y)\in D} L(f, x, y, \delta^\omega), \quad \delta^\omega \sim \mathcal{A}_\omega(L, f, x, y) \tag{3.6}$$

However, the ultimate goal of meta-learning is to apply $\mathcal{A}_{\omega^*}$ during *meta-testing* to unseen $(x, y) \sim \mathcal{D}$ and unseen $f \sim \mathcal{F}$ that allow only black-box and query-limited access, and maximize

$$\mathbb{E}_{\delta^\omega\sim\mathcal{A}_\omega(L,f,x,y)} L(f, x, y, \delta^\omega) \tag{3.7}$$

that is: the attack needs to generalize well across models and data.

Next, we define the **meta-representation**, that is how the adversarial attack $\mathcal{A}_\omega$ is designed and parametrized such that generalization across $f \sim \mathcal{F}$ is effective. In this work we focus on random search based adversarial attacks for black-box robustness evaluation since they have achieved strong results in prior work and are amenable to meta-learning. Let $\mathcal{A}_\omega$ be a

random search based attack with a query budget limited by T. Then an adversarial perturbation $\delta^\omega = \delta^T \sim \mathcal{A}_\omega(L, f, x, y)$ is obtained using the following iterative procedure:

$$\delta^0 \sim \mathcal{D}^0; \quad \delta^{t+1} = \underset{\delta \in \{\delta^t, \Pi_\mathcal{S}(\delta^t + \xi^{t+1})\}}{\arg\max} L(f, x, y, \delta); \quad \xi^{t+1} \sim \mathcal{D}_\omega(t, \delta^0, \xi^0, \ldots, \delta^t, \xi^t), \tag{3.8}$$

where $\Pi_\mathcal{S}$ corresponds to the projection onto the perturbation set $\mathcal{S}$.

That is, we assume a fixed distribution $\mathcal{D}^0$ for initializing the perturbation δ^0 but a meta-learnable $\mathcal{D}_\omega$ for the update proposals ξ^{t+1}. Importantly, $\mathcal{D}_\omega$ depends on the entire attack trajectory up to step t. Since this trajectory contains implicitly information on the classifier f when applied to data (x, y), our meta-learned random search attack can adapt to the classifier and data at hand. We provide more details on $\mathcal{D}_\omega$ for the specific case of Square attack (Andriushchenko et al., 2020) in Section 3.1.3.

The **meta-optimizer** (how we optimize the meta-objective) in our case assumes that both the loss function l and $\mathcal{A}_\omega$ are (stochastic) differentiable with respect to ω or we can find differentiable relaxations (as we will discuss in Section 3.1.3). Thus the meta-parameters ω can be optimized using stochastic gradient descent on mini-batches $B \subseteq D$ based on the (stochastic) gradient

$$g = \nabla_\omega R(F, D, \omega) = \sum_{f_j \in F} \sum_{(x_i, y_i) \in B \subseteq D} \nabla_\omega L(f_j, x_i, y_i, \delta_{i,j}), \tag{3.9}$$

where $\delta_{i,j} \sim \mathcal{A}_\omega(L, f_j, x_i, y_i)$. However, due to the stochasticity of $\mathcal{A}_\omega$ induced by $\mathcal{D}^0$ and $\mathcal{D}_\omega$, the discrete arg max in the update step of $\mathcal{A}_\omega$ (3.8), and the length T of the unrolled optimization (often in the order of hundreds to thousands queries), g would have very high variance and typically one would also face issues with vanishing or exploding gradients. To address this, we propose using a greedy alternative instead:

$$g = \frac{1}{T} \sum_{f_i} \sum_{(x_j, y_j)} \sum_{t=1}^{T-1} \nabla_\omega L(f_i, x_j, y_j, \Pi_\mathcal{S}(\delta^t + \xi^{t+1})). \tag{3.10}$$

Importantly, even though δ^t depends on ω for $t > 0$, we do not propagate gradients with respect to ω through δ^t, that is we set $\nabla_\omega \delta^t := 0$. By this, the gradient corresponds to optimizing $\mathcal{D}_\omega$ in a myopic way, such that proposals $\xi^{t+1} \sim \mathcal{D}_\omega(t, \delta^0, \xi^0, \ldots, \delta^t, \xi^t)$ are trained to maximally increase the immediate loss in step $t + 1$. While this introduces a bias of acting myopic and greedy, it works reasonably well in practice.

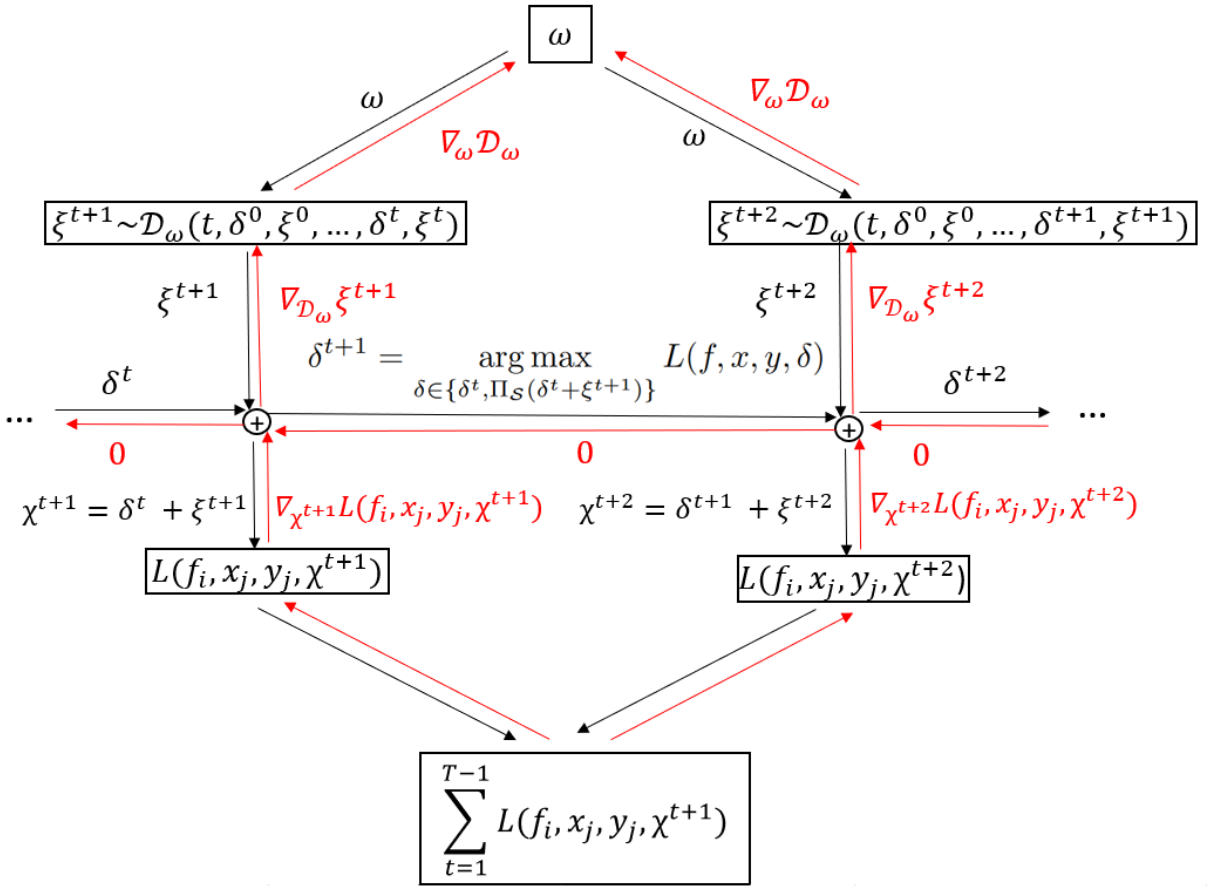


Figure 3.2: Schematic illustration of the general meta-learning procedure described in Section 3.1.2 for two subsequent steps of the random search attack (3.8). We describe forward (black) and backward (red) pass. $\nabla_x f$ denotes gradient for scalar-valued functions f and Jacobian for vector-valued functions. Differential expressions with search distribution $\mathcal{D}_\omega$ are informal and in need to be handled with reparametrization trick or other methods when applying the method to particular attacks (for examples see Section 3.1.3). Please note that the gradient with respect to ξ^t is set to 0 for all t (see Section 3.1.2).

By the chain rule we can now rewrite

$$\nabla_\omega L(f_i, x_j, y_j, \delta^t + \xi^{t+1}) = \nabla_{\xi^{t+1}} L(f_i, x_j, y_j, \delta^t + \xi^{t+1}) \nabla_\omega \xi^{t+1} \tag{3.11}$$

We note that $\nabla_\omega \xi^{t+1}$ can be a very high variance estimate of $\nabla_\omega \mathcal{D}_\omega(t, \xi^0, \delta^0, \ldots, \xi^t, \delta^t)$ depending on the stochasticity of $\mathcal{D}_\omega$. However, as it is an unbiased estimate and we average over very many steps T and models and data, g is still a sufficiently good estimate in practice. See the schematic illustration of the meta-learning procedure in Figure 3.2.

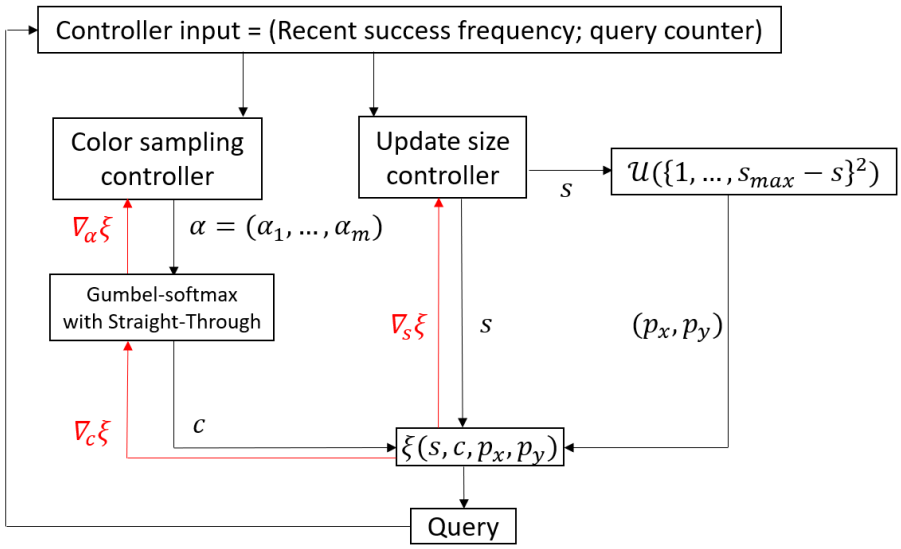


Figure 3.3: Illustration of Meta Square Attack described in Section 3.1.3. The search distribution $\mathcal{D}_{(s,c)}$ depends on parameters s and α that are provided by the update size controller $\pi^c_{\omega_c}$ and the color controller $\pi^s_{\omega_s}$, respectively. Square positions (p_x, p_y) are sampled from the uniform distribution $\mathcal{U}(\{1, ..., s_{max} - s\}^2)$

.

3.1.3 Meta Square Attack

In this section, we demonstrate how the proposed meta-learning approach can be applied to Square Attack (SA) (Andriushchenko et al., 2020) with ℓ_∞ threat model. We denote the resulting meta-learned attack as *Meta Square Attack* (MSA). We keep $\mathcal{D}^0$ as the stripe initialization from SA and focus on meta-learning $\mathcal{D}_\omega$ as it governs all but the first step. As discussed in Section 2.2.6, sampling $\delta^{t+1} \sim \mathcal{D}(t)$ in SA proceeds by computing a square size (its width in pixels) $s_t = \pi^s(t) \in \{1, \ldots, s_{max}\}$ and sampling a position $(p_x, p_y) \sim \pi^p(s) \in \{1, \ldots, s_{max} - s\}^2$ and a color $c \sim \pi^c \in \{c_1, \ldots c_m\}$. In SA, π^s is a heuristic schedule that depends on t (and differs in prior work (Andriushchenko et al., 2020; Croce and Hein, 2020b)) and both π^p and π^c are uniform distributions. The additive update δ^{t+1} is then chosen to be zero everywhere except for a square of size s at position p with color c. Possible colors c_i correspond to the eight corners of the RGB hypercube with ℓ_∞ norm ϵ. That is $c_i \in (\pm\epsilon, \pm\epsilon, \pm\epsilon)$, while s_{max} is chosen maximally with the constraint that all sampled squares must not exceed the image dimensions. We keep π^p as uniform distribution, but meta-learn the controllers $\pi^s_{\omega_s}$ and $\pi^c_{\omega_c}$ with parameters $\omega = (\omega_s, \omega_c)$. See a schematic illustrations of the Meta Square Attack training procedure in Figure 3.3

Update size controller $\pi^s_{\omega_s}$. We design $\pi^s_{\omega_s}$ as a multi-layer perceptron (MLP) with parameters

ω_s. The MLP outputs a scalar value $s' \in \mathbb{R}$ and we map this value to the actual update size via

$$s = \sigma(s') \cdot (s_{max} - 1) + 1 \text{ with } \sigma(x) = 1/(1 + e^{-x}) \tag{3.12}$$

This form allows s to be in the desired range $[1,\ s_{max}]$. During meta-testing, we round the continuous s to a discrete value $\lfloor s \rceil \in \{1, \ldots, s_{max}\}$. As this would block gradient flow during meta-training, we relax the square sampling in SA such that it supports continuous update sizes (see Section 3.1.4). Importantly, the relaxation is only conducted during meta-training and not in the final evaluation during meta-testing.

We provide two scalar inputs to the MLP $\pi^s_{\omega_s}$:

(a) the current query t encoded as $\log_2(\frac{t}{T} + 1)$ where T is the maximal number of queries. It ensures that the input stays in the range $[0, 1]$ for $t \leq T$. We use $T = 5000$.

(b) Let

$$r^{t+1} = H\left(L(f, x, y, \Pi_{\mathcal{S}}(\delta^t + \xi^{t+1})) - L(f, x, y, \delta^t)\right) \in \{0, 1\} \tag{3.13}$$

be an indicator of whether adding δ^{t+1} at time $t+1$ improved the loss (for H being the Heaviside step-function). If the loss value L has increased, we get 1, otherwise. The MLP $\pi^s_{\omega_s}$ gets the value $R^t = \gamma R^{t-1} + (1 - \gamma) r^t / r^0$ with $R^0 = 1$ as second input at time t. Here γ is a decay term that controls how quickly past experience is "forgotten" and $r^0 = 0.25$ is a constant whose purpose is to ensure that the MLP's second input has a similar scale as the first (namely in $[0, 1]$). Intuitively, (a) allows to schedule update sizes based on time step of the attack (this information is also used in SA itself) while (b) allows adapting the update size based on the recent success frequency R^t of the proposals (for instance, reducing the update size if few proposals were successful recently). Thus (b) allows the schedule to adapt to the classifier f and data (x, y) at hand.

Color controller $\pi^c_{\omega_c}$. We design the color controller $\pi^c_{\omega_c}$ as a categorical distribution

$$c^t \sim \text{Cat}(\alpha^t_1, \ldots, \alpha^t_m) \tag{3.14}$$

where each $\alpha^t_i \in \mathbb{R}$ is predicted by an MLP with weights ω_c. The m MLPs for the $\{\alpha^t_i\}^m_{i=1}$ share the weights ω_c but differ in their inputs. In order to differentiate through the categorical distribution at the meta-training time, we reparametrize the categorical distribution with the Gumbel-softmax and draw discrete (hard) samples in the forward pass but treat them as soft samples in the backward pass (Jang et al., 2017; Maddison et al., 2017). Additionally, we ensure that

every color c_i is sampled at least with probability p^c_{min} by assigning

$$P(c_i) := p^c_{min} + (1 - mp^c_{min})P_{\mathsf{Cat}(\alpha^t_1,\dots,\alpha^t_m)}(c_i) \tag{3.15}$$

This ensures continuous exploration of all m colors.

The m MLPs get two inputs: (a) the current query t encoded as $\log_2(\frac{t}{T} + 1)$ (same encoding as for the step size controller $\pi^s_{\omega_s}$ and also same for all m MLPs). (b) Information regarding the recent success frequency of proposals δ^t based on squares of the respective colors ($c^t = c_i$):

$$R^t_i = \begin{cases} \gamma R^{t-1}_i + (1-\gamma)r^t/r^0 & \text{if } c^t = c_i \\ R^{t-1}_i & \text{otherwise} \end{cases} \tag{3.16}$$

This second input allows the controller to learn, e.g., to sample those colors more often that resulted in higher success frequency recently.

3.1.4 Square relaxation

In Section 3.1.3, we formalize update size and color controllers that we learn for Meta Square Attack. Here we provide additional details on how we avoid blocking of gradient flow in our optimization scheme using relaxed square sampling.

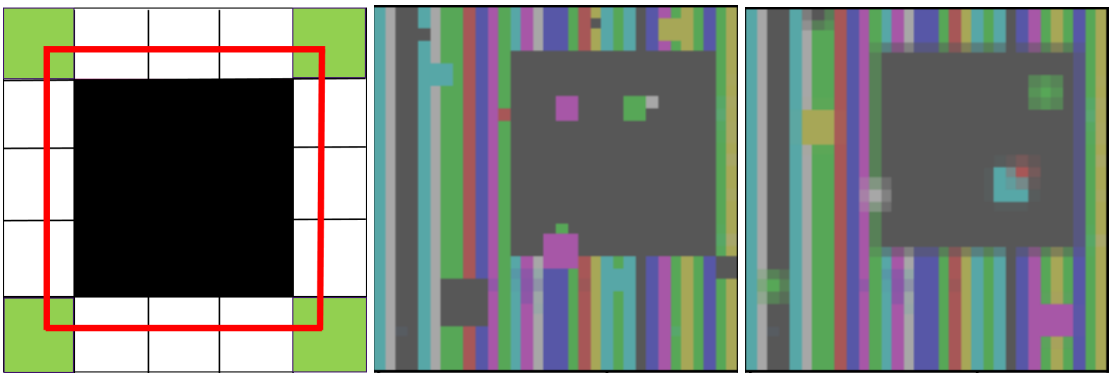
$$g = \frac{1}{T} \sum_{f_i} \sum_{(x_j,y_j)} \sum_{t=1}^{T-1} \nabla_\omega L(f_i, x_j, y_j, \delta^t + \xi^{t+1}), \tag{3.17}$$

for simplicity assuming that projection operator $\Pi_{\mathcal{S}}$ in Equation (3.10) is incorporated into L. Since by the chain rule we rewrite

$$\nabla_\omega L(f_i, x_j, y_j, \delta^t + \xi^{t+1}) = \nabla_{\xi^{t+1}} L(f_i, x_j, y_j, \delta^t + \xi^{t+1}) \nabla_\omega \xi^{t+1} \tag{3.18}$$

we need to compute the Jacobian $\nabla_\omega \xi^{t+1}$ of the update vector ξ^{t+1} with respect to the meta-parameters ω.

Recall that in Section 3.1.3 we denote $\omega = (\omega_s, \omega_c)$ and consider controllers $\pi^s_{\omega_s}$ and $\pi^c_{\omega_c}$ for the update size and color respectively. Since computing $\nabla_{\omega_c} \delta^{t+1}$ is done via Gumbel softmax (Jang et al., 2017; Maddison et al., 2017), here we concentrate on computing $\nabla_{\omega_s} \xi^{t+1}$. Since $\pi^s_{\omega_s}$ only controls update size, we assume its position and color to be fixed when computing the gradient.



(a) (b) (c)

Figure 3.4: (a) illustration of a square with non-integer size s (red), size $odd(s)$ (black), 4-neighborhood (white) and 8-neighborhood pixels that do not belong to 4-neighborhood (green), (b) standard square attack perturbation, (c) square attack perturbation with proposed square relaxation.

In SA (Andriushchenko et al., 2020) each update is parametrized by an integer square width from $\{1, ..., w\}$ where w is the image width. This parameter is obtained by rounding a real valued square size s obtained from the update size schedule to the closest integer in the feasible range. During meta-training we cannot round the output s of $\pi^s_{\omega_s}$ since rounding is a discrete non-differentiable operation and we get $\nabla_{\omega_s}\xi^{t+1} = 0$ almost everywhere. Therefore, we propose a differentiable relaxation for the square size (see Figure 3.4). We denote $odd(s)$ to be the largest odd integer which is not bigger than s and compute it as

$$odd(s) = 2 \cdot \lfloor \frac{s-1}{2} \rfloor + 1 \tag{3.19}$$

The inner part of the square with the width $odd(s)$ is filled with the sampled color c completely. The color of pixels in the 1-pixel boundary is interpolated between the background color c_0 and the new color c as: $k \cdot c + (1-k) \cdot c_0$. The coefficient k of the new color is equal to the fraction that the square of non-integer width s would occupy in the respective pixel. Therefore, for the 4-neighborhood the new color fraction is $k = \frac{s-\text{odd}(s)}{2}$ and for the pixel of the 8-neighborhood that do not belong to the 4-neighborhood $k = (\frac{s-\text{odd}(s)}{2})^2$.

3.1.5 Meta Square Attack for the ℓ_2 threat model

To meta-learn the update size controller for the ℓ_2 threat model we use the same procedure as discussed in Section 3.1.3. The only difference is the relaxation that we use to sample continous updates since the update geometry is different. See Section 3.1.4 for the ℓ_∞ case.

The sampling procedure of the ℓ_2 Square Attack is described in detail in the Algorithm 3 in Andriushchenko et al. (2020). On a high level the algoritm consists of 2 steps for the two regions of the image W_1 and W_2:

1. Take the mass from W_2
2. Add it to W_1

Let s be non-integer square size. $odd(s)$ – the largest odd integer number not exceeding s, $odd(s) = 2 \cdot \lfloor \frac{s-1}{2} \rfloor + 1$. The performed update is a linear interpolation between the squares of size $odd(s)$ and $odd(s) + 2$. We denote $frac(s) = \frac{s-odd(s)}{2} \in [0; 1)$ that will be an interpolation coefficient.

For the step 1 we consider the window W_2 of size $odd(s)$ and denote it's 1-pixel outer boundary as W_2^B. As in SA (Andriushchenko et al., 2020), we set the whole W_2 to 0 and add $||W_2||_2$ to the update budget. We also add $frac(s) \cdot ||W_2^B||_2$ to the budget, therefore taking $frac(s)$ part of the norm. We update the boundary as

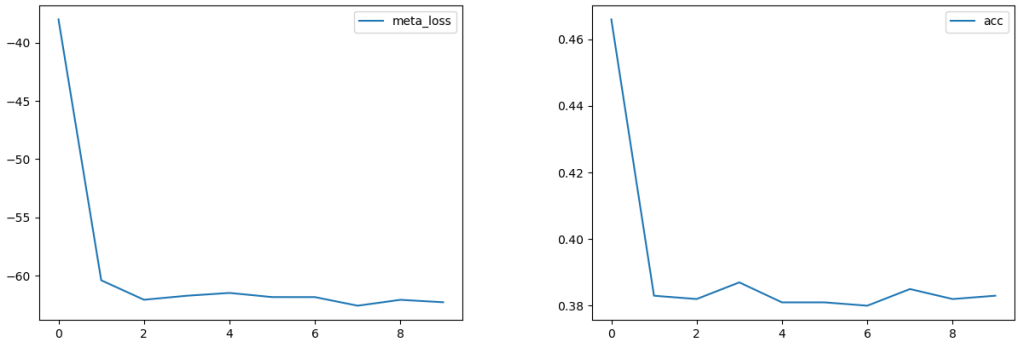
$$W_{2,new}^B := \sqrt{1 - frac(s)^2} \cdot W_2^B \tag{3.20}$$

$$||W_2^B||_2^2 = ||W_{2,new}^B||_2^2 + frac(s)^2 \cdot ||W_2^B||_2^2. \tag{3.21}$$

3.2 Experiments

We perform an empirical evaluation of the Meta Square Attack (MSA). First, we consider the data distribution $\mathcal{D}$ of CIFAR10 (Krizhevsky and Hinton, 2009) images and a classifier distribution $\mathcal{F}$ consisting of the classifiers robust with respect to the ℓ_∞-threat model. We use this setting for the meta-training as discussed in Section 3.2.1. We further consider how the controllers trained for these distributions generalize to working with the other data distributions of CIFAR100 and ImageNet and corresponding distributions of classifiers defined on this data. We also discuss the meta-training for the distribution of the classifiers robust with respect to the ℓ_2-threat model in the Section 3.1.5.

We compare the performance of MSA in 4 different query budget regimes to manually designed schedules for Square Attack proposed by Andriushchenko et al. (2020) (denoted by SA) and Croce and Hein (2020b) (denoted by AA): 500, 1000, 2500 and 5000 queries. The reason why we have chosen this evaluation mode instead of reporting accuracy and average number of queries for some single fixed budget is that the original SA approach proposes to scale the



(a) meta-loss

(b) robust accuracy

Figure 3.5: Meta-loss and robust accuracy on the training set during meta-training.

schedule to a given budget (Figure 1.1). Hence, it uses the knowledge of the attack budget. The schedule used for 500 queries is not a truncated version of the schedule used for 5000 queries as it is done in AA. Since adapting the schedule to different query budgets is a crucial factor for the manually designed schedules that we consider as baselines, we choose the evaluation regime that allows to take this factor into consideration. Section 3.2.2 summarizes the experimental results and Section 3.2.3 analyzes the behavior learned by the controllers.

3.2.1 Meta-Training and Controller Design

We meta-train the controller on a single robust model $f \sim \mathcal{F}$ with white-box access (the "source model"). Meta-training on more than one source model could improve generalization across models, but we found that even meta-training on a single model generalizes sufficiently well. The source model was designed such that the attackers could easily and cheaply acquire it themselves: the model has ResNet18 (He et al., 2016a) architecture and was trained on the CIFAR10 (Krizhevsky, 2009) training set using adversarial training (Madry et al., 2018a) with the advertorch (Ding et al., 2019) package. Adversarial training was done using the ℓ_∞-PGD attack with $\epsilon = 8/255$, fixed step size of 0.01, and 20 steps.

For both update size and color controllers, we use MLP architectures with 2 hidden layers, 10 neurons each, and ReLU activations. We purposefully did not finetune the MLP architecture. Meta-training was run on a training set D consisting of 1000 images from CIFAR10 test set (different from the ones used in evaluation of controllers in the next subsection) and Square Attack with a query budget of 1000 iterations. Therefore, controller behaviour on query regimes higher than 1000 are obtained by extrapolation of the behaviour learned for 1000 iterations. We use the default order of the CIFAR10 images (i. e., we do not shuffle). For the meta-training we

use images from 0 to 999 and for the evaluation we use images from 9000 to 9999 i. e. different subsets of the CIFAR10 test set. We do not use the CIFAR10 training set for the meta-training because it was used to train our source model.

Both controllers were trained simultaneously for 10 epochs using Adam optimizer with batch size 100 and cosine step size schedule (Loshchilov and Hutter, 2017b) with learning rate 0.03. The total loss improvement over the attack was used as the meta-loss that we optimized in the meta-training. Figure 3.5 demonstrates the minimization of $R_{MSA}(F, D, \omega)$ (see Equation 3.6 in Section 3.1.2) and corresponding behavior of the accuracy on the training set.

One can see that the proposed meta-loss $R_{MSA}(F, D, \omega)$ serves as a reasonable differentiable proxy for the robust accuracy. We observe that the loss reaches a close-to-minimal value already after two epochs. We always run the attack on all images for the full budget T, since removing images from the attacked batch would cause discontinuity in the meta-loss. All computations including meta-training and evaluation of the controllers were performed on a single Nvidia Tesla V100-32GB GPU.

3.2.2 Evaluation

In this section, we evaluate how the Meta Square Attack (MSA) obtained by training on a CIFAR10 model with white-box access discussed in the Section 3.2.1 performs for different models and datasets.

Table 3.2 illustrates that MSA transfers well to a broad range of 16 robust CIFAR10 models. Moreover, Table 3.3 reports aggregated results. We report mean, minimal, and maximal improvement across all the 16 models. We observe a consistent improvement for each considered query budget regime, which is especially pronounced for lower query regimes of 500 and 1000 queries. The improvements generalize to a budget of 5000 queries, which is five times higher than the one used during meta-training.

Table 3.2: **CIFAR10** (Krizhevsky, 2009): we compare the update size controller MSA_s to the schedules from the SA (Andriushchenko et al., 2020) and the AA (Croce and Hein, 2020b) in the ℓ_∞ threat model with $\epsilon = 8/255$ on 1000 CIFAR10 test images. We also compare the uniform color sampling denoted as "Uni" to our color controller MSA_c with 16 models from Robustbench. Averaged across at least 3 runs with different random seeds.

Model	Accuracy (%)		Square	Color	Query budget			
	Clean	Robust	size		500	1000	2500	5000
Wong et al. (2020)	83.34	43.21	SA	Uni	69.7±0.15	63.5±0.10	55.1±0.04	50.8±0.08
			AA	Uni	69.5±0.21	63.9±0.10	57.4±0.07	53.6±0.06
			MSA_s	Uni	**63.9±0.11**	59.8±0.10	54.0±0.16	51.1±0.08
			MSA_s	MSA_c	**63.9±0.12**	**59.1±0.09**	**53.0±0.16**	**49.8±0.08**
Ding et al. (2020)	84.36	41.44	SA	Uni	68.7±0.20	63.2±0.28	57.8±0.13	54.9±0.17
			AA	Uni	66.6±0.18	62.2±0.14	57.5±0.12	55.0±0.20
			MSA_s	Uni	62.4±0.15	59.4±0.09	56.1±0.10	54.6±0.06
			MSA_s	MSA_c	**62.2±0.14**	**59.1±0.16**	**55.9±0.15**	**54.1±0.15**
Engstrom et al. (2019)	87.03	49.25	SA	Uni	72.8±0.19	67.4±0.21	59.9±0.17	56.3±0.07
			AA	Uni	71.9±0.1	67.9±0.14	61.6±0.12	58.0±0.06
			MSA_s	Uni	67.9±0.12	64.2±0.15	58.9±0.05	56.4±0.12
			MSA_s	MSA_c	**67.8±0.12**	**63.4±0.18**	**58.2±0.06**	**55.9±0.04**
Gowal et al. (2021)	89.48	62.76	SA	Uni	80.6±0.09	76.7±0.06	70.8±0.14	67.5±0.07
			AA	Uni	80.0±0.17	76.8±0.11	72.2±0.10	69.2±0.08
			MSA_s	Uni	**76.9±0.05**	73.7±0.05	69.8±0.13	67.6±0.04
			MSA_s	MSA_c	**76.9±0.07**	**73.4±0.13**	**69.0±0.08**	**67.2±0.04**
Carmon et al. (2019)	89.69	59.53	SA	Uni	79.0±0.15	76.0±0.14	68.2±0.07	**6**5.4±0.09
			AA	Uni	78.0±0.10	74.5±0.11	69.6±0.04	67.1±0.05
			MSA_s	Uni	**74.4±0.09**	70.8±0.06	67.5±0.07	65.6±0.07
			MSA_s	MSA_c	74.6±0.10	**70.3±0.07**	**67.0±0.07**	**65.4±0.08**
Huang et al. (2020)	83.48	53.34	SA	Uni	72.3±0.1	66.6±0.16	60.5±0.09	57.3±0.08
			AA	Uni	70.6±0.10	66.5±0.07	61.2±0.10	58.5±0.11
			MSA_s	Uni	66.4±0.12	63.4±0.08	59.2±0.07	57.4±0.15
			MSA_s	MSA_c	**66.1±0.10**	**62.9±0.15**	**58.7±0.05**	**56.8±0.08**
Andriushchenko and Flammarion (2020)	79.84	43.93	SA	Uni	66.0±0.22	60.5±0.24	54.0±0.06	50.2±0.03
			AA	Uni	64.6±0.12	60.2±0.22	55.7±0.10	52.1±0.15
			MSA_s	Uni	60.4±0.09	57.0±0.07	52.5±0.19	50.0±0.06
			MSA_s	MSA_c	**60.1±0.07**	**56.8±0.15**	**51.9±0.15**	**49.4±0.22**
Zhang et al. (2019b)	84.92	53.08	SA	Uni	72.3±0.03	67.2±0.19	62.0±0.09	59.0±0.06
			AA	Uni	70.8±0.22	67.2±0.17	62.7±0.10	60.3±0.17
			MSA_s	Uni	67.5±0.06	64.2±0.18	60.8±0.07	59.0±0.07
			MSA_s	MSA_c	**66.8±0.09**	**63.9±0.07**	**60.4±0.06**	**58.7±0.13**

Model	Accuracy (%)		Square	Color	Query budget			
	Clean	Robust	size		500	1000	2500	5000
Hendrycks et al. (2019)	87.11	54.92	SA	Uni	75.3±0.30	69.8±0.19	64.2±0.15	60.8±0.00
			AA	Uni	74.7±0.17	70.5±0.26	64.7±0.12	62.8±0.15
			MSA_s	Uni	71.1±0.07	66.6±0.15	63.2±0.12	61.0±0.07
			MSA_s	MSA_c	**70.6±0.17**	**66.1±0.12**	**62.7±0.13**	**60.4±0.15**
Wang et al. (2020)	87.50	56.29	SA	Uni	77.7±0.12	72.2±0.03	65.8±0.15	62.2±0.03
			AA	Uni	76.7±0.06	72.8±0.12	67.6±0.20	64.0±0.17
			MSA_s	Uni	73.1±0.18	69.8±0.09	64.9±0.15	62.3±0.03
			MSA_s	MSA_c	**72.7±0.07**	**69.5±0.09**	**64.3±0.18**	**62.0±0.07**
Cui et al. (2021)	88.22	52.86	SA	Uni	75.5±0.22	69.6±0.09	62.9±0.20	59.2±0.15
			AA	Uni	74.2±0.13	70.2±0.07	64.8±0.07	61.1±0.23
			MSA_s	Uni	70.1±0.07	66.7±0.18	61.8±0.12	59.7±0.03
			MSA_s	MSA_c	**70.0±0.15**	**66.2±0.25**	**60.8±0.09**	**59.0±0.06**
Sitawarin et al. (2020)	86.84	50.72	SA	Uni	73.4±0.06	66.4±0.10	61.1±0.07	57.4±0.12
			AA	Uni	72.0±0.20	66.8±0.23	62.3±0.20	59.4±0.12
			MSA_s	Uni	**66.7±0.06**	63.6±0.03	60.3±0.17	57.5±0.03
			MSA_s	MSA_c	66.9±0.00	**63.1±0.09**	**59.3±0.12**	**57.0±0.00**
Wu et al. (2020)	85.36	56.17	SA	Uni	75.0±0.19	69.7±0.21	63.8±0.12	60.4±0.07
			AA	Uni	73.6±0.03	69.5±0.25	64.5±0.07	62.3±0.07
			MSA_s	Uni	69.6±0.09	66.1±0.20	63.1±0.17	60.7±0.07
			MSA_s	MSA_c	**69.4±0.23**	**65.7±0.12**	**62.6±0.12**	**60.3±0.03**
Zhang et al. (2021)	89.36	59.64	SA	Uni	79.6±0.27	74.6±0.03	66.9±0.07	64.0±0.07
			AA	Uni	78.4±0.06	75.3±0.03	68.9±0.06	65.6±0.03
			MSA_s	Uni	75.1±0.09	71.4±0.09	66.2±0.20	64.3±0.06
			MSA_s	MSA_c	**75.0±0.19**	**70.4±0.17**	**65.6±0.09**	**63.8±0.10**
Zhang et al. (2020a)	84.52	53.51	SA	Uni	73.4±0.03	67.5±0.09	61.5±0.12	**58.9±0.09**
			AA	Uni	72.3±0.06	67.7±0.00	62.3±0.06	60.4±0.06
			MSA_s	Uni	**67.4±0.09**	63.6±0.25	61.2±0.03	59.3±0.10
			MSA_s	MSA_c	67.6±0.06	**63.5±0.09**	**60.6±0.10**	59.0±0.10
Zhang et al. (2019a)	87.20	44.83	SA	Uni	73.1±0.00	66.2±0.26	56.5±0.15	52.5±0.12
			AA	Uni	71.8±0.23	66.5±0.07	59.2±0.09	54.7±0.12
			MSA_s	Uni	66.9±0.18	61.9±0.09	55.2±0.15	52.6±0.09
			MSA_s	MSA_c	**66.4±0.06**	**60.8±0.15**	**54.6±0.09**	**51.9±0.12**

As described in Section 3.1.3, the inputs to the MSA_s and MSA_c are not specific to the data distribution and only use the current attack iteration and improvement rate. Therefore, we also show that the adaptation principles learned on a CIFAR10 model transfer well to attacking models that not only have different architecture but also operate on significantly different data distributions: Table 3.4 demonstrates generalization of the learned controllers for attacking robust models for CIFAR100 (Krizhevsky and Hinton, 2009) which contains significantly more classifi-

Table 3.3: Improvement in l_∞-robust accuracy of our MSA with respect to the *best* of the previous Square Attack configurations, SA (Andriushchenko et al., 2020) and AA (Croce and Hein, 2020b), in the setting of Table 3.2. The results are accumulated across 16 robust CIFAR10 models from RobustBench (Croce et al., 2020a) (see Table 3.2 for full results).

Query budget	500			1000			2500			5000		
Improvement in	mean	min	max	mean	min	max	mean	min	max	mean	min	max
robust accuracy (%)	4.29	3.1	5.6	3.87	2.7	5.4	1.63	0.9	2.1	0.38	-0.1	1.0

cation categories. We also consider the transfer to ImageNet (Deng et al., 2009) dataset that has significantly higher input dimension and number of classes than CIFAR10. In Table 3.5, one can see that MSA significantly improves the results even in the high extrapolation regime of 5000 queries. We also observe considerable improvement of the robust accuracy estimate for the targeted attacks on undefended ImageNet models: ResNet-50 (He et al., 2016b), VGG-16-BN (Simonyan and Zisserman, 2015), Inception v3 (Szegedy et al., 2016) (Table 3.6).We consider ℓ_∞ threat model with $\epsilon = 0.05$. We compare our method with SA and set $p^0 = 0.05$ for the untargeted case and $p^0 = 0.01$ for the targeted case as suggested by Andriushchenko et al. (2020). For the targeted attacks robust accuracy is the fraction of the total number of images that was initially correctly classified by the model and not shifted to the target class during the attack. We provide clean accuracy of the models on a subset of 1000 ImageNet validation set images that we consider.

Table 3.4: **Transfer to CIFAR100:** MSA trained on a CIFAR10 model consistently outperforms SA (Andriushchenko et al., 2020) and AA (Croce and Hein, 2020b) on CIFAR100 (1000 images) in robust accuracy for the ℓ_∞-threat model with $\epsilon = 8/255$. Averaged across 3 runs with different random seeds.

Model	Accuracy (%)		Attack	Query budget			
	Clean	Robust		500	1000	2500	5000
Wu et al. (2020)	60.38	28.86	SA	43.3±0.17	38.7±0.09	33.9±0.28	32.3±0.20
			AA	41.3±0.20	37.6±0.00	35.0±0.06	32.7±0.35
			MSA	**37.8±0.07**	**35.5±0.12**	**33.1±0.03**	**32.2±0.03**
Cui et al. (2021)	70.25	27.16	SA	48.9±0.03	42.3±0.20	33.6±0.09	30.5±0.06
			AA	47.5±0.17	42.8±0.09	35.9±0.13	32.5±0.15
			MSA	**42.6±0.13**	**37.8±0.12**	**32.5±0.30**	**30.1±0.09**

Table 3.7 demonstrates the results for the ℓ_2 threat model for five ℓ_2 robust models from RobustBench (Croce et al., 2020a). We have chosen the models for which Croce and Hein (2020b)

Table 3.5: Results of attacking 1000 ImageNet validation set images with ℓ_∞ threat model and $\epsilon = 4/255$ as in Croce and Hein (2020b). For the SA update size schedule, we use the parameter $p^0 = 0.05$ as suggested in Andriushchenko et al. (2020). AA and Uni are defined as in Table 3.2. MSA$_s$ and MSA$_c$ are meta-trained on CIFAR10 (see Section 3.2.1 for details). We report mean and standard error of robust accuracy for different queries budgets across 3 runs with different random seeds.

Model	Accuracy (%)		Square	Color	Query budget			
	Clean	Robust	size		500	1000	2500	5000
resnet18 Salman et al. (2020a)	52.5	25.0	SA	Uni	50.6±1.43	48.1±1.18	43.9±1.00	40.3±1.21
			AA	Uni	45.2±1.09	43.5±0.86	41.0±1.07	39.0±1.21
			MSA$_s$	Uni	43.4±0.94	**41.7±1.13**	39.5±1.07	38.3±1.33
			MSA$_s$	MSA$_c$	**43.3±1.00**	**41.7±0.94**	**39.1±1.23**	**37.8±1.36**
resnet50 Engstrom et al. (2019)	63.4	27.6	SA	Uni	59.8±0.64	57.2±0.79	52.9±1.11	48.6±1.31
			AA	Uni	54.6±0.99	52.8±1.09	50.3±1.43	48.1±1.18
			MSA$_s$	Uni	52.6±1.07	51.2±1.40	48.3±1.22	**45.8±1.26**
			MSA$_s$	MSA$_c$	**52.5±1.23**	**50.8±1.47**	**48.0±1.15**	**45.8±1.35**

provide their evaluation of the Square Attack. They evaluate on the whole CIFAR10 test set and we evaluate on a subset of 1000 test images. Therefore their estimate is not identical to our entry AA+Uni. But we still provide it as Sq AA (Croce and Hein, 2020b) in the Table 3.7 for additional reference. We observe consistent improvement of about 3% robust accuracy for all of the considered query budgets. The magnitude of improvement for different datasets and threat models depend on two factors: how well the adaptation mechanism learned by our controllers generalizes in the given setting and how suitable the hand-designed search distributions of the baselines (Andriushchenko et al., 2020; Croce and Hein, 2020b) are for each particular problem. The consistency of the improvement indicates that the adaptive search distribution makes Meta Square Attack more efficient in the majority of the settings.

3.2.3 Analysis of Learned Controllers

As the learned controllers are black-boxes (implemented by MLPs), it may be non-trivial to understand their realized strategy. We present some analysis of the controllers' internal strategy based on their empirical behavior.

Figure 3.6 illustrates the behaviour of the update size controller MSA$_s$. It shows how the controller chooses the update size over time in an artificial scenario where the success probability $P(r^t = 1)$ is modeled to be constant. As implemented also by the heuristic schedules SA and AA, update size decays over time. However, the decay pattern depends heavily on $P(r^t = 1)$, with slower decay for larger $P(r^t = 1)$. This property is not implemented by heuristic sched-

Table 3.6: MSA trained on a CIFAR10 model attacking the **undefended ImageNet models** (ResNet-50 (He et al., 2016b), VGG-16-BN (Simonyan and Zisserman, 2015), Inception v3 (Szegedy et al., 2016)) in the ℓ_∞ threat model with $\epsilon = 0.05$.

Model	Clean acc. (%)	Attack	Untargeted				Targeted			
			500	1000	2500	5000	500	1000	2500	5000
ResNet-50	77.3	SA	8.8	5.1	0.2	**0.0**	76.9	75.1	62.5	34.4
		MSA	**2.9**	**0.8**	**0.0**	**0.0**	**67.1**	**52.0**	**27.8**	**12.1**
VGG-16-BN	75.0	SA	2.8	0.9	**0.0**	**0.0**	74.5	72.2	51.5	17.4
		MSA	**1.8**	**0.2**	**0.0**	**0.0**	**62.5**	**45.2**	**16.5**	**3.5**
Inception v3	77.6	SA	16.6	6.1	**2.3**	**1.0**	77.5	76.4	70.9	59.8
		MSA	**10.2**	**5.1**	2.6	1.3	**74.4**	**70.6**	**60.1**	**49.5**

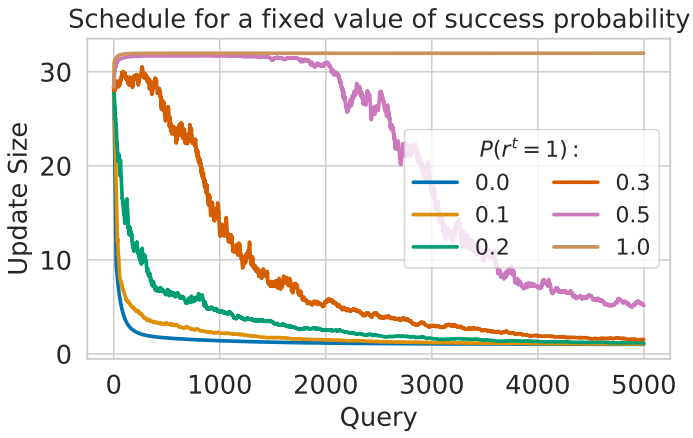


Figure 3.6: Square-size schedule obtained from MSA_s over time for fixed values of success probability $P(r^t = 1)$ (averaged over 25 runs).

ules, but makes sense intuitively: high $P(r^t = 1)$ corresponds to a situation where the current perturbation ξ^t can be improved relatively easily by coarse-grained changes implemented by the current update sizes; in this case it makes sense to first get the coarse-grained structure "right", before proceeding to fine-grained details that can be captured by small squares.

Figure 3.7 illustrates the empirical behavior of the color controller MSA_c when attacking the model by Ding et al. (Ding et al., 2020). Shown are histograms over 500 images for the frequency of specific colors being sampled up to the respective iteration. Prior work like SA and AA maintained a uniform distribution of colors. However, the learned controller shows a clear preference for sampling black and white more often than uniform ($p \approx 0.18$ vs. $p = 0.125$ for uniform), blue and yellow approximately with $p = 0.125$, and the other colors less often than uniform. Since the color controller depends on the success rates R^t of colors, this behavior

Table 3.7: MSA_s^2 is the update size controller trained for the ℓ_2 attack on a CIFAR10 model described in the Section 3.2.1. MSA_s^∞ denotes the update size controller meta-trained for the ℓ_∞ Square Attack on CIFAR10. The color controller MSA_c is the same as for the ℓ_∞ case. We compare to the ℓ_2 versions of SA (Andriushchenko et al., 2020) and AA (Croce and Hein, 2020b) with $\epsilon = 0.5$ on 1000 CIFAR10 images.

Model	Accuracy (%)			Square	Color	Query budget			
	Clean	Robust	Sq AA	size		500	1000	2500	5000
Ding et al. (2020)	88.02	66.09	76.99	SA	Uni	85.5±0.06	83.9±0.08	81.1±0.00	78.7±0.06
				AA	Uni	82.8±0.09	81.4±0.06	79.2±0.00	77.7±0.15
				MSA_s^∞	Uni	82.6±0.09	81.7±0.12	79.8±0.12	77.9±0.07
				MSA_s^∞	MSA_c	82.5±0.22	81.5±0.03	78.5±0.06	76.9±0.17
				AA	MSA_c	82.6±0.03	81.1±0.17	78.2±0.10	76.5±0.09
				MSA_s^2	MSA_c	**82.3±0.03**	**80.9±0.07**	**77.4±0.09**	**75.8±0.19**
Rice et al. (2020)	88.67	67.68	79.01	SA	Uni	86.3±0.07	84.7±0.10	81.4±0.15	79.7±0.07
				AA	Uni	83.7±0.12	81.4±0.09	79.9±0.09	78.6±0.18
				MSA_s^∞	Uni	83.2±0.12	81.8±0.07	80.1±0.07	79.1±0.09
				MSA_s^∞	MSA_c	83.0±0.15	81.2±0.06	79.6±0.03	78.3±0.03
				AA	MSA_c	83.4±0.12	81.2±0.10	79.3±0.09	78.0±0.06
				MSA_s^2	MSA_c	**82.6±0.09**	**81.0±0.07**	**78.7±0.03**	**76.9±0.25**
Augustin et al. (2020)	91.08	72.91	83.10	SA	Uni	89.0	88.4	86.9	84.2
				AA	Uni	87.8±0.03	86.8±0.09	84.8±0.17	83.3±0.17
				MSA_s^∞	Uni	87.7±0.06	87.0±0.09	85.2±0.03	83.4±0.10
				MSA_s^∞	MSA_c	87.4±0.15	86.5±0.12	84.1±0.20	82.8±0.13
				AA	MSA_c	87.7±0.12	86.6±0.09	83.9±0.13	82.7±0.09
				MSA_s^2	MSA_c	**87.5±0.12**	**86.3±0.06**	**83.4±0.03**	**81.8±0.07**
Engstrom et al. (2019)	90.83	69.24	80.92	SA	Uni	87.3	86.1	84.0	80.8
				AA	Uni	85.3±0.06	83.7±0.15	81.5±0.24	79.5±0.18
				MSA_s^∞	Uni	85.2±0.12	84.2±0.17	82.0±0.09	79.9±0.07
				MSA_s^∞	MSA_c	85.1±0.07	83.7±0.03	80.6±0.06	78.8±0.09
				AA	MSA_c	85.2±0.07	83.5±0.07	80.6±0.06	78.5±0.15
				MSA_s^2	MSA_c	**84.7±0.09**	**83.1±0.06**	**79.7±0.13**	**77.4±0.00**
Rony et al. (2019)	89.05	66.44	78.05	SA	Uni	85.4	83.5	80.5	78.3
				AA	Uni	82.0±0.10	80.8±0.10	78.9±0.03	77.0±0.15
				MSA_s^∞	Uni	81.8±0.03	81.0±0.10	79.1±0.15	77.7±0.07
				MSA_s^∞	MSA_c	81.9±0.07	80.7±0.06	78.5±0.17	76.5±0.03
				AA	MSA_c	81.9±0.09	80.6±0.12	78.3±0.07	76.2±0.03
				MSA_s^2	MSA_c	**81.6±0.03**	**80.4±0.00**	**77.2±0.00**	**75.7±0.09**

is not hard-coded into the controller but identified on-the-fly during the attack (so behavior can differ for models with different vulnerabilities).

Table 3.8 demonstrates the ablation studies with respect to the used controllers on the model by Gowal et al. (2021). We observe that MSA_s combined with the uniform color distribution alone significantly improves the results of SA and AA in most of the query budgets (with the exception of the strong extrapolation regime of 5000 queries). The controller MSA_c combined with all schedules improves the performance with the exception of 500 queries regime for SA and MSA_s where it provides an equal or a slightly worse result in some cases.

Since our controllers are functions of 2 inputs as described in the Section 3.1.3 we can illustrate

Table 3.8: We compare the update size controller MSA_s to the schedules from SA (Andriushchenko et al., 2020) and AA (Croce and Hein, 2020b) in the ℓ_∞ threat model with $\epsilon = 8/255$ on a model by Gowal et al. (2021) with 1000 CIFAR10 test images. We also compare the uniform color sampling to our color controller MSA_c. Averaged across 5 runs with different random seeds.

Update size schedule	Color sampling	Query budget 500	1000	2500	5000
SA	Uniform	80.6±0.09	76.7±0.07	70.8±0.14	67.5±0.07
SA	MSA_c	81.0±0.05	76.4±0.08	69.9±0.10	**67.2±0.07**
AA	Uniform	80.0±0.17	76.8±0.11	72.2±0.10	69.2±0.08
AA	MSA_c	79.9±0.12	76.7±0.03	71.6±0.17	68.8±0.07
MSA_s	Uniform	**76.9±0.05**	73.7±0.05	69.8±0.13	67.6±0.04
MSA_s	MSA_c	**76.9±0.07**	**73.4±0.13**	**69.0±0.08**	**67.2±0.04**

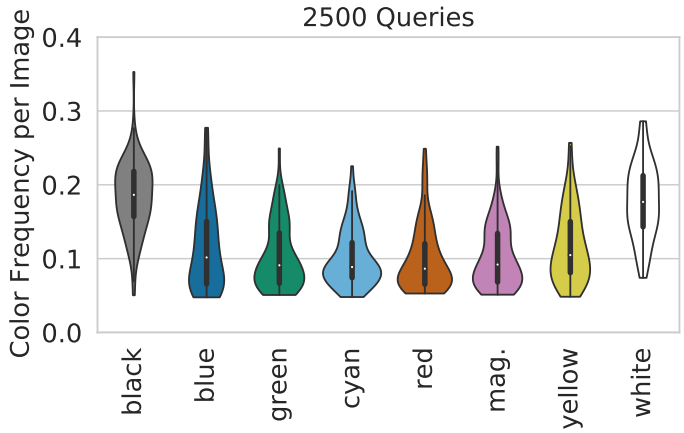
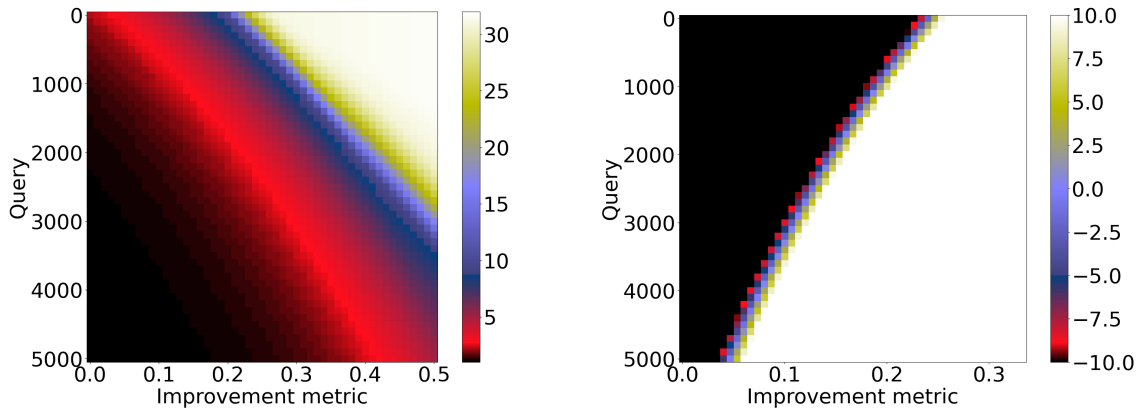


Figure 3.7: Illustration of color frequency histogram of 500 images after 2500 queries of MSA_c.

the dependence of their outputs on these inputs. We show it in Figure 3.8a for the update size controller and Figure 3.8b for the color controller.

The Figure 3.9 illustrates observed schedules for idealized (and untypical) target schedules: these target schedules are unknown to the controller and are encoded in the success probabilities by setting $p(r^t = 1) = 0.4$ for update sizes smaller or equal to the value of the target schedules and to $p(r^t = 1) = 0.1$ otherwise. This abrupt change of the success probabilities and the shape of the target schedules "constant" and "linear" are very unlike the behavior of the attacks during meta-training; nevertheless the empirical schedules by the controller follow the target behavior reasonably good, indicating that the learned square-size controller generalizes well.



(a) Update size controller

(b) Color controller

Figure 3.8: Additional analysis of the meta-learned controllers. The plot of the (a) update size controller MSA_s and (b) color controller MSA_c as functions of their inputs.

3.3 Possible future applications of the meta-learning framework

In this work, we have mostly focused on applying and evaluating the proposed general framework (Chapter 3.1.2]) for the case of Square Attack with different ℓ_p threat models. In this section, we would like to briefly discuss the opportunity for applying our methodology for other search-based methods. One example of this can be SimBA (Guo et al., 2019). The attack has two parameters: the step size ϵ and the set of orthonormal vectors Q for sampling the update. Guo et al. (2019) show that the results of their attack are dependent on both parameters. In particular, they compare two hand-crafted options of cartesian basis and discrete cosine basis. However, the trade-off between attack efficiency and success rate with different parameters is not well-studied. There were attempts to enhance the attack by applying Output Diversified Sampling (Tashiro et al., 2020) to the set Q. We believe that our method can provide an alternative view on this problem. In particular, one can see the analogy between the step size parameter ϵ for the SimBA and update schedule for the Square Attack. Or the analogy between the update set Q in the SimBA and finding the update strategy in the Square Attack by controlling the update color. As was mentioned above, we have focused on the Square Attack because it outperforms most of other score-based attacks including SimBA (Andriushchenko et al., 2020) but we believe that applying our meta-learning framework can improve other search-based attacks as well.

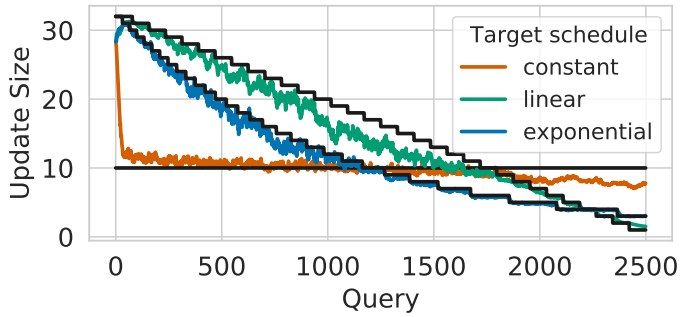


Figure 3.9: MSA$_s$ update size schedule adjustment to the target schedules (averaged over 25 runs).

3.4 Conclusion

In this work we propose a theoretical framework for meta-learning search distributions that help to improve efficiency of random search based black-box adversarial attacks. We implement and investigate this framework for Square Attack with ℓ_∞ and ℓ_2 perturbations. Our experimental results show that learned adaptive controllers improve attack performance across different query budgets and generalize to new datasets as well as targeted attacks. Future directions may include applying our framework to other random-search based attacks and threat models as well as learning controllers for sampling positions or even geometric primitives (going beyond squares).

Chapter 4

BagCert

This work was published by us as a conference paper at ICLR 2021 (Metzen and Yatsura, 2021). In this chapter, we propose BagCert, which combines high certified accuracy (60% on CIFAR10 for 5×5 patches) and clean performance (86% on CIFAR10), efficient inference (43 seconds on a single GPU for 10.000 CIFAR10 test samples), and end-to-end training for robustness against patches of varying size, aspect ratio, and location. BagCert is based on the following contributions:

- We propose three different conditions that can be checked for certifying robustness. One of these corresponds to the condition proposed by Levine and Feizi (2020). However, we show that an alternative condition improves certified accuracy of the same model typically by roughly 3 percent points while remaining broadly applicable.
- We derive a loss function that directly optimizes for certified accuracy against a uniform distribution of patch sizes at arbitrary positions. This loss corresponds to a specific type of the well known class of margin losses.
- Similarly to Levine and Feizi (2020), we classify images via a majority voting over a large number of predictions that are based on small local regions of a single input. However, the proposed model achieves this via a single forward-pass on the unmodified input, by utilizing a neural network architecture with very small receptive fields, similar to BagNets (Brendel and Bethge, 2019). This enables efficient inference with surprisingly high clean accuracy and was concurrently proposed by Zhang et al. (2020b) and Xiang et al. (2021).

Contributions

Please refer to the Table 4.1 for the contributions of the co-authors. Maksym Yatsura (the author of the thesis) has contributed to developing the method, in particular, certification conditions

4.1.1 and 4.1.2 and the loss for the end-to-end model training (Section 4.1.5). Using BagNets (Brendel and Bethge, 2019) in the certification architecture was suggested by Jan Hendrik Metzen.

Table 4.1: Contributions of the co-authors to the content of this chapter

Author	Author position	Scientific ideas %	Data generation %	Analysis & interpretation %	Paper writing %
Jan Hendrik Metzen	1	80	80	70	70
Maksym Yatsura	2	20	20	30	30
Paper title:		Efficient Certified Defences against Patch Attacks on Image Classifiers			
Status in publication process:		Accepted at the International Conference on Learning Representations 2021			

4.1 Methodology

We introduce BagCert, a framework which consists of novel conditions for certifying robustness, a specific model architecture, and a new end-to-end training procedure. BagCert allows end-to-end training of classifiers whose robustness against adversarial patch attacks can be certified efficiently. We outline our approach for the task of image classification but note that it can be extended to other tasks with grid-structured inputs. We refer to Figure 1.2 for an illustration of the training phase and to Figure 4.1 for an illustration of certification of BagCert.

4.1.1 Threat Model

We consider a threat model in which an attacker can conduct an image-dependent patch attack. Let $x \in [0,1]^{w_{in} \times h_{in} \times c_{in}}$ be an input image of resolution $w_{in} \times h_{in}$ with c_{in} channels. Let p be a patch and l be a region of an image x having the same size as patch p.

We denote a set of feasible regions l as $\mathcal{L}$. For example, for a patch $p \in [0,1]^{n \times c_{in}}$ consisting of n pixels, $\mathcal{L}$ could be the set of all $w_p \times h_p$ rectangular regions l of an image x with $w_p \cdot h_p = n$. We define an operator A such that $A(x,p,l)$ is the result of placing a patch p onto an image x over a region l. We assume that the attacker has white-box knowledge of the model and conducts an input-dependent attack, that is attack region l and inserted patch p can be chosen for every input independently.

4.1.2 Certification

We base our method for certification on assuming a certain structure of the classifier. More specifically, we decompose the classifier into two components:

- A region scorer f_θ that maps from inputs x to region scores $s \in \{0,1\}^{w_{out} \times h_{out} \times c_{out}}$, where $w_{out} \times h_{out}$ is the output resolution, c_{out} is the number of classes, and θ are trainable parameters. Please note that we allow $\sum_c s_{i,j,c} \neq 1$.

- A spatial aggregator g that maps from region scores s to (global) class scores $S \in [0,1]^{c_{out}}$. In this work, we restrict g to be monotonically increasing, that is: for class c and two patch score maps $s^{(1)}$ and $s^{(2)}$ with $s^{(1)}_{i,j,c} \geq s^{(2)}_{i,j,c} \, \forall i,j$, we require $g(s^{(1)})_c \geq g(s^{(2)})_c \, \forall c$.

Generally, we base certification on upper bounding the effect of an actual attack in the threat model. For this, we only exploit architectural properties of f and g that are valid for any choice of model parameters θ. More specifically, we only exploit the output dependency map R of f, which we define as $R(l) = \{(i,j) \mid \exists x, \theta, p : f_\theta(A(x,p,l))_{i,j} \neq f_\theta(x)_{i,j}\}$. Informally, $R(l)$ is the set of all indices of the score map that can be affected by a patch applied at region l, for any choice of input x, patch p, and parameters θ. That is: the set of all outputs of f_θ whose receptive fields overlap with l. We discuss options for f and the resulting R in Section 4.1.4.

For input x with class label c_t and $s = f_\theta(x)$, we define the "worst-case" score map $s^{wc}(s,l)$ as

$$s^{wc}_{i,j,c}(s,l) = \begin{cases} s_{i,j,c} & \text{if } (i,j) \notin R(l) \\ 1 & \text{if } (i,j) \in R(l) \wedge c \neq c_t \\ 0 & \text{if } (i,j) \in R(l) \wedge c = c_t \end{cases}$$

Moreover, we define $\Delta_{i,j,c} = s_{i,j,c_t} - s_{i,j,c}$ and similarly $\Delta^{wc}_{i,j,c} = s^{wc}_{i,j,c_t} - s^{wc}_{i,j,c}$. It follows directly that $\Delta^{wc}_{i,j,c} = \Delta_{i,j,c} \, \forall (i,j) \notin R(l)$ and $\Delta^{wc}_{i,j,c} = -1 \, \forall (i,j) \in R(l), c \neq c_t$.

For certifying robustness in the threat model for input x with class label c_t, we need to show $g(f_\theta(A(x,p,l)))_{c_t} > g(f_\theta(A(x,p,l)))_c \, \forall c \neq c_t \, \forall l \in \mathcal{L} \, \forall p$. For this, it suffices to check

Condition 4.1.1. $g(s^{wc}(s,l))_{c_t} > g(s^{wc}(s,l))_c \, \forall c \neq c_t, \forall l \in \mathcal{L}$

Proof. Consider arbitrary $l \in \mathcal{L}$ and p and let $s^{adv} = f_\theta(A(x, p, l))$. With $s^{adv}_{i,j,c} \in \{0, 1\}$ we obtain[1]

$$s^{adv}_{i,j,c} \begin{cases} = s^{wc}_{i,j,c}(s, l) = s_{i,j,c}(s, l) & \text{if } (i, j) \notin R(l) \\ \leq s^{wc}_{i,j,c}(s, l) = 1 & \text{if } (i, j) \in R(l) \wedge c \neq c_t \\ \geq s^{wc}_{i,j,c}(s, l) = 0 & \text{if } (i, j) \in R(l) \wedge c = c_t \end{cases}$$

With g being monotonically increasing, we obtain $g(s^{adv})_{c_t} \geq g(s^{wc}(l))_{c_t}$ and for all $c \neq c_t$ $g(s^{wc}(l))_c \geq g(s^{adv})_c$. Condition 4.1.1 implies $g(s^{adv})_{c_t} > g(s^{adv})_c \, \forall c \neq c_t$. □

Checking the Condition 4.1.1 requires one forward-pass through f_θ to obtain $s = f_\theta(x)$ and $|\mathcal{L}|$ times the construction $s^{wc}(s, l)$ and the evaluation of g. We now consider a special case where this can be implemented very efficiently.

4.1.3 Spatial Sum Aggregation

For the case $g = g_\sum(s) = \sum_{i=1,j=1}^{w_{out},h_{out}} s_{i,j}$, Condition 4.1.1 simplifies to

Condition 4.1.2. $\min\limits_{c \neq c_t} \sum\limits_{i,j \notin R(l)} \Delta_{i,j,c} > |R(l)| \quad \forall l \in \mathcal{L}$

Proof. For all $c \neq c_t$, we exploit $\forall (i, j) \in R(l) : \Delta^{wc}_{i,j,c} = -1$. With Condition 4.1.2, we obtain

$$\begin{aligned} g_\sum(s^{wc}(l))_{c_t} - g_\sum(s^{wc}(l))_c &= \sum_{i=1,j=1}^{w_{out},h_{out}} s^{wc}_{i,j,c_t} - \sum_{i=1,j=1}^{w_{out},h_{out}} s^{wc}_{i,j,c} = \sum_{i=1,j=1}^{w_{out},h_{out}} \Delta^{wc}_{i,j,c} \\ &= \sum_{i,j \notin R(l)} \Delta^{wc}_{i,j,c} + \sum_{i,j \in R(l)} \Delta^{wc}_{i,j,c} \\ &= \sum_{i,j \notin R(l)} \Delta_{i,j,c} - |R(l)| > 0. \end{aligned}$$ □

We note that

$$\sum_{i,j \notin R(l)} \Delta_{i,j,c} = \sum_{i=1,j=1}^{w_{out},h_{out}} \Delta_{i,j,c} - \sum_{i,j \in R(l)} \Delta_{i,j,c_t}.$$

For the special case that all $R(l)$ are rectangular, $\sum_{i,j \in R(l)} \Delta_{i,j,c}$ can be computed efficiently for all $l \in \mathcal{L}$ simultaneously via integral images/summed-area tables (Crow, 1984). For instance, $R(l)$ is rectangular for l being rectangular input patches and the R resulting from an CNN with grid-aligned kernels.

[1] We would like to note that these are "trivial" lower and upper bounds for s^{adv} and we see the potential to improve upon these bounds in future work, for instance by relaxing $s \in [0, 1]^{w_{out} \times h_{out} \times c_{out}}$ and applying interval bound propagation (Gowal et al., 2019). However, the proposed simple bounds have the advantage of not requiring additional forward passes through the model and thus being computationally efficient.

For the case that the $R(l)$ are not all rectangular and $|\mathcal{L}|$ becomes large, checking Condition 4.1.2 can become prohibitively expensive. For this case, we derive a condition that corresponds to an upper bound on Condition 4.1.2 and can be evaluated in constant time with respect to $|\mathcal{L}|$:

Condition 4.1.3. $\min_{c \neq c_t} \sum_{i=1,j=1}^{w_{out},h_{out}} \Delta_{i,j,c} > 2R^{max}(\mathcal{L})$ with $R^{max}(\mathcal{L}) = \max_{l \in \mathcal{L}} |R(l)|$

Proof. $\Delta_{i,j,c} \leq 1$ implies $\sum_{i,j \in R(l)} \Delta_{i,j,c} \leq |R(l)| \leq R^{max}(\mathcal{L})$. For all $c \neq c_t$, using Condition 4.1.3:

$$\sum_{i,j \notin R(l)} \Delta_{i,j,c} = \sum_{i=1,j=1}^{w_{out},h_{out}} \Delta_{i,j,c} - \sum_{i,j \in R(l)} \Delta_{i,j,c_t} > 2R^{max}(\mathcal{L}) - R^{max}(\mathcal{L}) \geq |R(l)| \qquad \square$$

We note that Condition 4.1.3 corresponds to the condition proposed by Levine and Feize Levine and Feizi (2020). It is, however, a strictly weaker condition than Condition 4.1.2. Thus, Condition 4.1.2 is preferable if all $R(l)$ are rectangular or $|\mathcal{L}|$ is of moderate size. We refer to Figure 4.1 for an illustration of Condition 4.1.2 and 4.1.3.

4.1.4 Model

Crucially, the quality of the certification depends on $R^{max}(\mathcal{L}) = \max_{l \in \mathcal{L}} |R(l)|$: the larger this quantity becomes, the larger the left-hand side of Condition 4.1.2 or Condition 4.1.3 needs to be to fulfill the condition. We focus on the specific case where f_θ is realized by a convolutional neural network (CNN). In that case, $|R(l)|$ is determined fully by l and the receptive field of the CNN. More specifically, we obtain

$$R(l) = \{(i,j) \mid \exists (\tilde{i},\tilde{j}) \in l : |i - \tilde{i}| \leq \lfloor w_{rf}/2 \rfloor \wedge |j - \tilde{j}| \leq \lfloor h_{rf}/2 \rfloor\}$$

for a receptive field size of $w_{rf} \times h_{rf}$ and ignoring operation strides.

Receptive field sizes of CNNs are determined by the shapes of the convolutional kernels as well as operation strides. We propose using standard CNN architectures such as ResNets but replacing most 3×3 convolutions by 1×1 convolutions, using stride 1 in (nearly) all operations, and removing all dense layers. This results in a network with very small receptive field sizes and thus small $R(l)$. We note that the proposed architecture is similar to BagNets (Brendel and Bethge, 2019) and using this type of model was concurrently proposed for certifying robustness against patch attacks by Zhang et al. (Zhang et al., 2020b) and Xiang et al. (Xiang et al., 2021). BagNets obtain surprisingly high classification accuracy despite small receptive field sizes (Brendel and Bethge, 2019). Importantly, in contrast to BagNets, we do not apply a global average pooling on the final feature layer. This results in a dense output of shape $w_{out} \times h_{out} \times c_{out}$. The ratios w_{in}/w_{out} and h_{in}/h_{out} depend on the strides applied in the network and control mostly the com-

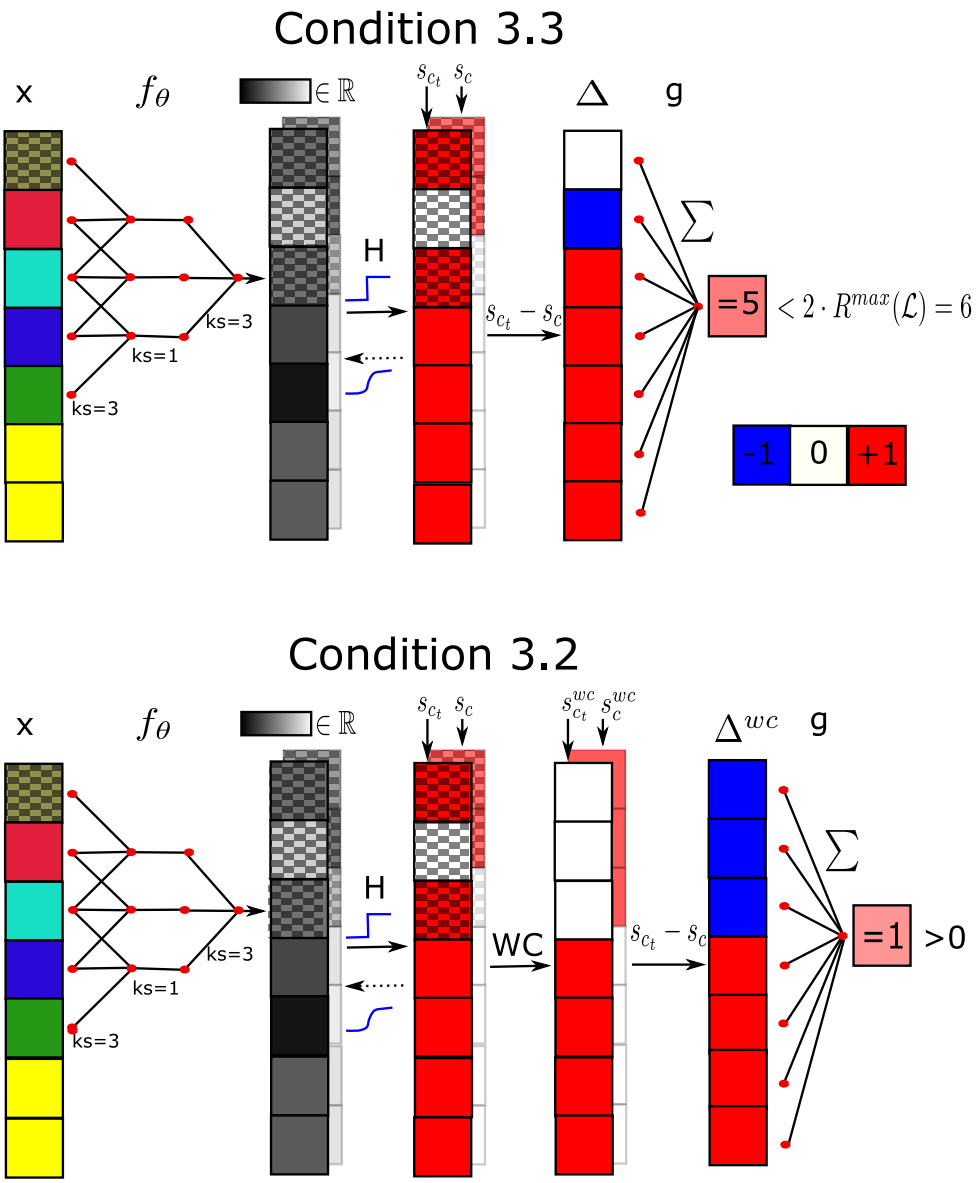


Figure 4.1: Illustration of BagCert certification for a 1D input and two classes. We assume for this example that $\mathcal{L}$ consists only of a single element; that is: the attacker can only place a patch at location $l = \{0\}$, shown by the checkerboard pattern in the input. The resulting $R(l)$ consists of the three top elements in region score space s (shown again by a checkerboard pattern). Accordingly, $R^{max}(\mathcal{L}) = 3$. (Top) Certification via Condition 4.1.3: The regular network output $+5$ is compared to $2 \cdot R^{max}(\mathcal{L}) = +6$. Since $5 \leq 6$, the robustness of the prediction cannot be certified. (Bottom) Certification via Condition 4.1.2: region scores s are replaced by s^{wc} based on $R(l)$. The resulting network output is $+1$, which is greater than 0. Thus, robustness of prediction can be certified.

putational overhead. We note that the cost for forward/backward passes in BagNets are in the same order of magnitude as those of a corresponding residual network. Because of the small receptive fields of BagNets, $|R(l)|$ is small if l is a small contiguous region of the input, such as a rectangular patch.

We apply a Heaviside step function

$$H(x) = \begin{cases} 0, \text{ for } x < 0 \\ 1, \text{ for } x \geq 0 \end{cases}$$

as final layer of f_θ, which ensures $f_\theta(X) \in \{0,1\}^{w_{out} \times h_{out} \times c_{out}}$. Similar to clipping Zhang et al. (Zhang et al., 2020b) and masking (Xiang et al., 2021) this also ensures that a patch cannot flip the global classification by perturbing a local score so strongly that it dominates the globally aggregated score. However, since H is constant nearly everywhere, it does not provide useful gradient information and thereby precludes end-to-end training. We address this by applying a "straight-through" type trick (Bengio et al., 2013) where we replace H in the backward pass by its smooth approximation, the logistic sigmoid function $s(x) = \frac{1}{1+e^{-x}}$. That is, we use $H(x)$ in the forward pass but replace the true gradient of H with $H'(x) := s'(x) = s(x)(1 - s(x))$.

While the proposed model computes $f_\theta(X)$ in a single forward-pass and controls $|R(l)|$ indirectly via the architecture of f, we note that alternative models are compatible with BagCert. For instance, one could compute every element of the output $s_{i,j}$ via a separate forward pass of an arbitrary model on an ablated (Levine and Feizi, 2020) or cropped version of the input similar to Mask-DS-ResNet (Xiang et al., 2021). This also ensures that a specific element of the output depends only on the cropper/non-ablated part of the input. While these works are more flexible in terms of model architecture, they require a number forward passes proportional to the resolution of the output s, which would make inference (and end-to-end) training computationally much more expensive.

4.1.5 End-to-End Training

Having derived conditions that can be used for certifying robustness against patch attacks in Section 4.1.2 as well as differentiable model for the region scorer f in Section 4.1.4, we now define a loss function for end-to-end training. We restrict ourselves to the case of a spatial sum aggregation $g_\sum$.

We recall Condition 4.1.3: $\min_{c \neq c_t} \sum_{i=1,j=1}^{w_{out},h_{out}} \Delta_{i,j,c} > 2R^{max}(\mathcal{L})$. The corresponding loss for this

can be defined as $L_H(\Delta, c_t, R^{max}) = H(\min_{c \neq c_t} \sum_{i=1,j=1}^{w_{out},h_{out}} \Delta_{i,j,c} \leq 2R^{max})$, that is: the loss is 1 if there is a target class c such that $\sum_{i=1,j=1}^{w_{out},h_{out}} \Delta_{i,j,c}$ becomes smaller/equal two times the size of the maximum affected patch score region. However, this requires choosing $\mathcal{L}$ and the resulting $R^{max}(\mathcal{L})$ before training, which is undesirable. Instead, we stay agnostic with respect to the specific $\mathcal{L}$ and simply assume a uniform distribution[2] for $R^{max}(\mathcal{L})$, that is $R^{max}(\mathcal{L}) \sim \mathcal{U}(0, R)$. Here, R corresponds to the maximum patch size (in region score space) we consider. This results in the loss

$$
\begin{aligned}
L_R(\Delta, c_t) &= \int_0^R p(\tilde{R}) L_H(\Delta, c_t, \tilde{R}) d\tilde{R} = \int_0^R \frac{1}{R} H(\min_{c \neq c_t} \sum_{i=1,j=1}^{w_{out},h_{out}} \Delta_{i,j,c} \leq 2\tilde{R}) d\tilde{R} \\
&= 1 - \frac{1}{R} \int_0^R H(\min_{c \neq c_t} \sum_{i=1,j=1}^{w_{out},h_{out}} \Delta_{i,j,c} > 2\tilde{R}) d\tilde{R} \\
&= 1 - \frac{1}{R} \min(\frac{1}{2} \min_{c \neq c_t} \sum_{i=1,j=1}^{w_{out},h_{out}} \Delta_{i,j,c}, R) = 1 - \frac{1}{2R} \min(\min_{c \neq c_t} \sum_{i=1,j=1}^{w_{out},h_{out}} \Delta_{i,j,c}, 2R).
\end{aligned}
$$

In practice, we minimize $\tilde{L}_R(\Delta, c_t) = -\min(\min_{c \neq c_t} \sum_{i=1,j=1}^{w_{out},h_{out}} \frac{\Delta_{i,j,c}}{w_{out} \cdot h_{out}}, M)$ with $M = \frac{2R}{w_{out} \cdot h_{out}}$. This loss can be interpreted as a margin loss with margin M, where the margin corresponds to twice the maximum patch size in region score space against which we want to become certifiably robust.

One-hot penalty While we do not strictly enforce $\sum_c s_{i,j,c} = 1$, we sometimes found it beneficial to add a term in the loss that encourages $S = g_{\sum}(s)$ being approximately "one-hot", that is $L_{oh}(S) = \max_{c \neq c_{max}} S_c - S_{c_{max}}$ with $c_{max} = \arg\max_c S_c$. Since $S_c \in [0, 1]$, it holds that $L_{oh}(S) \in [-1, 0]$ and $L_{oh}(S) = -1$ iff $S_{c_{max}} = 1$ and $S_c = 0 \; \forall c \neq c_{max}$. The term $L_{oh}(S)$ prevents training from prematurely converging to a solution where $s_{i,j,c}$ is approximately constant for all i, j, c, which we observed otherwise for tasks with many classes (e.g. ImageNet). The total loss becomes $L_{total} = \tilde{L}_R(\Delta, c_t) + \sigma L_{oh}(S)$, where σ controls the strength of this one-hot penalty.

4.2 Experiments

4.2.1 Experimental setup

We perform an empirical evaluation of BagCert on CIFAR10 (Krizhevsky, 2009) and ImageNet (Russakovsky et al., 2015). We report clean and certified accuracy and compare to Interval Bound Propagation (IBP) (Chiang et al., 2020), Derandomized Smoothing (DS) (Levine and

[2]We note that other choices than the uniform distribution would be an interesting direction for future work, in particular if the defender has prior knowledge about more likely patch sizes and shapes.

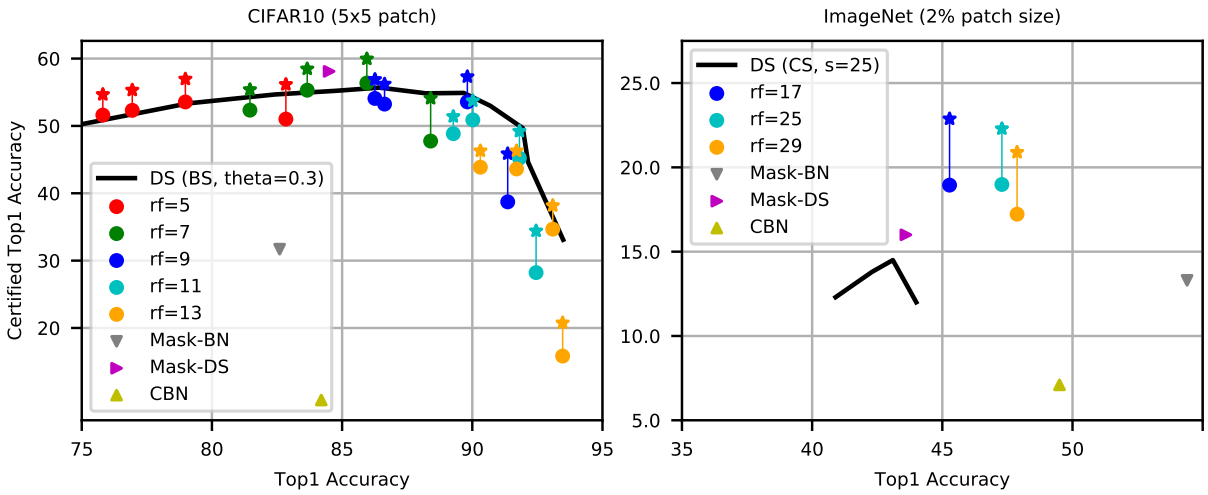


Figure 4.2: Clean versus certified accuracy on CIFAR10 and ImageNet for BagCert with different receptive fields and train margins ($M \in \{0.25, 0.5, 0.75, 1.0\}$ for CIFAR10, $M = 0.25$ for ImageNet) when certifying via Condition 4.1.3 (circles) and Condition 4.1.2 (stars), same setting connected by thin line. Smaller M generally corresponds to larger clean accuracy for CIFAR10. Baselines are Derandomized Smoothing (DS) (Levine and Feizi, 2020), Masked BagNet (Mask-BN) and Masked DS-ResNet (Mask-DS) (Xiang et al., 2021), and Clipped BagNet (CBN) (Zhang et al., 2020b). Results for these baselines are taken from the respective papers.

Feizi, 2020), Clipped BagNet (CBN) (Zhang et al., 2020b), and PatchGuard (Xiang et al., 2021). For DS, we focus on block-smoothing and for PatchGuard, we focus on the masked BagNet (Mask-BN) because column smoothing for DS (and the derived Mask-DS for PatchGuard) perform poorly for non-square patches that are "short-but-wide" (see Figure 4.4). We notice that column smoothing and Mask-DS perform better than column smoothing and Mask-BN against square-patches; however, there is no reason an attacker should prefer square over non-square rectangular patches. Moreover, we focus on certified accuracy, a lower bound on the actual robustness of a model.

CIFAR10. We use the following class of models for CIFAR10: we use a ResNet (He et al., 2016c) base architecture, consisting of a single 3x3 convolution stem, followed by 8 residual blocks. We use stride one in all operations (that is: output resolution is 32×32) and use a constant width of 768 throughout the network. The last layer consists of a 1×1 convolution with 10 outputs. All layers use batch normalization (Ioffe and Szegedy, 2015) and ReLU. Blocks get assigned either a kernel size of 1 or 3, depending on the desired receptive field of the network. The following table summarizes the kernel-sizes of different blocks used in the experiments:

RF of BagCert	stem	b1	b2	b3	b4	b5	b6	b7	b8
5	3	3	1	1	1	1	1	1	1
7	3	3	1	3	1	1	1	1	1
9	3	3	1	3	1	3	1	1	1
11	3	3	1	3	1	3	1	3	1
13	3	3	3	3	1	3	1	3	1

Residual blocks use shake-shake regularization (Gastaldi, 2017) in the batch-wise mode. For residual blocks with kernel size 3, a special form of shake-shake regularization is used: the first residual path applies a 3×3 convolution followed by a 1×1 convolution, while the second residual path applies first a 1×1 convolution followed by a 3×3 convolution. This increases diversity of paths without changing the total receptive field of the network. Besides that, no additional regularization is applied, that is weight decay is 0.0, and the one-hot penalty is set to $\sigma = 0.0$.

For training, we use the Adam optimizer with learning rate 0.001, batch size 96, and train for 350 epochs. We apply a cosine decay learning rate schedule (Loshchilov and Hutter, 2017a) with a warmup of 10 epochs. Moreover, we apply random horizontal flips and random crops with padding 4 for data augmentation.

ImageNet. We work on 224×224 inputs, which are extracted by rescaling the shorter side of the image to 256 pixels and extracting a random crop (training phase) or center crop (test phase) of size 224×224. Note that this input resolution differs from the 299×299 resolution used by Derandomized Smoothing Levine and Feizi (2020). In order to achieve comparable results, we evaluate against patches of size 32×32 ($32^2/224^2 \approx 2.04\%$) while DS test against patches of size 42×42 ($42^2/299^2 \approx 1.97\%$).

We use the following class of models for ImageNet: We use a ResNet base architecture, consisting of a single 3x3 convolution stem, followed by 8 residual blocks. We use stride 2 in blocks 1 and 3 and stride 1 otherwise (that is: output resolution is 56×56). We use width 64 in the stem and the first two blocks, width 128 in blocks 3 and 4, width 256 in blocks 5 and 6, and width 512 in blocks 7 and 8. The last layer consists of a 1×1 convolution with 1000 outputs. All layers use batch normalization and ReLU. Blocks get assigned either a kernel size of 1 or 3, depending on the desired receptive field of the network. The following table summarizes the kernel-sizes of different blocks used in the experiments:

RF of BagCert	5	7	9	11	13	DS (BS)	DS (CS)
Certification time (seconds)	39.0	40.6	43.2	45.9	48.5	788.0	28.0
Number of parameters	28M	38M	47M	57M	66M	11M	11M

Table 4.2: Certification time for 10.000 CIFAR10 test examples and number of model parameters.

RF of BagCert	stem	b1	b2	b3	b4	b5	b6	b7	b8
17	3	3	1	3	1	3	1	1	1
25	3	3	1	3	1	3	1	3	1
29	3	3	3	3	1	3	1	3	1

We apply neither shake-shake regularization nor weight decay. However, we set the one-hot penalty to $\sigma = 1.0$. For training, we use the Adam optimizer with learning rate 0.00033, batch size 64, and train for 60 epochs. We apply a cosine decay learning rate schedule with a warmup of 10 epochs. Moreover, we apply random horizontal flips for data augmentation.

4.2.2 Results

Figure 4.2 shows results for different methods against 5×5 patches for CIFAR10 corresponding to 2.4% of the image size and patches of 2% of the image size for ImageNet. For CIFAR10, when certifying accuracy via Condition 4.1.3, the Pareto frontier of BagCert follows closely the one reported for DS with block smoothing and $\theta = 0.3$. This is somewhat surprising given that both model and training procedure are very different and only the condition for certifying robustness is identical. We hypothesize that both approaches have reached close to optimal Pareto frontiers when certifying robustness via Condition 4.1.3. However, as Table 4.2 shows, BagCert requires (depending on its receptive field size) only between 39.0 and 48.5 seconds for certifying all 10.000 test examples on a single Tesla V100 SXM2 GPU while DS with block smoothing requires 788 seconds. BagCert also clearly dominates Mask-BN and CBN, which utilize a similar model architecture, as well as IBP (not shown) which reaches 47.8% clean and 30.3% certified accuracy. Moreover, when applying Condition 4.1.2 for certification, certified accuracy is increased by approx. 3 percent points without changes in clean accuracy or any noticeable increase in certification time. In summary, the strongest BagCert model with receptive field 7×7 and margin $M = 0.5$ can certify all 10.000 test examples in 43.2 seconds, reaching clean accuracy of 86% and certified accuracy of 60%.

On ImageNet, BagCert also dominates all baselines in terms of certified accuracy, reaching

18.9% via Condition 4.1.3 and 22.9% via Condition 4.1.2 for receptive field size 17 and margin $M = 0.25$. Running certification for the entire validation set of 50.000 images takes roughly 7 minutes.

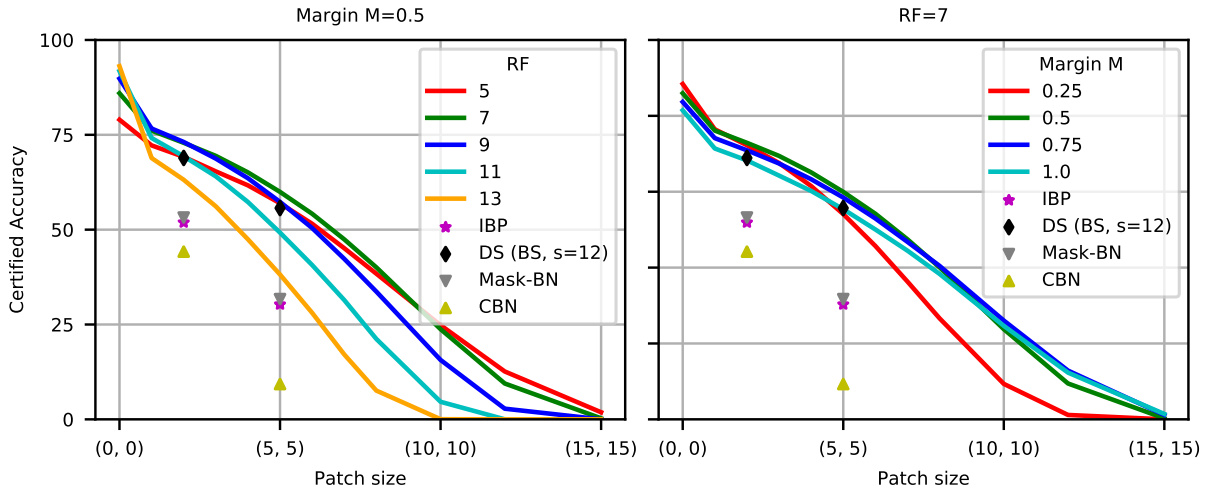


Figure 4.3: Certified accuracy against square patches of different sizes on CIFAR10. Shown is the performance for different receptive fields of BagCert (left) and train margins (right). Lines correspond to the same model (without retraining), evaluated against patches of different size.

Figure 4.3 shows accuracy of BagCert certified via Condition 4.1.2 for square patches of different sizes on CIFAR10. Again, baselines are dominated for both 2×2 and 5×5 patches. Moreover, a single configuration of BagCert with receptive field size 7 and margin $M = 0.5$ performs close to optimal for all patch sizes and can certify non-trivial performance for up to 10×10 patch size. This implies that a single model can be used for a broad range of threat models.

Figure 4.4 shows a similar analysis for non-square patches of a total size of 24 pixels. While BagCert with the same configuration as above achieves a certified accuracy of 40% or more for any patch aspect ratio, performance of DS with column smoothing varies greatly with aspect ratio. In particular, "short-but-wide" patches of shape 24×1 or 12×2 reduce certified accuracy of column smoothing close to 0%. Since there is no reason to assume attackers will restrict themselves to square patches, we do not consider DS with column smoothing or Mask-DS (Xiang et al., 2021) general patch defenses, despite good performance for square patches and efficient certification according to Table 4.2.

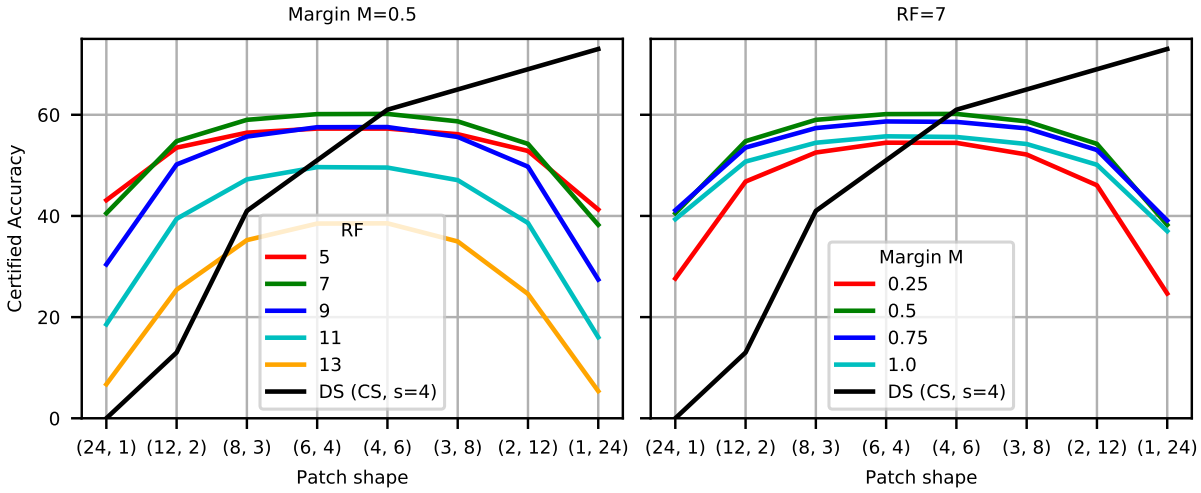


Figure 4.4: Certified accuracy against non-square patches of total size 24 pixels on CIFAR10. Shown is the performance for different receptive fields of BagCert (left) and train margins (right) compared to Derandomized Smoothing with Column-Smoothing (Levine and Feizi, 2020). Lines correspond to the same model (without retraining), evaluated against patches of different aspect rations.

4.2.3 Robustness against Heuristic Patch Attack

While the certification conditions proposed in Section 4.1.2 allow computing a *lower bound* of a model's robustness against a specific type of patch attack, a model's true robustness against such attacks can be anywhere between this lower bound and the clean accuracy. In order to determine a tighter *upper bound* on robustness than clean accuracy, we perform a heuristic adversarial patch attack on the model and evaluate the model's accuracy on inputs that were modified by the attacker. Our threat model from Section 4.1.1 allows an attacker to place an arbitrary patch $p \in [0,1]^{n \times c_{in}}$ at an arbitrary region $l \in \mathcal{L}$. We employ the following approach: we first select a region $l^* \in \mathcal{L}$ and target class c^*, and (once selected) keep this region and target class fixed and optimize the patch p accordingly. Please note that no guarantee exists that actually the best region for an attack or the best patch are determined; thus, the resulting adversarial accuracy is only an upper bound.

Specifically, we focus in this evaluation on 5×5 square patches on CIFAR-10. Accordingly, $\mathcal{L}$ consists of all possible 5×5 subregions of a 32×32 input. Ideally, one would perform independent attacks at all possible regions $l \in \mathcal{L}$. However, this becomes quickly computationally intractable. We exploit specific design choices of BagCert to select one region and target class that may be particularly problematic for a model on an input assuming a spatial sum aggregation is applied. For this, we make directly use of Condition 3.2 and choose $l^*, c^* = \arg\min_{l,c} \sum_{i,j \notin R(l)} \Delta_{i,j,c}$.

This choice corresponds to assuming a maximally effective patch attack that is able to achieve $\Delta_{i,j,c^*} = -1 \ \forall (i,j) \in R(l)$. A practical patch attack might not be able to achieve this ideal outcome (see also Figure 4.5) and thus l^*, c^* are not necessarily optimal. However, they are reasonable choices that can be determined efficiently.

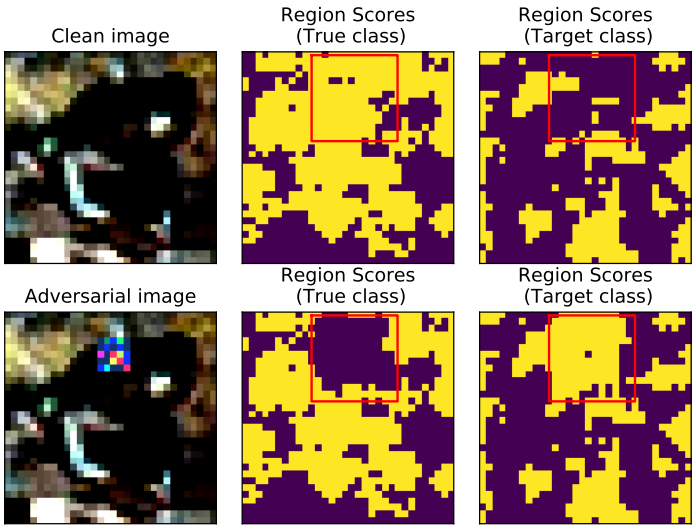


Figure 4.5: Illustration of an adversarial patch attack and its effect on region scores. Top row corresponds to the clean image (left), the resulting score maps for the true class (middle), and the score maps for the chosen target class (right). The bottom row shows the same for the image with an adversarial 5×5 patch inserted at the chosen region l. The red rectangle corresponds to $R(l)$.

Once l^* and c^* are fixed, we perform a PGD attack (Madry et al., 2018b) with 100 steps, a step size of 0.025, and the objective of maximizing the loss $\tilde{L}_R$ from Section 4.1.5 with margin $M = 1$. An illustration of such an attack is shown in Figure 4.5.

Figure 4.6 shows scatter plots of clean versus adversarial accuracy (left) and certified versus adversarial accuracy (right) for the BagCert models also shown in Figure 4.2. Interestingly, while clean and adversarial accuracy are highly correlated, the same does not hold true for certified and adversarial accuracy. In particular, adversarial accuracy seems to favor slightly larger receptive fields than certified accuracy. A potential reason for this can be seen in Figure 4.5: while a patch attack is typically effective for flipping the score of true and target class for the inner part of $R(l)$, it seems a lot harder to flip also scores close to the boundary of $R(l)$. For larger receptive fields of the model, this boundary effect seems to be amplified since the patch is smaller relative to the receptive field size. We consider reducing this gap between certified and adversarial accuracy further (that is: making bounds tighter) important future work. This will require both developing more effective attacks as well as improving certification procedures.

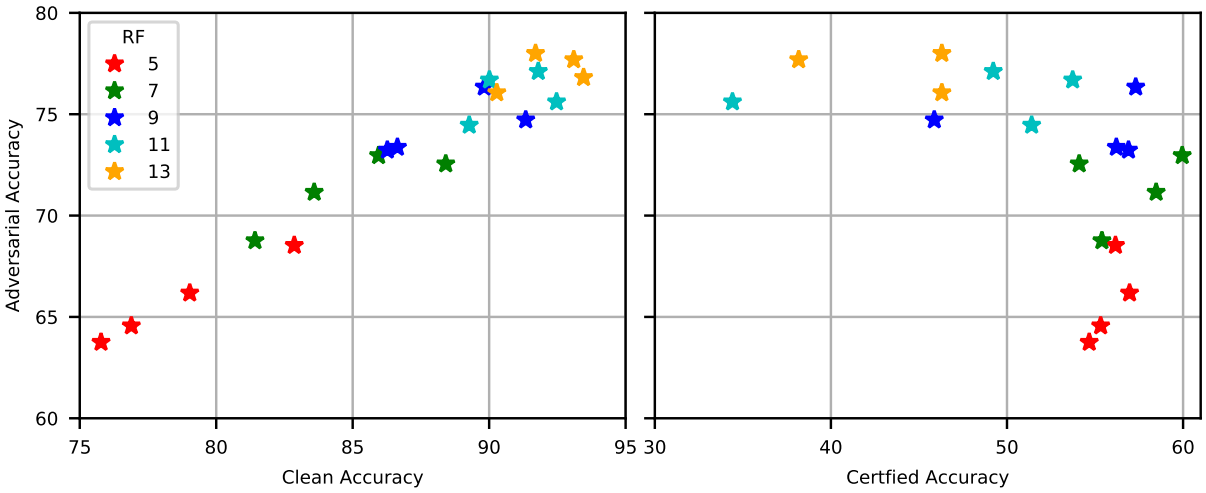


Figure 4.6: Scatter plots of clean versus adversarial accuracy (left) and certified versus adversarial accuracy (right). Color encodes different receptive filed sizes.

4.3 Conclusion

We have introduced a novel framework BagCert that combines efficient certification with end-to-end training for certified robustness. The main contributions are a model architecture based on a CNN with small receptive field, certification conditions that are applicable to a broad range of models, and a margin-loss based objective that is derived from the certification condition. The resulting model achieves high certified robustness against patches with a broad range of sizes, aspect ratios, and locations on CIFAR10 and ImageNet. Promising directions for future work are the exploration of other choices for the spatial aggregation function g (such as ones using the “detect-and-mask” mechanism from PatchGuard (Xiang et al., 2021)) and corresponding certification conditions and losses that can be used for end-to-end training. Moreover, the development of alternative choices for models with small receptive fields could be promising, such as ones based on learnable receptive fields or based on self-attention.

Chapter 5

Demasked Smoothing

This work was accepted as a conference paper at the ICLR 2023 (Yatsura et al., 2023). We summarize our contributions in this chapter as follows:

- We propose Demasked Smoothing which is the first certified recovery or certified detection based defence against adversarial patch attacks on semantic segmentation models.
- Demasked Smoothing can do certified detection and certified recovery with any off-the-shelf segmentation model without requiring finetuning or any other adaptation.
- We implement Demasked Smoothing, evaluate it for different certification objectives and masking schemes. We can certify 65% of all pixels in certified detection for a 1% patch and 47% in certified recovery for a 0.5% patch for the BEiT-L (Bao et al., 2022) segmentation model on the ADE20K Zhou et al. (2017) dataset.

Contributions

Please refer to the Table 5.1 for the contributions of the co-authors. Kaspar Sakmann has helped with implementing the dataloading, pointed out to using ZITS (Dong et al., 2022) model for demasking and trained a GIN model Li et al. (2020a) on ADE20K for experiments in Table 5.9. Grace Hua has helped with performing patch attacks 5.10 and debugging the experimental pipeline. Matthias Hein and Jan Hendrik Metzen have contributed to developing the method 5.2, evaluation metrics 5.4.2 analyzing the experimental results 5.4. Matthias Hein has contributed to the writing of the Section 5.8.2.

Table 5.1: Contributions of the co-authors to the content of this chapter

Author	Author position	Scientific ideas %	Data generation %	Analysis & interpretation %	Paper writing %
Maksym Yatsura	1	86	86	86	92
Kaspar Sakmann	2	2	8	2	2
Grace Hua	3	2	2	2	2
Matthias Hein	4	5	0	5	2
Jan Hendrik Metzen	5	5	4	5	2
Paper title:		Certified Defences against Adversarial Patch Attacks on Image Segmentation			
Status in publication process:		Accepted at the International Conference on Learning Representations 2023			

(a) image 1 (ADE20K) (b) segmentation 1 (c) image 2 (COCO10K) (d) segmentation 2

Figure 5.1: Semantic segmentation examples with images from ADE20K (Zhou et al., 2017) (image 1) and COCO10K (Caesar et al., 2018) (image 2). Each pixel is assigned a certain sematic class. Different classes are illustrated with different colors in the segmentation map.

5.1 Problem Setup

5.1.1 Semantic Segmentation

In this work, we focus on the semantic segmentation task. Let $\mathcal{X}$ be a set of rectangular images. Let $x \in \mathcal{X}$ be an image with height H, width W and the number of channels C. We denote $\mathcal{Y}$ to be a finite label set. The goal is to find the segmentation map $s \in \mathcal{Y}^{H \times W}$ for x. For each pixel $x_{i,j}$, the corresponding label $s_{i,j}$ denotes the class of the object to which $x_{i,j}$ belongs (see Figure 5.1). We denote $\mathbb{S}$ to be a set of segmentation maps and $f : \mathcal{X} \to \mathbb{S}$ to be a segmentation model.

5.1.2 Threat model

Let us consider an untargeted adversarial patch attack on a segmentation model. Consider an image $x \in [0, 1]^{H \times W \times C}$ and its ground truth segmentation map s. Assume that the attacker can

modify an arbitrary rectangular region of the image x which has a size of $H' \times W'$. We refer to this modification as a *patch*. Let $l \in \{0,\ 1\}^{H \times W}$ be a binary mask that defines the patch location in the image in which ones denote the pixels belonging to the patch. Let $\mathcal{L}$ be a set of all possible patch locations for a given image x. Let $p \in [0, 1]^{H \times W \times C}$ be the modification itself. Then we define an operator A as

$$A(x,\ p,\ l) = (1 - l) \odot x + l \odot p, \tag{5.1}$$

where $\odot$ is element-wise product. The operator A applies the $H' \times W'$ subregion of p defined by a binary mask l to the image x while keeping the rest of the image unchanged. We denote $\mathcal{P} := [0,\ 1]^{H \times W \times C} \times \mathcal{L}$ to be a set of all possible patch configurations (p, l) that define an $H' \times W'$ patch. Let $s \in \mathbb{S}$ be the ground truth segmentation for x. Let $Q(f(x), s)$ be some quality metric such as global pixel accuracy or mean intersection over union (mIoU). The goal of an attacker is to find $(p^\star,\ l^\star)$ s. t.

$$(p^\star,\ l^\star) = \underset{(p,\ l) \in \mathcal{P}}{\arg\min}\ Q(f(A(x, p, l)), s) \tag{5.2}$$

5.1.3 Defence objective

In this chapter, we propose certified defences against patch attacks. It means that we certify against *any possible attack* from $\mathcal{P}$ including $(p^\star,\ l^\star)$. We consider two robustness objectives.

Certified recovery. For a pixel $x_{i,j}$ our goal is to verify that the following statement is true

$$\forall\ (p,\ l) \in \mathcal{P} :\ f(A(x,\ p,\ l))_{i,j} = f(x)_{i,j} \tag{5.3}$$

Certified detection. We consider a verification function v defined on $\mathcal{X}$ such that $v(x) \in \{0, 1\}^{H \times W}$. If $v(x)_{i,j} = 1$, then the adversarial patch attack on $x_{i,j}$ can be detected by applying the function v to the attacked image $x' = A(x, p, l)$.

$$v(x)_{i,j} = 1 \Rightarrow \Big[\forall\ (p,\ l) \in \mathcal{P} : v(A(x, p, l))_{i,j} = 1 \rightarrow f(A(x,\ p,\ l))_{i,j} = f(x)_{i,j}\Big] \tag{5.4}$$

$v(x')_{i,j} = 0$ means an alert on pixel $x'_{i,j}$. However, if x' is not an adversarial example, then this is a false alert. In that case the fraction of pixels for which we return false alert is called *false alert ratio* (FAR). The secondary objective is to keep FAR as small as possible.

Depending on the objective, our goal is to certify one of the conditions 5.3, 5.4 for each pixel $x_{i,j}$. This provides us an upper bound on an attacker's effectiveness under any adversarial

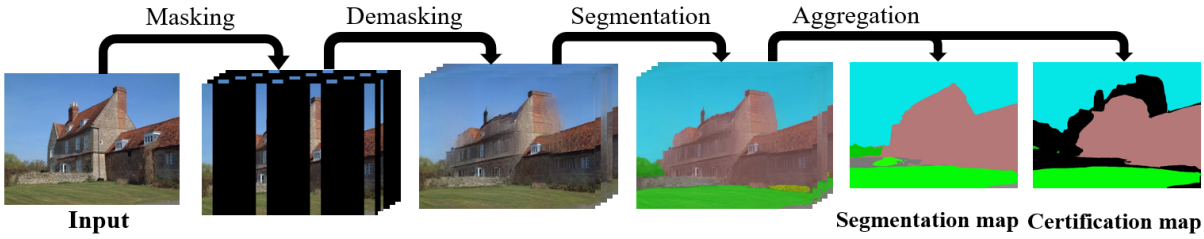


Figure 5.2: A sketch of Demasked Smoothing for certified image segmentation. First, we generate a set of masked versions of the image such that each possible patch can only affect a certain number of masked images. Then we use image inpainting to partially recover the information lost during masking and then apply an arbitrary segmentation method. The output is obtained by aggregating the segmentations pixelwise. The masking strategy and aggregation method depend on the certification mode (detection or recovery).

patch attack from $\mathcal{P}$.

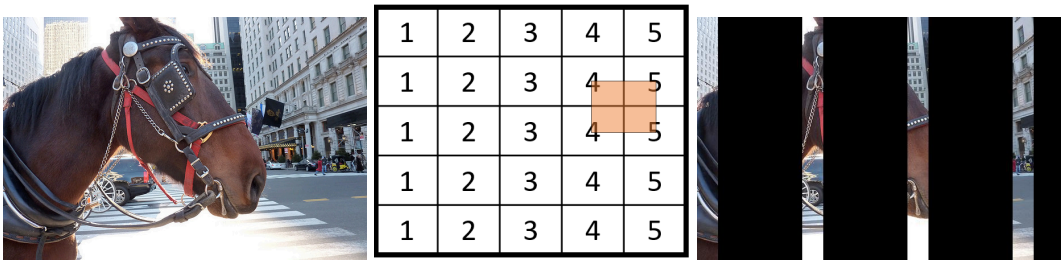
5.2 Demasked Smoothing

Demasked Smoothing (Figure 5.2) consists of several steps. First, we apply a predefined set of masks with specific properties to the input image to obtain a set of masked images. Then we reconstruct the masked regions of each image based on the available information with an inpainting model g. After that we apply a segmentation model f to the demasked results. Finally, we aggregate the segmentation outcomes and make a conclusion for the original image with respect to the statements (5.3) or (5.4).

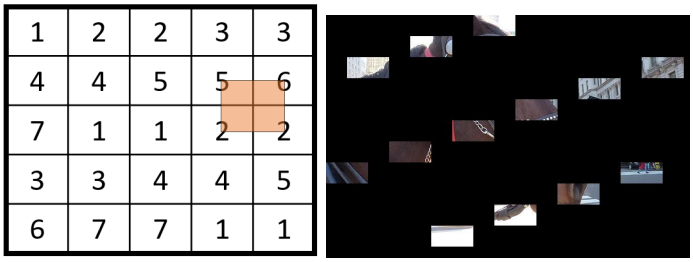
5.2.1 Input masking

Motivation. Like in previous work (Section 2.3.3) we apply masking patterns to the input image and use predictions on masked images to aggregate the robust result. If an adversarial patch is completely masked, it has no effect on further processing. However, in semantic segmentation, we predict not a single whole-image label like in the classification task, but a separate label for each pixel. Thus, making prediction on a masked image must allow us to predict the labels also for the masked pixels.

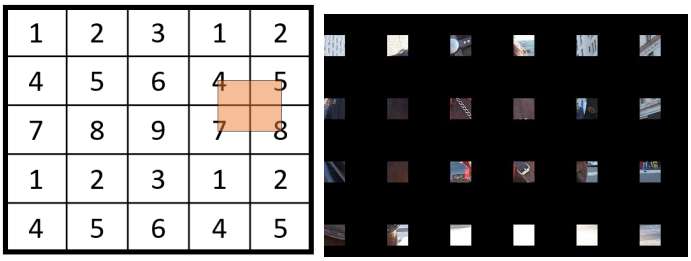
Preliminaries. Consider an image $x \in [0,1]^{H \times W \times C}$. We define "$*$" to be a special masking symbol that does not correspond to any pixel value and has the property $\forall z \in \mathbb{R} : \; z \times * = *$. Please note that $*$ needs to be different from 0 since 0 is a valid pixel value in unmasked inputs. Let $m \in \{*, 1\}^{H \times W}$ be a *mask*. We call the element-wise product $x \odot m$ a *masking* of x. In a masking, a subset of pixels becomes $*$ and the rest remains unchanged. We consider the



(a) original image (b) $T = 2$ (c) column mask



(d) $T = 3$ (e) 3-mask



(f) $T = 4$ (g) 4-mask

Figure 5.3: Examples of the masks for certified recovery. Column masks: $T = 2$ (b, c), $K = 5$ masks. 3-mask: $T = 3$ (d, e), $K = 7$ masks. 4-mask: $T = 4$ (f, g), $K = 9$ masks. The number on each block in (b, d, f) denotes in which mask the block is visible. There is only one such mask for each block. For each mask set, we show one of the locations l in which an adversarial patch $(p,\ l)$ affects T different maskings.

threat model $\mathcal{P}$ with patches of size $H' \times W'$ (Section 5.1.2).

Certified recovery. To define the structure of our masks, we break m into an array B of non-intersecting blocks, each having the same size $H' \times W'$ as the adversarial patch. We index the blocks as

$$B[q,\ r], \quad 1 \leq q \leq \lceil H/H' \rceil, \quad 1 \leq r \leq \lceil W/W' \rceil. \tag{5.5}$$

We say that the block $B[q,\ r]$ is *visible* in a mask m if $\forall (i,\ j) \in B[q,\ r] :\ m_{i,j} = 1$ Consider an array M of K masks. We define each mask $M[k]$ by a set of blocks that are visible in it. For certified recovery, each block is visible in exactly one mask and masked in the others. We say

that a mask m is *affected* by a patch (p, l) if $A(x,\ p,\ l) \odot m \neq x \odot m$. We define

$$T(M) = \max_{(p,l)\in\mathcal{P}} |\{m \in M\ |A(x,\ p,\ l) \odot m \neq x \odot m\}| \tag{5.6}$$

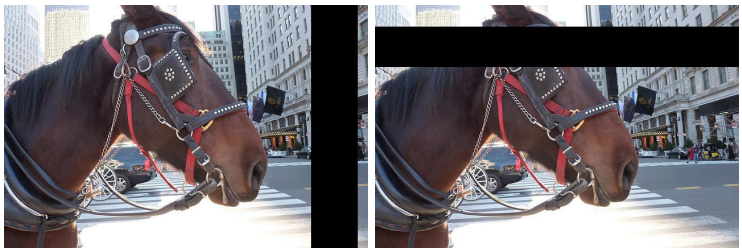
That is: $T(M)$ is the largest number of masks that one patch can possibly affect. If M is defined, we refer to the value $T(M)$ as T for simplicity. For a set M of K masks we define the mapping $\mu_M :\ B \to \{1, \ldots, K\}$. If $\mu(B[q,\ r]) = k$, then $B[q,\ r]$ is not masked in $M[k]$. Therefore, each mask $M[k]$ is defined by a $B_k \subset B$ s. t. for $b \in B_k\ \mu(b) = k$.

We define column masking M for which $T = 2$. We assign every k-th block column to be visible in the mask $M[k]$, $\mu(B[q,\ r]) = k$ if $r\%k = 0$ (Figure 5.3c). Any $(p, l) \in \mathcal{P}$ can intersect at most two adjacent columns since (p, l) has the same width as a column. Thus, it can affect at most two masks (Figure 5.3b). A similar scheme can be proposed for the rows.

We define a set M that we call *3-mask* for which $T(M) = 3$. We assign the blocks in each row to the masks as follows: $\mu(B[1,\ 1]) = 1$; $\mu(B[1,\ 2]) = \mu(B[1,\ 3]) = 2$; $\mu(B[1,\ 4]) = \mu(B[1,\ 5]) = 3$ and so on until we reach the end of the row. If we finish the first row with the value k, then we start the second row as follows $\mu(B[2,\ 1]) = \mu(B[2,\ 2]) = k+1$; $\mu(B[2,\ 3]) = \mu(B[2,\ 4]) = k+2$: .... If we finish the second row on n, we start the third row similarly to the first: $\mu(B[3,\ 1]) = n + 1$; $\mu(B[3,\ 2]) = \mu(B[3,\ 3]) = n + 2$; ... When we reach the number K, we start from 1 again (Figure 5.3d). Due to the block size, the patch cannot intersect more than four blocks at once. Our parity-alternating block sequence ensures that in any such intersection of four blocks either the top ones or the bottom ones will belong to the same masking, so at most three different maskings can be affected.

We define a set M that we call *4-mask* for which $T(M) = 4$. Due to our block size any assignment of masks will work because the patch cannot intersect more than four blocks. We consider the one that allows uniform distribution of the unmasked blocks (Figure 5.3g). We point out that for the described methods each masking keeps approximately $1/K$ of the pixels visible and the unmasked regions are uniformly distributed in the image. This means that for any masked pixel there exists an unmasked region located close enough to this pixel. It is the core difference between our masks and the ones proposed for certified classification such as block or column smoothing Levine and Feizi (2020). It was observed that the image demasking is facilitated when the visible regions are uniformly spread in the masked image He et al. (2021).

Certified detection. We define M_d to be a set of masks for certified detection (we use subscript d for distinction). M_d should have the property: $\forall\ (p, l) \in \mathcal{P}\ \exists\ m \in M_d : A(x,\ p,\ l) \odot m = x \odot m$



(a) column masking (b) row masking

Figure 5.4: Column masking and row masking for certified detection.

i. e. for every patch exists at least one mask not affected by this patch. For a patch of size $H' \times W'$ we consider $K = W - W' + 1$ masks such that the mask $M_d[k]$ masks a column of width W' starting at the horizontal position k in the image (Figure 5.4a). To obtain the guarantee for the same $\mathcal{P}$ with a smaller K, we consider a set of strided columns of width $W'' \geq W'$ and stride $W'' - W' + 1$ that also satisfy the condition.

Lemma 1. Consider an image of the size $H \times W$. Let $H' \times W'$ be a fixed adversarial patch size. Let M^d be a set of K masks where each mask is masking an $H \times W''$ vertical column, $W'' \geq W'$. Let the stride between the columns in two adjacent masks be $W'' - W' + 1$. Then for any location $l \in \mathcal{L}$ of the patch, there exists a mask that covers it completely.

Proof. (Adapted from the proof of Lemma 4 in Patch Cleanser (Xiang et al., 2022a)). Without loss of generality, we consider the first two adjacent column masks. The first one covers the columns from 1 to W''. The second mask covers the columns from $1 + (W'' - W' + 1) = W'' - W' + 2$ to $(W'' - W' + 2) + (W'' - 1) = 2W'' - W' + 1$ (See Figure 5.5). Now consider an adversarial patch of size $H' \times W'$. Let us find the smallest possible start index of this patch so that it does not get covered by the first mask. For that it should be visible at the column $W'' + 1$ and, therefore, start at the column with index not smaller than $(W'' + 1) - W' + 1 = W'' - W' + 2$. However, it is the same column in which second mask starts. Therefore, given that $W'' \geq W'$ we have that the patch is completely masked by the second mask. Then for a patch which is only partially masked by the second mask from the left we use an analogous argument to show that it is completely masked by the third mask and so on. □

A similar scheme can be proposed for the rows (Figure 5.4b). Alternatively, we could use a set of block masks of size $H' \times W'$. Then the number of masks grows quadratically with the image resolution. Hence, in the experiments we focus on the column and the row masking schemes.

Segmentation array. Let g be a demasking model, $g(x \odot m) \in [0, 1]^{H \times W \times C}$. The goal of g is

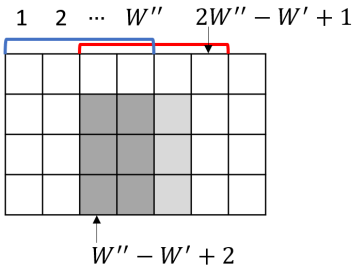


Figure 5.5: The masked columns of the first two adjacent masks (blue for the first one and red for the second one). If the patch is not completely masked by the first mask, it should be visible at the column $W'' + 1$ (the masked part of the patch is dark-grey and the visible part is in light-grey). However then the patch will be completely masked by the second mask.

to make the reconstruction $g(x \odot m)$ as close as possible (in some metric) to the original image x. For a segmentation model f we define a *segmentation array* $S(M, x, g, f)$:

$$S[k] := f(g(x \odot M[k])), \quad 1 \leq k \leq K. \tag{5.7}$$

5.2.2 Certification

Certified recovery. For the threat model $\mathcal{P}$ (Section 5.1) consider a set M of K masks. We define a function $h : \mathcal{X} \to \mathbb{S}$ that assigns a class to the pixel $x_{i,j}$ via majority voting over class predictions of each reconstructed segmentation in S. A class for the pixel that is predicted by the largest number of segmentations is assigned. We break the ties by assigning a class with a smaller index.

Theorem 1. If the number of masks K satisfies $K \geq 2T(M) + 1$ and for a pixel $x_{i,j}$ we have

$$\forall\, S[k] \in S :\ S[k]_{i,j} = h(x)_{i,j}$$

i.e. all the votes agree, then $\forall\ (p,\ l) \in \mathcal{P} :\ h(A(x, p, l))_{i,j} = h(x)_{i,j}$.

Proof. We prove the statement by contradiction. Assume that

$$\exists\ (p,\ l) \in \mathcal{P} :\ h(A(x, p, l))_{i,j} \neq h(x)_{i,j}$$

Let us denote $x' := A(x, p, l)$ and S' to be the segmentation array for x'. We denote the class $h(x)_{i,j}$ predicted for the pixel $x_{i,j}$ as C. $h(A(x, p, l))_{i,j} \neq C$ means that the class C did not get the majority in the votes over the segmentation array S'. However, by definition of $T(M)$ we

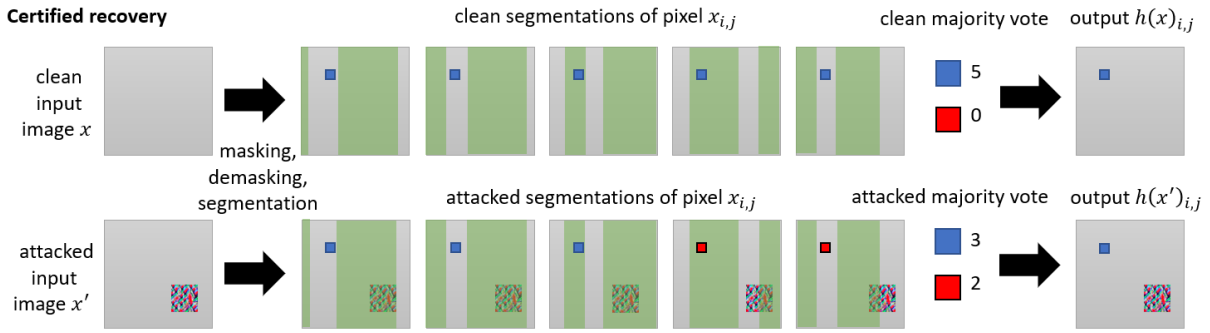


Figure 5.6: Illustration for Theorem 1 with $K = 5$ masks and $T = 2$. If the majority voting across the five masks is univocal for the clean image, the patch can only affect two out of five segmentations and, thus, cannot shift the majority. The green shade schematically shows which parts were masked in the masking step.

know that $(p,\ l)$ could affect at most $T(M)$ segmentations out of K and change their vote. Since all K segmentations of S have voted for C, then at least $K - T(M)$ of them are still voting for C in S'. And by our assumption $K \geq 2T(M) + 1$, we have that $K - T(M) \geq T(M) + 1$. Thus, the class C for $x_{i,j}$ still gets the majority vote in S'. Therefore $h(x')_{i,j} = C = h(x)_{i,j}$. We have arrived to a contradiction. □

A schematic illustration for the certified recovery mechanism is provided in Figure 5.6. In Section 5.2.1 we have stated that by construction each block is visible in exactly one mask. Assume that two masks keep the same block of the image visible. By placing a patch so that it intersects this block, the attacker can affect both of the corresponding segmentations. Thus, they can be counted as one when computing $T(M)$. It would result in producing redundant segmentations that do not affect the certification. Therefore, we consider the case of masks with non-overlapping visible blocks in Section 5.2.1. Since each masking keeps approximately $1/K$ of the image pixels visible, we are interested in keeping K as small as possible to maintain more visual information for the demasking model g. Thus, for the evaluation of masking schemes with $T = 2, 3, 4$ we use the smallest possible values of $K = 5, 7, 9$ respectively (Figure 5.3). We observe that although for $T = 2$ we can keep a larger fraction of pixels unmasked, $T = 3, 4$ provide visible blocks that are spread in the image more uniformly. We evaluate these masking approaches in Section 5.4.

Certified detection. Consider $M_d = \{M_d[k]\}_{k=1}^{K}$. For a set of demasked segmentations S we define the verification map $v(x)_{i,j} := [f(x)_{i,j} = S[1]_{i,j} = \ldots = S[K]_{i,j}]$ i.e. the original segmentation coincides with all the other segmentations including the one in which the potential patch was completely masked.

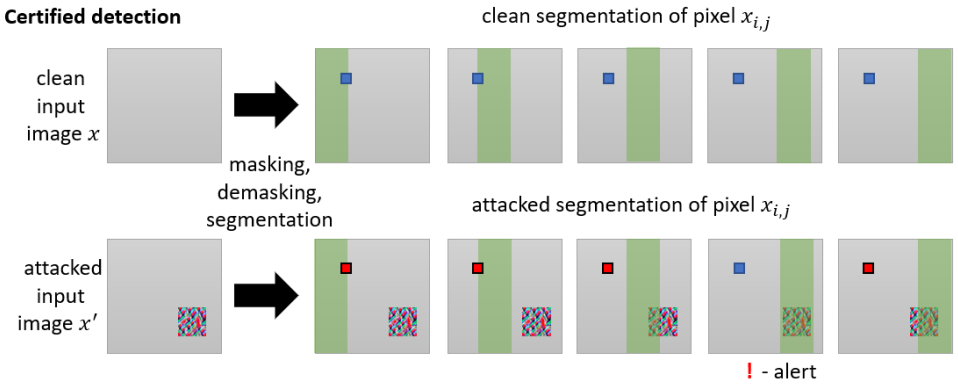


Figure 5.7: Illustration for Theorem 2. There exists at least one segmentation for which the patch was masked. If the patch has managed to affect the segmentations in which it is not masked, there will be an inconsistency. The green shade schematically shows which parts were masked in the masking step.

Theorem 2. Assume that $v(x)_{i,j} = 1$. Then

$$\forall\ (p,\ l) \in \mathcal{P} : v(A(x,p,l))_{i,j} = 1 \Rightarrow f(A(x,\ p,\ l))_{i,j} = f(x)_{i,j}$$

Proof. We prove the statement by contradiction. Assume that

$$\exists\ (p,\ l) \in \mathcal{P} :\ v(A(x,p,l))_{i,j} = 1 \wedge f(A(x,\ p,\ l))_{i,j} \neq f(x)_{i,j}$$

Let us denote $x' := A(x,p,l)$ and S' to be the segmentation set for x'. By definition of M_d, $\exists\ M_d[k] \in M_d$ s. t. $M_d[k]$ masks the patch $(p,\ l)$ Hence,

$$g(x \odot M_d[k]) = g(x' \odot M_d[k]),$$

$$S[k] = f(g(x \odot M_d[k])) = f(g(x' \odot M_d[k])) = S'[k],$$

Since $v(x)_{i,j} = 1$, we have $f(x)_{i,j} = S[k]_{i,j}$. Since $v(x')_{i,j} = 1$, we have $f(x')_{i,j} = S'[k]_{i,j}$. Thus, $f(x')_{i,j} = f(x)_{i,j}$. We have arrived to a contradiction. □

A schematic illustration for the certified recovery mechanism is provided in Figure 5.7. For a given image x the verification map $v(x)$ is complementary to the model segmentation output $f(x)$ that stays unchanged. Thus, there is no drop in clean performance however we may have some false positive alerts in the verification map v in the clean setting.

Summary. We present the general Demasked Smoothing procedure in Algorithm 1. The ex-

plicit Demasked Smoothing pipeline with the column masking is illustrated in Figure 5.8 for certified detection and in Figure 5.9 for certified recovery. See the illustrations for other masking schemes in the Section A.1 in the Appendix.

Algorithm 1 Demasked Smoothing

Input: image $x \in [0,1]^{H \times W \times C}$, patch size (H', W'), certification type CT (recovery or detection), mask type MT (column, row, 3-mask, 4-mask), inpainting model g, segmentation model f

Output: segmentation map $h \in \mathcal{Y}^{H \times W}$, certification (or verification) map $v \in \{0,\ 1\}^{H \times W}$

$M \leftarrow$ CreateMaskArray$(H, W, H', W', \text{CT}, \text{MT})$ ▷ according to section 5.2.1

for $k \leftarrow 1, \ldots, |M|$ **do** ▷ this loop can be paralellized

  $S[k] \leftarrow f(g(x \odot M[k]))$ ▷ mask input, inpaint the masked regions, and apply segmentation

end for

if CT $=$ 'recovery' **then** $h \leftarrow$ MajorityVote(S) ▷ vote over the classes predicted for each pixel

else $h \leftarrow f(x)$ ▷ in detection case, output clean segmentation

end if

$v \leftarrow$ AllEqual(S, h) ▷ assign 1 for the pixels where all $S[k]$ agree with h and 0 otherwise

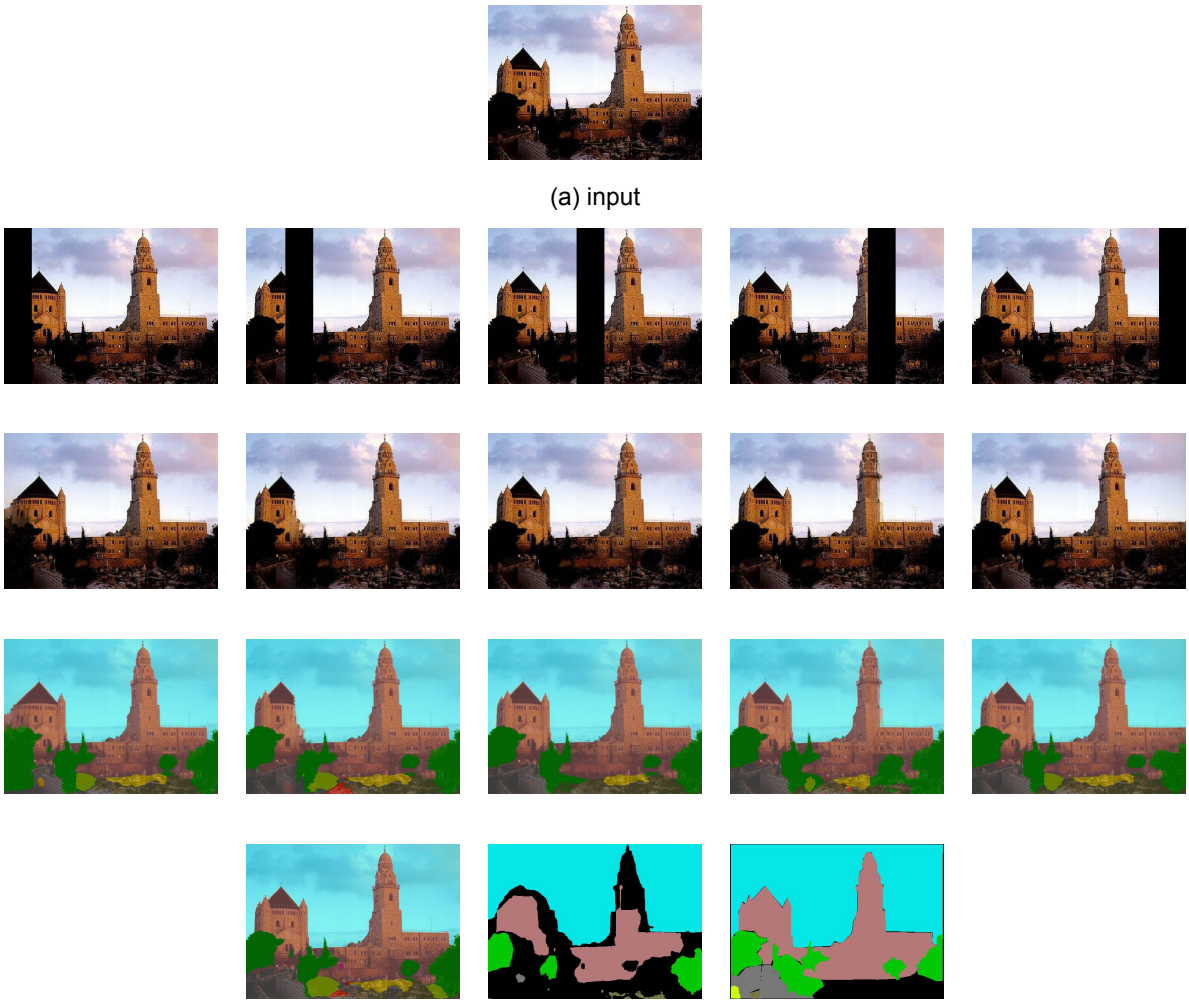
Return h, v

5.3 Defence example

In this section, we demonstrate an example of a real adversarial patch for a semantic segmentation model similar to the one illustrated in the Figure 1.3 and show how it is handled by our certified defences. We illustrate it for the Swin (Liu et al., 2021) model on one of the images from the ADE20K (Zhou et al., 2017) dataset.

5.3.1 Patch optimization

We set the patch size to 1% of the image surface. We select a fixed position for a patch on the rear window of a car (Figure 5.10a). For each pixel we extract a list of predicted logits corresponding to each class and apply multi-margin loss with respect to the ground truth label of the respective pixel. We use random patch initialization without restarts. As an optimizer we use projected gradient descent (PGD) with 1000 steps and initial step size of 0.01. We use cosine step size schedule and momentum for the gradient with the rate of 0.9. The optimization plot and the patch efficiency at different iterations of the PGD are illustrated in the Figure 5.10.



(a) input

(b) segmentation (c) certification (d) ground truth

Figure 5.8: Demasked Smoothing detection column masking illustration for an image from ADE20K (Zhou et al., 2017) with ZITS demasking(Dong et al., 2022) and BEIT-B segmentation (Bao et al., 2022). We illustrate five masks out of twenty.

5.3.2 Certified recovery

We denote the original image as x and the patched image as x'. The voting-based segmentation function h (Section 5.2.2) provides the majority-vote prediction $h(x)$ and the corresponding certification map which shows the pixels where all the votes agree. In Figure we see that a part of the building and the road is certified which means that this prediction cannot be affected by an adversarial patch. Figure demonstrates $h(x')$ which correctly segments those regions in presence of an adversarial patch that fools the original model.

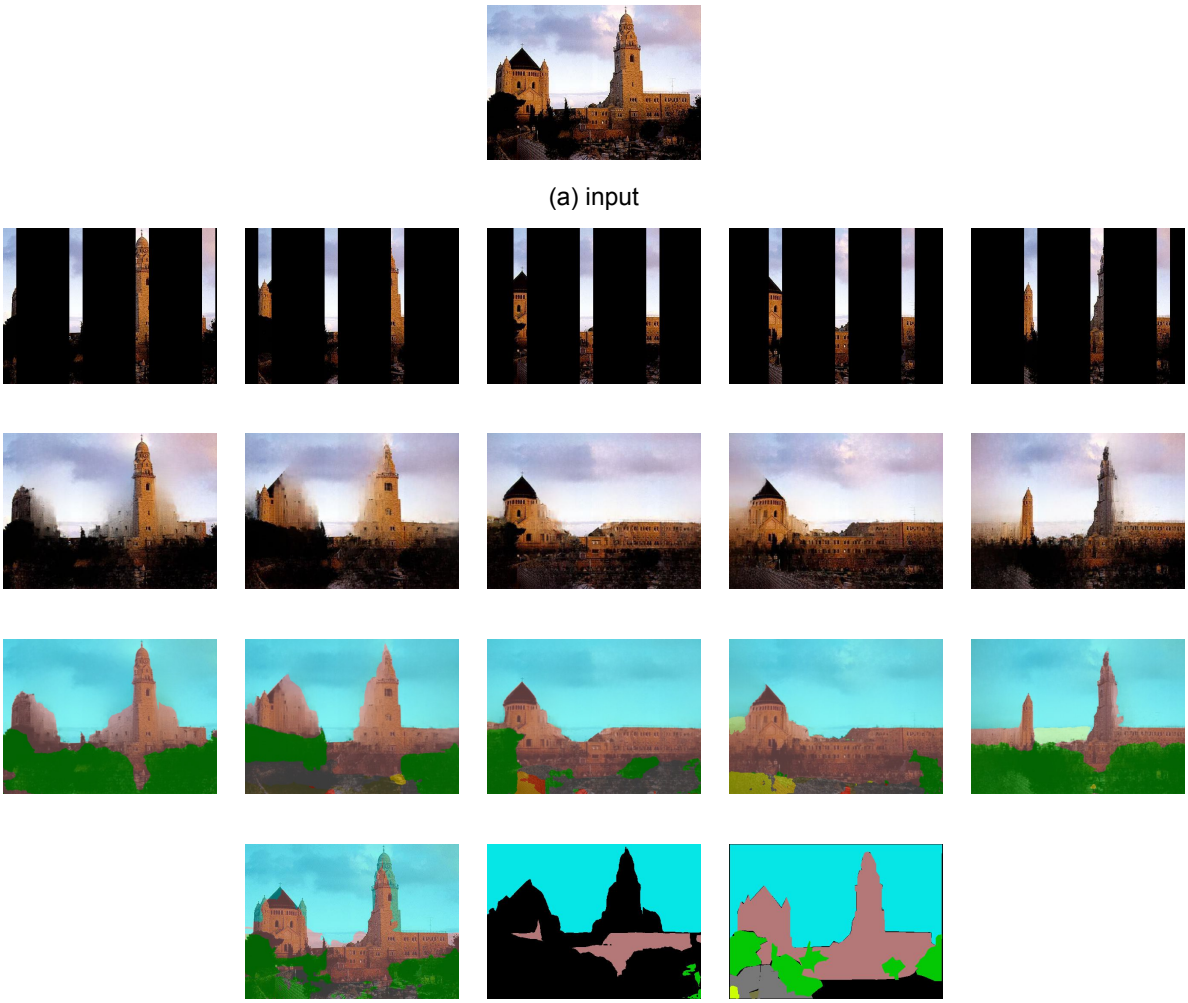
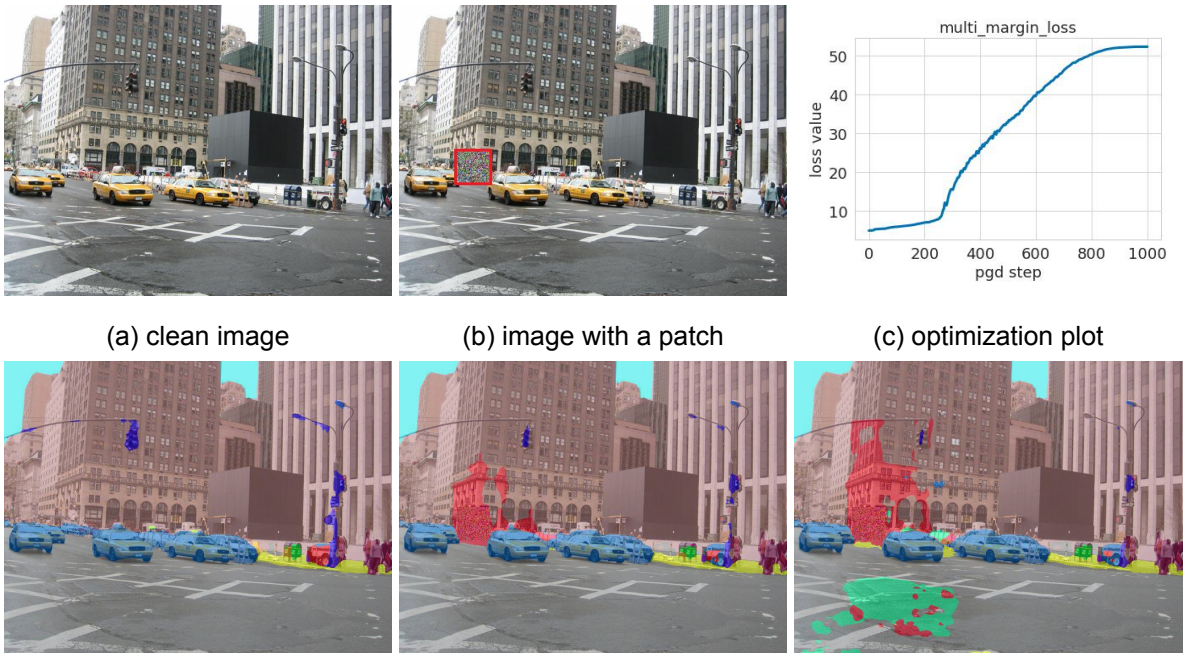


Figure 5.9: Demasked Smoothing recovery column masking illustration for an image from ADE20K (Zhou et al., 2017) with ZITS demasking(Dong et al., 2022) and BEIT-B segmentation (Bao et al., 2022).

5.3.3 Certified detection

We perform our analysis by evaluating the verification map v (Section 5.2.2) for the original image x and for the patched image x'. We see that in $v(x)$ a major part of the building is certified i. e. for a part of pixels $x_{i,j}$ that belong to the building and the road we have $v(x)_{i,j} = 1$. However, $v(x')_{i,j} = 0$ for those pixels. It means that we have detected that the prediction on this input is potentially affected by an adversarial patch.



(a) clean image (b) image with a patch (c) optimization plot

(d) clean segmentation (e) 300 PGD steps (f) 1000 PGD steps

Figure 5.10: Patch attack illustration with Swin (Liu et al., 2021) and an ADE20K image. A patch occupying 1% of the image surface changes the segmentation.

5.4 Evaluation

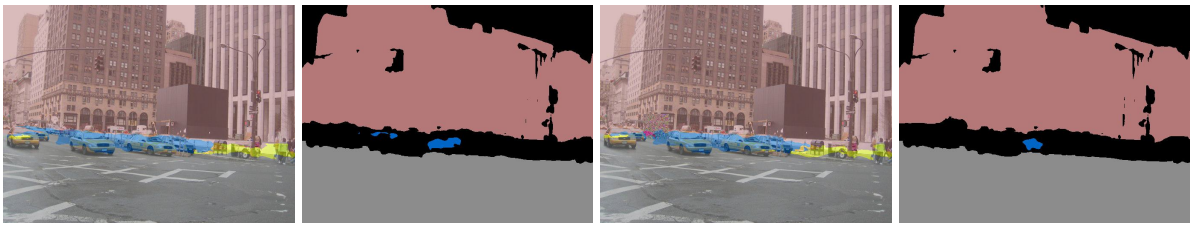
In this section, we evaluate Demasked Smoothing with the masking schemes proposed in Section 5.2. Certified recovery and certified detection provide certificates of different strength (Section 5.2) which are not comparable. We evaluate them separately for different patch sizes.

5.4.1 Experimental Setup

We evaluate Demasked Smoothing on two challenging semantic segmentation datasets: ADE20K (Zhou et al., 2017) (150 classes, 2000 validation images) and COCO-Stuff-10K (Caesar et al., 2018) (171 classes, 1000 validation images).

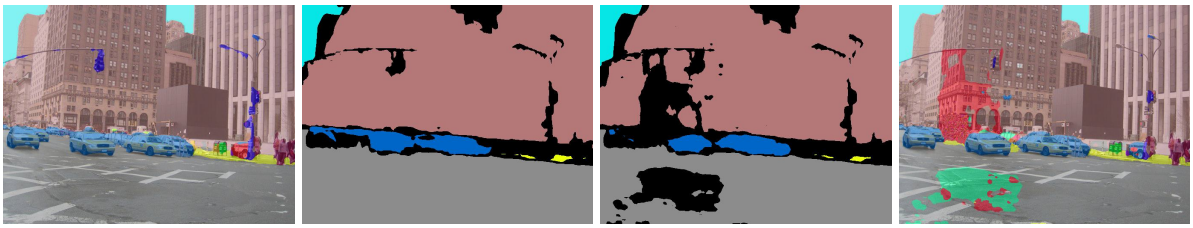
For demasking we use the ZITS Dong et al. (2022) inpainting model with the checkpoint provided in the official paper repository [1]. The model was trained on Places2 (Zhou et al., 2016) dataset with images resized to 256×256. An illustration of the image reconstruction can be found in Figure 5.13. As a segmentation model f we use BEiT-L, BEiT-B Bao et al. (2022), Swin Liu et al. (2021), PSPNet Zhao et al. (2017) with R-101-D8 backbone and DeepLab v3 (Chen et al., 2018). We note that the first two models are based on transformers. PSPNet and DeepLab v3 are CNN-based segmentation methods that we consider to demonstrate that Demasked Smoothing is not

[1] https://github.com/DQiaole/ZITS_inpainting



(a) segmentation of the original image, $h(x)$ (b) certification map of the original image x (c) segmentation of the patched image, $h(x')$ (d) certification map of the patched image x'

Figure 5.11: Certified recovery for a 1 % patch used in the attack. The majority vote function h recovers the prediction in presence of an adversarial patch that fools the undefended model. The segmentation for the original and patched image in (a) and (c) are the same for the regions certified in the certification maps (b) or (d). The certification maps (b) and (d) are also almost the same.



(a) $f(x)$ (original) (b) $v(x)$ (original verified) (c) $v(x')$ (x' verified) (d) $f(x')$ (patched)

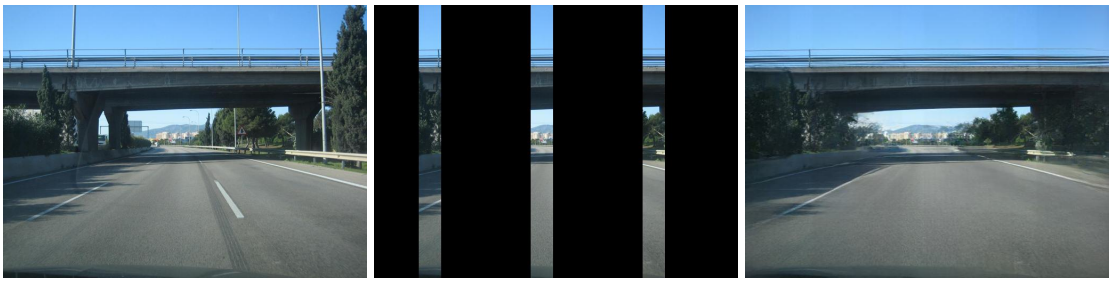
Figure 5.12: f is a segmentation model (Swin Liu et al. (2021)) and v is the verification function (Section 5.2.2). For an attacked image x', $v(x')$ detects the region of $f(x')$ which was (potentially) affected by an adversarial patch.

specific to transformer-based architectures. We use the model implementations provided in the *mmsegmentation* framework Contributors (2020).

5.4.2 Evaluation metrics

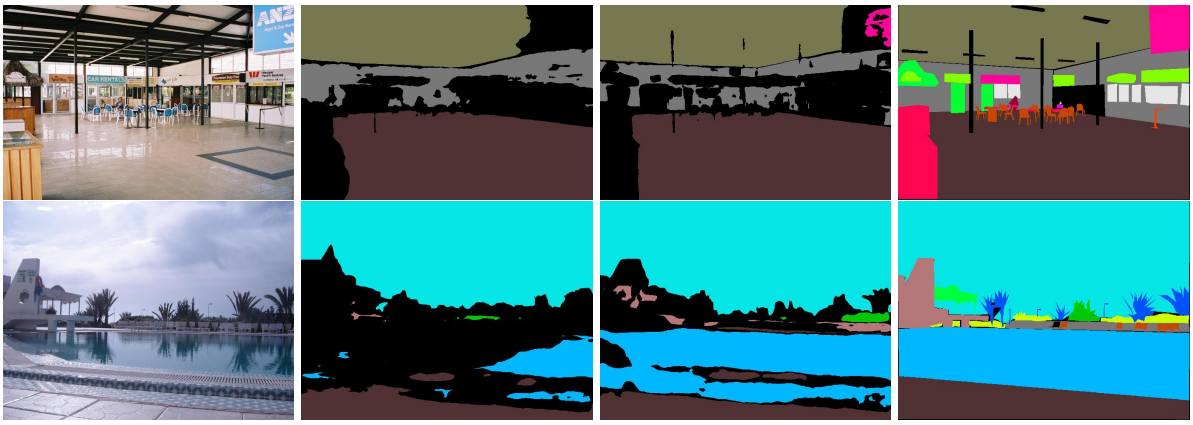
For both certified recovery and certified detection, we generate a standard segmentation output (without any abstention) and a corresponding certification map (Figure 5.14). In case of certified detection, the segmentation output remains the same as for the original segmentation model, however, there may be false alerts in the certificaton map. For the certified recovery, the output is obtained by a majority vote over the segmentations of demasked images (Section 5.2.2) and is different from the original model output. See additional certification map examples in appendix A.

We evaluate the mean intersection over union (mIoU) for these outputs. The certification map



(a) original image (b) masking (c) demasking

Figure 5.13: Reconstructing the masked images with ZITS (Dong et al., 2022)



(a) original image (b) recovery map (c) detection map (d) ground truth

Figure 5.14: Certification map examples on ADE20K Zhou et al. (2017) with ZITS Dong et al. (2022) and Swin Liu et al. (2021). See additional examples in Appendix A.

is obtained by assigning to each certified pixel the corresponding class from the segmentation output and assigning a special *uncertified* label to all non-certified pixels. For each image we evaluate the fraction of pixels which are certified and correct (coincide with the ground truth). %C is a mean of these fractions over all the images in the dataset. In semantic segmentation task, the class frequencies are usually skewed, therefore global pixel-wise accuracy alone is an insufficient metric.

Matching the certification map separately for each class $y \in \mathcal{Y}$ with the ground truth segmentation for y in the image x allows us to compute the guaranteed lower bound ($cTP_y(x)$) on the number of true positive pixel predictions ($TP_y(x)$) i.e. those that were correctly classified into y. If a pixel was certified with a correct class, then this prediction cannot be changed by a patch (or, alternatively, the change will be detected by the verification function v in certified detection). We consider *recall*

$$R_y(x) = \frac{TP_y(x)}{TP_y(x) + FN_y(x)} \tag{5.8}$$

Table 5.2: The list of 19 “big” classes for ADE20K (Zhou et al., 2017) (out of 150 classes in total) with their average fraction of occupied pixels in the images where they are present (%) and index in the list of dataset classes. We define a class to be “big” if it occupies on average more than 20% of the pixels in the images in which this class appears.

#	index	name	fraction	#	index	name	fraction
1	0	wall	25.88	11	79	hovel	25.93
2	1	building	32.36	12	88	booth	23.91
3	2	sky	21.54	13	96	escalator	20.96
4	7	bed	21.25	14	103	ship	26.81
5	21	water	22.10	15	104	fountain	28.81
6	29	field	22.97	16	107	washer	22.07
7	46	sand	21.22	17	109	swimming pool	28.87
8	48	skyscraper	42.92	18	114	tent	34.57
9	54	runway	28.05	19	128	lake	34.57
10	55	case	37.57				

where $FN_y(x)$ is the number of false negative predictions for y in x. $P_y(x) = TP_y(x) + FN_y(x)$ is the total area of y in the ground truth and does not depend on our prediction. We can evaluate *certified recall*

$$cR_y(x) = \frac{cTP_y(x)}{P_y(x)} \tag{5.9}$$

a lower bound on the recall $R_y(x)$. *Total recall* of class y in a dataset D is

$$TR_y(D) = \frac{\sum_{x \in D} TP_y(x)}{\sum_{x \in D} P_y(x)}. \tag{5.10}$$

Certified total recall:

$$cTR_y(D) = \frac{\sum_{x \in D} cTP_y(x)}{\sum_{x \in D} P_y(x)} \tag{5.11}$$

Then, we can obtain *mean recall* over all classes y in the dataset D

$$mR(D) = \frac{1}{|\mathcal{Y}|} \sum_{y \in \mathcal{Y}} TR_y(D) \tag{5.12}$$

and it guaranteed lower bound *certified mean recall*

$$cmR(D) = \frac{1}{|\mathcal{Y}|} \sum_{y \in \mathcal{Y}} cTR_y(D) \tag{5.13}$$

These are the metrics that we provide in our experimental results for certified semantic segmentation. Evaluating lower bounds on other popular metrics such as mean precision or mIoU this

Table 5.3: The list of 21 “big” classes for COCO-Stuff-10K (Caesar et al., 2018) (out of 171 classes in total) with their average fraction of occupied pixels in the images where they are present (%) and index in the list of dataset classes. We define a class to be “big” if it occupies on average more than 20% of the pixels in the images in which this class appears.

#	index	name	fraction	#	index	name	fraction
1	6	bus	21.46	11	105	floor-stone	20.10
2	7	train	23.11	12	111	fruit	20.48
3	20	cow	24.17	13	113	grass	23.25
4	21	elephant	28.50	14	134	playingfield	38.64
5	49	sandwich	23.99	15	137	river	40.01
6	51	broccoli	20.18	16	143	sand	26.37
7	54	pizza	25.86	17	144	sea	36.51
8	60	bed	36.86	18	146	sky-other	22.94
9	61	dining table	21.71	19	148	snow	51.60
10	95	clouds	24.11	20	159	vegetable	20.35
				21	167	water-other	21.67

way results in vacuous upper bound since they depend on the upper bound on false positive (FP) predictions. For the pixels that are not certified we cannot guarantee that they will not be assigned to a certain class, therefore, a non-trivial upper bound on FP is not straightforward. We leave this direction for future work. In certified detection, we additionally consider false alert ratio (FAR) which is the fraction of correctly classified pixels for which we return an alert on a clean image. Smaller FAR is preferable.

Big classes. Due to our threat model, certifying small objects in the scene can be difficult because they can be partially or completely covered by an adversarsial patch in a way that there is not chance to recover the prediction. To provide an additional perspective on our methods, we also evaluate mR and cmR specifically for the “big” classes, which occupy on average more than 20% of the images in which they appear. Correctly segmenting these classes is important for understanding the scene. In Tables 5.2 and 5.3 we provide the full list of such classes in ADE20K (Zhou et al., 2017) and COCO-Stuff-10K (Caesar et al., 2018) respectively together with the average fraction of pixels that they occupy in the images in which they are present. We point out that for COCO-Stuff-10K some typically smaller object classes such as “sandwich” or “fruit” get included in the list of big classes because of the macro-scale images in which they occupy a significant part of the scene.

Table 5.4: The certified detection results (%) for a patch occupying no more than 1% of the image. mIoU - mean intersection over union, mR - mean recall, cmR - certified mean recall (see details on the metrics in Section 5.4.2). %C - mean percentage of certified and correct pixels in the image. FAR - false alert ratio (lower is better).

dataset	segm model	mask	mIoU	big		all		%C	FAR ↓
				mR	cmR	mR	cmR		
ADE20K	BEiT-L	column	56.33	74.26	**61.15**	68.40	**35.88**	**65.44**	**19.51**
		row			52.77		30.25	60.48	24.47
	BEiT-B	column	53.08	70.92	**57.33**	64.45	**32.55**	**63.55**	**20.04**
		row			50.05		26.65	58.34	25.24
	Swin	column	48.35	68.33	**55.91**	59.17	**29.87**	**61.91**	**19.82**
		row			47.56		23.36	57.05	25.33
	PSPNet	column	44.39	61.83	**50.01**	54.74	**26.41**	**60.06**	**19.95**
		row			42.19		19.75	54.03	25.98
COCO10K	PSPNet	column	37.76	71.71	**56.86**	49.65	**26.80**	**47.09**	**21.43**
		row			51.05		23.51	42.78	25.74
	DeepLab v3	column	37.81	72.52	**56.54**	49.98	**26.86**	**47.17**	**21.89**
		row			50.51		23.89	43.19	25.89

5.4.3 Results

Certified detection. The results for certified detection can be found in Table 5.4. mIoU and mR are the clean metrics of the model since the segmentation outcome is not affected in this mode. We observe that in all datasets and models column masking outperforms row masking for all of the considered certification metrics. Thus, we recommend using column masking for Demasked Smoothing certified detection. With BEiT-L model Demasked Smoothing can certify 65.44% of correctly classified pixels, achieve certified mean recall 61.15% for big classes and 35.88% for all classes. We observe %C for all models is not lower than 60% on ADE20K and not lower than 47% on COCO10K.

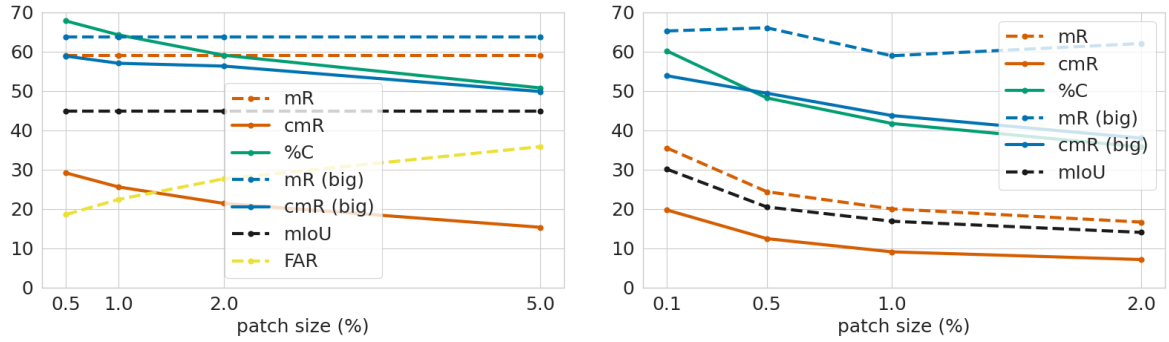
PSPNet and DeepLab v3 that demonstrate close clean performance on COCO10K also demonstrate close performance in certification with Demasked Smoothing. At the same time, using segmentation models with better clean performance allows to achieve better certification results. This suggests that the new state-of-the-art segmentation models proposed in the future can furhter improve Demasked Smoothing certification performance.

Certified recovery. Table 5.5 provides the evaluation results for certified recovery. In this case, the segmentation output is obtained by majority voting and is different from the clean output.

Table 5.5: The certified recovery results(%) against a 0.5% patch. 3-mask and 4-mask correspond to $T = 3$ and $T = 4$ respectively (Figure 5.3). mIoU - mean intersection over union, mR - mean recall, cmR - certified mean recall (see details on the metrics in Section 5.4.2). %C - mean percentage of certified and correct pixels in the image.

dataset	segm	mask	mIoU	big		all		%C
				mR	cmR	mR	cmR	
ADE20K	BEiT-L	column	**28.64**	**71.95**	**50.84**	**34.65**	**16.04**	**47.76**
		row	18.82	53.77	21.24	22.74	5.95	32.30
		3-mask	22.40	64.83	33.96	26.89	8.97	39.59
		4-mask	19.93	60.90	25.03	24.22	6.43	35.01
	BEiT-B	column	**24.92**	**60.77**	**41.26**	**29.84**	**12.98**	**46.22**
		row	16.33	46.91	16.72	19.51	4.83	31.71
		3-mask	19.90	56.90	26.51	23.86	7.54	38.64
		4-mask	18.82	52.96	23.75	22.56	5.87	34.36
	Swin	column	**22.43**	**59.75**	**34.88**	**27.09**	**11.70**	**46.14**
		row	13.58	42.88	15.13	16.70	4.46	30.64
		3-mask	17.06	51.03	24.15	20.74	6.65	38.27
		4-mask	14.77	46.67	17.74	10.05	4.72	34.04
	PSPNet	column	**19.17**	**51.90**	**34.11**	**23.66**	**10.76**	**44.90**
		row	12.00	36.26	12.03	15.03	3.74	28.29
		3-mask	15.00	44.93	19.55	18.41	5.58	35.85
		4-mask	12.74	40.41	15.86	15.87	4.14	31.22
COCO10K	PSPNet	col	**21.94**	**61.56**	**36.67**	**29.94**	**11.13**	**29.51**
		row	18.87	58.04	20.90	26.16	6.14	19.31
		3-mask	18.82	59.26	29.00	25.85	7.56	25.21
		4-mask	17.46	58.47	23.63	24.35	5.51	20.36
	DeepLab v3	col	**23.12**	**62.60**	**33.84**	**31.59**	**11.55**	**28.71**
		row	20.04	55.71	17.80	27.89	6.28	17.04
		3-mask	20.14	58.02	27.14	27.82	8.05	24.30
		4-mask	19.35	58.22	22.01	26.74	5.79	19.38

Thus we provide mIoU and mR for each masking scheme. These values can be compared to the clean values in Table 5.4. Here, we observe that in all datasets and models column masking outperforms all other masking schemes by a large margin in both clean and certified metrics which is consistent with the certified detection results. Among other maskings, 3-mask consistently performs the best followed by 4-mask. Row masking performs the worst in all cases even though each row mask leaves approximately the same number of pixels visible in each mask as column masking. We attribute the effectiveness of column masking to the fact most of the images in the considered datasets have a distinct horizon line. Therefore having a visible column provides a slice of the image that intersects most of the scene background objects.



(a) certified detection

(b) certified recovery

Figure 5.15: Performance for different adversarial patch sizes evaluated on 200 ADE20K images for (a) certified detection and (b) certified recovery.

Once again, we observe the best results with BeiT-L. mIoU is 28.64% which is two times lower than the clean mIoU of 56.33%. A roughly similar ratio is observed in other cases. Mean recall for big classes of 71.95% is surprisingly close to the clean value of 74.26% for BEiT-L with the gap being slightly bigger for other models. It suggests that majority voting based segmentation favors big classes and does not cause a significant recall drop on them. This difference is bigger for the recall on all classes.

5.4.4 Effect of the maximal patch size

Figure 5.15 shows how the performance of Demasked Smoothing depends on the patch size. We see that certified detection metrics remain high even for a patch as big as 5% of the image surface. False alert ratio grows from about 20% with a 0.5% patch to about 35% for a 5% patch. For the recovery mR for big objects remains high for different patch sizes while other metrics slowly deteriorate as we increase the patch size to 2%.

5.5 Comparison to simplified Derandomized Smoothing

Derandomized Smoothing (DRS) Levine and Feizi (2020) was proposed for certified recovery. Therefore, in this section we focus on this task. Direct adaptation of derandomized smoothing to semantic segmentation task requires training a model that is able to predict the full image segmentation from a small visible region. Since it is not immediately clear to us what architectural design and training procedure would be needed to train such a model, we consider a simplified version of DRS that we call DRS-S. In this version, we consider an off-the-shelf semantic segmentation model and evaluate how it performs with column masking from DRS. Therefore, we

Table 5.6: Comparison of our method with simplified Derandomized Smoothing Levine and Feizi (2020). We consider column masking. mIoU - mean intersection over union, mR - mean recall, cmR - certified mean recall. %C - mean percentage of certified and correct pixels in the image. We use Swin model on 200 ADE20K images.

method	mIoU	big		all		%C
		mR	cmR	mR	cmR	
Demasked (our)	**19.09**	**66.03**	**52.71**	**23.02**	**12.66**	**47.05**
DRS-S	0.42	11.35	9.08	1.04	0.83	28.01
DRS-E	9.12	54.67	41.78	11.04	7.86	45.03

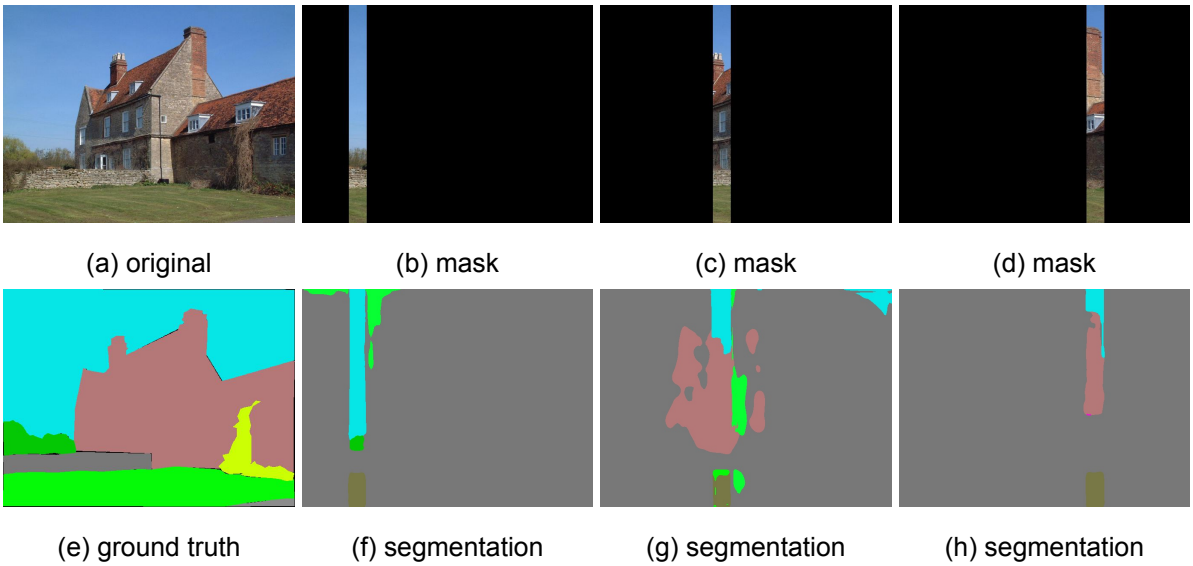
do not encode the masked regions with the special 'NULL' value like in DRS but use black color instead. That is because an off-the-shelf model cannot work with 'NULL' values.

We run our experiments on ADE20K dataset. We consider the DRS parameters from the recent SOTA version of Derandomized Smoothing by Salman et al. Salman et al. (2021). They use column width $b = 19$ and stride $s = 10$ for certified classification of 224x224 ImageNet images. To account for the fact that ADE20K images have larger resolution than ImageNet, we scale the parameters to column width $b = 42$ and stride $s = 22$. To make the comparison consistent with the rest of our results, we use the patch occupying 0.5% of the image.

From Table 5.6 we can see that DRS-S performs poorly on semantic segmentation task. The reason for that is illustrated in Figure 5.16. Processing the column region in 5.16c would probably be sufficient for a classification model to classify the image into the class "house". But it is clearly not sufficient to reconstruct the whole segmentation map 5.16e as can be seen in the Figure 5.16g. Whether doing this would be possible with a model specifically trained to reconstruct the segmentation map from a very small visible region is an open research question (up to our knowledge).

We point out that the value %C of certified and correctly classified pixels in the Table 5.6 is still surprisingly high for DRS-S compared to other metrics. We attribute this to the fact that the solid black regions are usually treated as a wall by the segmentation model, therefore the images are usually segmented as a wall by the DRS majority voting. And the wall is a common part of both indoor and outdoor scenes in ADE20K as can be implied from the Table 5.2 of "big" ADE20K classes. Therefore, always classifying the output as a wall provides a decent fraction of correctly classified pixels because of the skewed classes.

However, to provide a better comparison with DRS, we emulate the model which is able to



(a) original (b) mask (c) mask (d) mask

(e) ground truth (f) segmentation (g) segmentation (h) segmentation

Figure 5.16: The illustration for the poor performance of Derandomized Smoothing column masking in a dense prediction task.

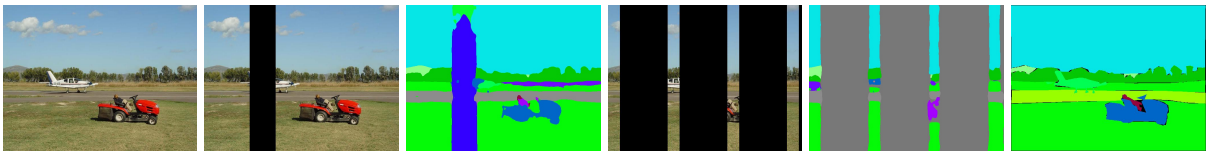
reconstruct the whole segmentation map from the column masking proposed in DRS. We do this by applying the demasking approach proposed in this work. We first try to reconstruct the whole image from one column and then segment it with an off-the-shelf model as we did with the masks proposed in this paper. We call this approach DRS-E and the results can be found in Table 5.6.

In this work, we consider deterministic certified defences. Randomized Cropping (Lin et al., 2021) is a non-deterministic defence i. e. the certification results only hold for a given confidence interval. Thus, a direct comparison with our work is infeasible.

5.6 Demasking ablation studies

In this section, we perform ablation studies with respect to the demasking step (Figure 5.2). First, we consider masking without demasking i. e. processing the images with the masked regions simply inpainted with some solid color. Then, we compare the ZITS transformer-based inpainting mode used in our evaluation (Section 5.4) to a CNN Fourier transform-based method called LAMA. We demonstrate that the results are still good if we substitute the inpainter with a different one demonstrating stability of the Demasked Smoothing. Finally, we do ablation with respect to the training set on which the inpainter was trained.

Solid color inpainting. We see that inpainting the masked regions with solid color provides poor results (Table 5.7). Figure 5.17 demonstrates that the regions are considered to be objects



(a) original (b) detection col (c) segmented (d) recovery col (e) segmented (f) ground truth

Figure 5.17: Results without image demasking. The solid color inpainting is treated as a separate object in the scene because we need to classify every pixel in semantic segmentation task. Therefore, it is hard to achieve a situation where all the demasked segmentation agree on some pixel which is represented in the Table 5.7.

Table 5.7: Comparison for demasked smoothing with and without demasking step. mIoU - mean intersection over union, mR - mean recall, cmR - certified mean recall. %C - mean percentage of certified and correct pixels in the image. We use Swin model on 200 ADE20K images with column masking for certified detection and certified recovery. We compare masking the columns with solid black color without demasking to ZITS demasking.

mode	patch size	demasking	mIoU	big		all		%C
				mR	cmR	mR	cmR	
detection	1.0%	✓	38.56	67.25	**58.85**	53.37	**23.35**	**62.89**
		✗			19.49		3.09	21.19
recovery	0.5%	✓	**19.09**	**66.03**	**52.71**	**23.02**	**12.66**	**47.05**
		✗	1.10	15.09	7.71	1.79	0.72	18.59

in the scene. Thus, inpainting step is a crucial step of our procedure that cannot be neglected.

Comparing different inpainting models. We have considered Demasked Smoothing with the LAMA (Suvorov et al., 2022) inpainting approach based on Fast Fourier convolutional neural network. We have compared it to using ZITS (Dong et al., 2022) method based on incremental transformer structure. ZITS is reported to outperform LAMA on different inpainting metrics (Dong et al., 2022).

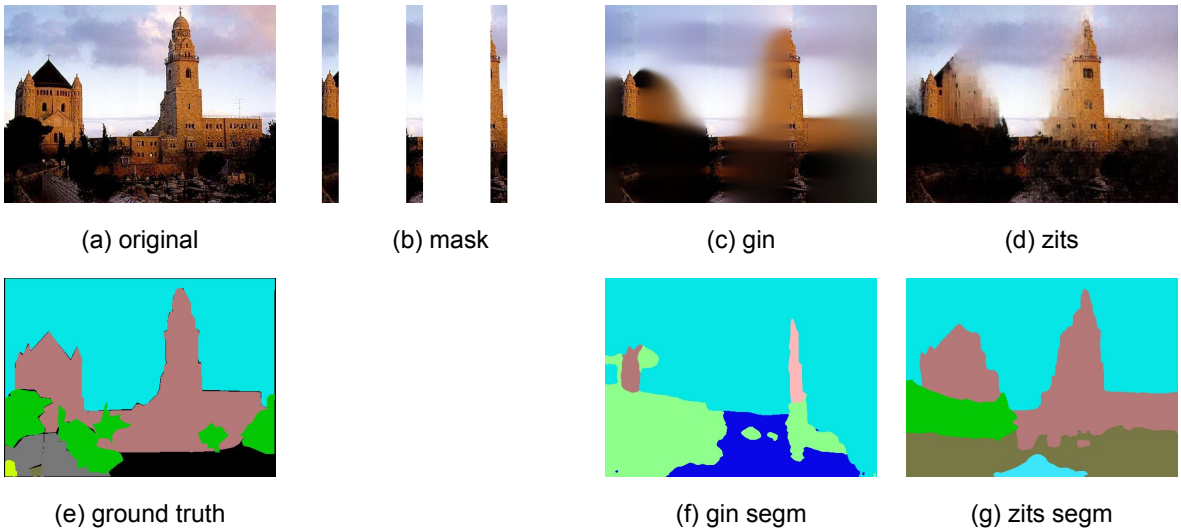
In the Table 5.8, we provide a detailed comparison with respect to all certification modes and masking schemes proposed in our work for the ADE20K validation set and BEiT-B segmentation model. We observe that using a different well-performing inpainting method has a marginal effect on the results of Demasked Smoothing certification demonstrating the reliability of the method. At the same time using a stronger inpainting models allows to achieve better clean and certified accuracy. We consider this property to be a strength of our method since it will automatically benefit from future research and developments of stronger inpainting methods.

Table 5.8: Comparison of different inpainting models: LAMA Suvorov et al. (2022) and ZITS Dong et al. (2022). mIoU - mean intersection over union, mR - mean recall, cmR - certified mean recall. %C - mean percentage of certified and correct pixels in the image. For detection, we provide clean mIoU since the output is unaffected and mean false alert rate (FAR) (lower is better).

mode	mask	demasker	mIoU	big		all		%C	FAR ↓
				mR	cmR	mR	cmR		
detection 1% patch	column	ZITS	53.08	70.92	**57.33**	64.45	**32.55**	**63.55**	**20.04**
		LAMA			56.99		31.67	64.21	19.37
	row	ZITS	53.08	70.92	50.05	64.45	26.65	58.34	25.24
		LAMA			49.06		26.58	59.21	24.38
recovery 0.5% patch	column	ZITS	**24.92**	**60.77**	**41.26**	**29.84**	**12.98**	**46.22**	N/A
		LAMA	22.48	58.20	37.51	26.49	11.49	45.95	
	row	ZITS	16.33	46.91	16.72	19.51	4.83	31.71	
		LAMA	15.64	43.07	16.51	18.78	4.95	32.84	
	3-mask	ZITS	19.90	56.90	26.51	23.86	7.54	38.64	
		LAMA	18.54	53.59	27.39	22.12	7.58	39.52	
	4-mask	ZITS	18.82	52.96	23.75	22.56	5.87	34.36	
		LAMA	17.00	50.60	18.18	20.22	5.23	35.98	

Using a model trained on a different dataset. As stated in Section 5.4, we use ZITS model trained on Places 2 dataset. Then we apply it to segmenting images from ADE20K dataset. To provide an additional perspective on our methods, we consider an inpainting model trained on ADE20K. We consider GIN method Li et al. (2020a) based on a generative model that we train on ADE20K for 100 epochs without using style losses based on ImageNet trained VGG. We note that this model was proposed several years ago and is impaired by the requirement to rely only on ADE20K dataset. Therefore, the inpainting results are not state-of-the-art. In Figure 5.18, we demonstrate that the inpainting for the certified recovery masking with GIN is less quality than for ZITS. It results in a subpar downstream segmentation which negatively affects the final results.

The experimental results in Table 5.9 support this observation and demonstrate that having a state-of-the-art inpainting model such as ZITS allows to have a significantly better performance. At the same time Demasked Smoothing obtains non-trivial certification results with GIN model. Although we believe that these results can be further improved by adjusting the training procedure of the inpainting model, in this work we focus on methods that require no specific pre-training. Nevertheless, we note that the results that we provide in Table 5.9 can be directly used



(a) original (b) mask (c) gin (d) zits

(e) ground truth (f) gin segm (g) zits segm

Figure 5.18: Comparison between GIN and ZITS inpainting

Table 5.9: Comparison of our two demasking methods: ZITS and GIN. mIoU - mean intersection over union, mR - mean recall, cmR - certified mean recall. %C - mean percentage of certified and correct pixels in the image. We use Swin model on 200 ADE20K images.

demasking	trained on	mIoU	big		all		%C
			mR	cmR	mR	cmR	
ZITS (Dong et al., 2022)	Places2	**19.09**	**66.03**	**52.71**	**23.02**	**12.66**	**47.05**
GIN (Li et al., 2020a)	ADE20K	5.46	32.27	19.05	7.62	3.52	32.08

as a benchmark in the orthogonal line of research dedicated to training the models specifically for certified robustness of semantic segmentation.

5.7 Complexity analysis and execution time

Complexity analysis. In Demasked Smoothing, we propose a set of K masks that are applied to the original image (denote the cost of applying a single masking by M). As illustrated in Figure 1.5, the masked images are demasked (denote the cost of demasking an image by D) and segmented (denote the cost of segmenting an image by S); thereupon per-mask segmentations are aggregated into a final segmentation and certification (cost of aggregation proportional to K). Asymptotically, compute grows thus with $O(K(M+D+S)+K)$ while the cost of a standard segmentation is $O(S)$. Thus, for large K or $M+D \gg S$, real-time applicability would actually be impractical. However, we note that:

1. $M+D$ is roughly of the same size as S for typical DL-based inpainting and segmentation

models.

2. For certified recovery, we operate in a setting where K is small ($K \in 5, 7, 9$) and does not grow with the image resolution. This is unlike Derandomized Smoothing and its derivatives, where the number of masks in the recovery task grows with the image resolution (or randomized smoothing with thousands of samples per input). This small value of K benefits our the method in time-sensitive applications. For certified detection, we can adjust the number of masks for the computational speed by using strided masking as suggested in Section 5.2.1.

3. Moreover, masking, demasking, and segmenting for different masks do not use any shared data and can thus be fully parallelized if sufficiently powerful hardware is available. Only the aggregation step requires the results of all the previous stages. However, aggregation time is small compared to the other stages. In terms of latency, a fully parallelized version of our procedure would thus have a latency proportional to $O(M + D + S + K)$. For small K and $M + D \approx S$, application to real-time video can be facilitated by means of parallelization.

Execution time. In Table 5.10, we provide the execution time measurements with ZITS inpainting model (Dong et al., 2022) for the BEiT-B segmentation model (Bao et al., 2022) on 2000 ADE20K images. We consider both recovery and detection with all of the considered masking schemes. We observe the fastest execution time for certified recovery with row and column masking having only $\times$17 and $\times$18 overhead with respect to the standard propagation time. Certified detection with columns and rows has $\times$47 and $\times$48 overhead due to a larger number of masks than in certified detection.

5.8 Test-time input certification

In this section, we discuss how certified recovery (Theorem 1) can be applied to guaranteed verification of the robustness on a test image. We also discuss how robustness guarantees for the test-time images can be evaluated by using a dataset of clean images such as ADE20K Zhou et al. (2017) or COCO-Stuff-10K Caesar et al. (2018).

5.8.1 Test-time certified recovery

Let x' be a test-time input which can be either a clean image or an image attacked with an adversarial patch. We know that there exists a clean image x corresponding to x' which removes the patch if it is present. We have either $x' = x$ or $x' \in A(x)$, where $A(x) := \{A(x, p, l) \mid (p, l) \in$

Table 5.10: Execution time comparison for the BEiT-B model on 2000 ADE20K images. Vanilla segmentation average per-image run time is 387 miliseconds. Computations were done on a single Nvidia Tesla V100 GPU. We provide total Demasked Smoothing time as well as the execution time of different stages of the method. We use $K = 20$ masks for detection and $K = 5, 7, 9$ masks for recovery with $T = 2, 3, 4$ respectively.

mode	mask type	average per-image run time (ms) mask	demask	segment	aggregate	total	overhead
detection	column	428	9125	8429	352	18334	×47
1% patch	row	251	9485	8503	348	18587	×48
	column	376	3439	2446	563	6824	×18
recovery	row	194	3329	2460	645	6628	×17
0.5% patch	3-mask	194	4370	3319	750	9025	×22
	4-mask	243	5020	4063	825	10151	×26

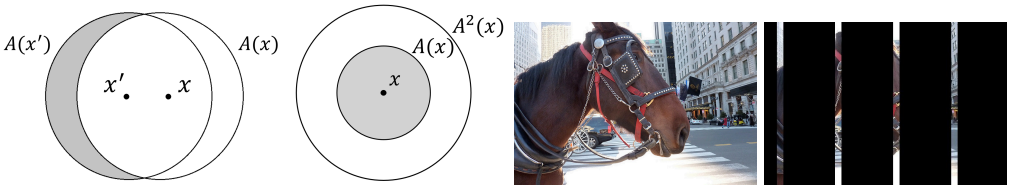
$\mathcal{P}\}$. However, at test time we do not have access to the clean image x.

Our goal is to certify that for our segmentation model h and a pixel $x_{i,j}$ we have $h(x')_{i,j} = h(x)_{i,j}$. We can achieve this result by applying the recovery certification (Theorem 1) to the test-time image. It allows us to verify whether

$$\forall\ (p,\ l) \in \mathcal{P}\ :\ h(A(x', p, l))_{i,j} = h(x')_{i,j}. \tag{5.14}$$

We also know that if $x' \in A(x)$, then $x \in A(x')$ (Figure 5.19a). Indeed, if x' is only different from x by one patch, then x can be be obtained from x' by removing this patch. Therefore, by obtaining the guarantee for $A(x')$, we implicitly obtain the guarantee also for the image x even though we do not have direct access to it. The scenario of the real-world robustness verification where the defender doesn't have access to the clean version of the image x but only to the attacked version x' was discussed by Hong and Hong (2023) for the Randomized Smoothing (Cohen et al., 2019) and ℓ_p threat model. Note that their work was published after Demasked Smoothing.

We note that this test-time guarantee is only possible for certified recovery. In certified detection, we would need to evaluate the verification function v (Theorem 2) for both the clean image x and the attacked image x' to obtain the result. This cannot be done if x is implicit.



(a) certified inference (b) double adversarial neighbourhood (c) original image (d) column mask for certified recovery against two patches

Figure 5.19: Test-time input certification scheme and double-patch certification mask

5.8.2 Robustness guarantees evaluation

The typical certified robust error for a given test data set (and pixel (i, j) in the segmentation case) is an estimate for

$$\mathop{\mathbb{E}}_{X\sim D}[\max_{(p,l)\in P} \mathbb{1}_{h(A(X,p,l))_{i,j}\neq h(X)_{i,j}}], \tag{5.15}$$

where D is the data generating probability measure and we assume that our test set to be an i.i.d. sample of it. This is the expected robust error (worst case over our threat model P for clean inputs) for a given pixel (i, j). Using the test sample to get an estimate of this quantity, we get a probabilistic guarantee that the corresponding pixel (i, j) of a new *clean* test sample x' drawn i.i.d. from P will have its whole "patch"-neighborhood certified.

However, more important for a practical security analysis is that we can certify a given instance, which can be even potentially adversarially perturbed. Formally, this means that for an input $z \in A(x)$, where $x \sim P$ is an unknown sample from P, that we guarantee

$$\forall (p, l) \in P : h(A(z, p, l))_{i,j} = h(z)_{i,j}, \tag{5.16}$$

and as $x \in A(z, p, l)$ this implies that we certify that the pixel (i, j) of the potentially manipulated image is classified the same as pixel (i, j) of the unperturbed image x.

However, it is now tricky to get even a probabilistic estimate of the quantity

$$\mathop{\mathbb{E}}_{x\sim D} \max_{(p,l)\in P}[\max_{(q,m)\in P} \mathbb{1}_{h(A(A(x,p,l),q,m))_{i,j}=h(A(x,p,l))_{i,j}}], \tag{5.17}$$

as the outer maximization process cannot be simply simulated by doing adversarial patch attacks on a clean test dataset.

Table 5.11: Inference recovery robustness estimate. To illustrate our point we certify an example for a 0.1% patch.

dataset	segm	mask	mIoU	big		all		%C
				mR	cmR	mR	cmR	
ADE20K	BEiT-B	col	19.97	55.27	32.65	24.38	9.43	39.50
COCO10K	PSPNet		19.28	59.40	26.47	26.96	7.60	21.51

We propose a way to evaluate a guaranteed lower bound on the fraction of certified test-time inputs by using a dataset of clean images. Instead of considering a standard one-patch neighbourhood $A(x)$ defined by our threat model (Section 5.1), we propose to consider a neighbourhood $A^2(x)$ of two independent patches (Figure 5.19b). $A^2(x)$ contains all the images $x' \in A(x)$ as well as their respective patch neighbourhoods $A(x')$. Therefore, by verifying that

$$\forall\ (p_1,\ l_1), (p_2,\ l_2) \in \mathcal{P}\ :\ h(A(A(x,\ p_1,\ l_1),\ p_2,\ l_2))_{i,j} = h(x)_{i,j}, \tag{5.18}$$

we guarantee that

$$\forall\ x' \in A(x)\ \forall\ (p,\ l) \in \mathcal{P}\ :\ h(A(x',\ p,\ l))_{i,j} = h(x')_{i,j}. \tag{5.19}$$

We note that corresponding reasoning could be applied to certification in ℓ_p models. Then $A^2(x)$ would correspond to doubling the radius of the ϵ-ball instead of adding a second patch.

Note that Theorem 1 can be directly extended to a threat model of N patches. In the worst case each of the N patches can affect T different maskings. Therefore, we need to change the condition of Theorem 1 to $K \geq 2NT + 1$. We apply the described method to evaluating the test-time certification guarantees for a toy example of a 0.1% patch in Table 5.11. We also illustrate how a column mask looks in this case in Figure 5.19.

5.9 Limitations

The performance of Demasked Smoothing certified recovery may be insufficient for the downstream task if we certify against big patches (Figure 5.15) unless robustness is prioritized over clean performance. We point out that robustly segmenting small objects is fundamentally difficult under the adversarial patch threat model since the objects themselves can be completely or partially covered by an adversarial patch which makes it impossible to properly segment them even for a human being. Demasked Smoothing certification requires an upper bound on the

size of the expected patch (Section 5.2).

5.10 Conclusion

In this chapter, we propose Demasked Smoothing, the first certified defence framework against patch attacks on segmentation models. Due to its novel design based on masking schemes and image demasking, Demasked Smoothing is compatible with any segmentation model and can on average certify 65% of the pixel predictions for a 1% patch in the detection task and 47% against a 0.5% patch for the recovery task on the ADE20K dataset with BEiT-L segmentation model.

Chapter 6

Promising Directions and Open Problems

In this chapter, we summarize interesting directions and important unsolved problems in the field of deep learning robustness. In particular, we emphasize two broad and dynamic fields that were the focus of the research performed in this thesis: black-box adversarial attacks and adversarial patches. We consider these topics to be paramount for the further advances of robust deep learning and its application in the safety-critical tasks (see Chapter 2 for details).

6.1 Black-box adversarial attacks

As discussed in Section 2.2.5, black-box adversarial attacks proposed up to date can be broadly categorized into transfer-based, query-based and combined approaches by the type of the information that they use to craft the adversarial perturbation. The current trends in the literature clearly show that combined methods attract the most interest in the field since different ways to combine the priors obtained from the surrogate models with the query-based communication with the target model allow to take the best from the both worlds (Yatsura et al., 2021; Lord et al., 2022; Cai et al., 2023).

In particular, we find the further extension of the approach proposed in BASES (Cai et al., 2023) to be promising. The authors optimize the weighting of different surrogate models in the ensemble during the attack using the queries to the target model. We think that adapting the ensemble on the fly is a reasonable approach given the abundance of different architectures and models proposed in the past years (He et al., 2016c; Dosovitskiy et al., 2021). The authors acknowledge that the efficiency of the attack still relies a lot on the ensemble diversity as it

always is in the transfer-based attacks (Huang and Zhang, 2020). Thus, when increasing the ensemble size, more sophisticated methods need to be used to optimize the weights of the different models. In particular, one can consider selecting a better initialization based on the prior knowledge or adapting the weights for all the models simultaneously instead of doing it for one model at a time.

Pure query-based approaches may be of interest as well when using an ensemble of surrogate models is not feasible. For this setting we find it interesting to identify the connection between gradient-based (Cheng et al., 2019), heuristic (Guo et al., 2019; Andriushchenko et al., 2020) and meta-learned (Yatsura et al., 2021) perturbation update mechanisms in order to combine their strengths.

One interesting direction is defending the models specifically against black-box attacks. For example, Chen et al. (2022) propose to slightly modify the output logits in order to confuse the score-based attacks and prevent them from finding adversarial examples. This is an empirical defence that can potentially be countered by an adaptive attack (Tramer et al., 2020). However, work in this direction can help us better understand defending the models against black-box attacks and, possibly, even obtain provable robustness guarantees. We propose to consider a novel line of research, namely, defending the models against transfer-based attacks. One way to do this could be applying adversarial training (Madry et al., 2018b) with adversarial examples generated for the models from a diverse ensemble. A model trained this way should be especially robust against transfer-attacks since it should be able to resist the adversarial examples transferred from potential surrogate models.

Researching novel black-box settings might be a promising direction as well. Sun et al. (2022) have recently proposed lightweight black-box attacks to operate in a scenario where an attacker only has access to very limited model information e. g. one correctly classified sample per classification category. We find it promising to investigate the transfer of adversarial examples between models having not only different architecture and weights but also operating on different datasets. In Chapter 3 we have shown how our Meta Square Attack can transfer the attack knowledge between very different datasets e. g. CIFAR10 and ImageNet having different label sets. In particular the data-free setting (Wang et al., 2022; Nayak et al., 2022) in which the attacker doesn't know even the exact number of classification categories would be of interest. Other novel threat models could consider the gray-box attacks with some other available model information such as architecture or optimization procedure but unknown weights.

Another trend in the literature shows growing interest in applying approaches inspired by meta-

learning to improving black-box adversarial attacks (Du et al., 2020; Yuan et al., 2021; Yatsura et al., 2021; Fu et al., 2022; Yin et al., 2022). Since both fields are rapidly growing, we think that further interesting results can be found in their intersection. We see numerous opportunities for extending the meta-learning framework proposed in this work (Chapter 3). Possible future directions involve extending the number of parameters controlled by the learned optimizer (such as position and shape of the update or initialization in Square Attack) as well as evaluating our adversarial attack optimization framework on other white-box and black-box attacks such as SimBA (Guo et al., 2019) (see details in Section 3.3). Besides that, we find it promising to consider other meta-learning strategies, in particular, black-box optimization to allow using the source model with black-box access. Further simplification of the controller architecture would also be desirable.

In Section 2.2.5 we note that most of the black-box robustness research is focused on the classification task. Nevertheless, there is a lot of promise and challenges in applying this type of attacks to other computer vision problems such as object detection (Cai et al., 2022b), semantic segmentation (Gu et al., 2021) or video classification (Li et al., 2021). It was already shown that extending black-box attacks to the dense prediction tasks is not straightforward and requires significant amount of novel design. For example, Cai et al. (2023) show that the losses, optimized during the training of the dense prediction models, are more complex than the ones applied in the classification setting and need to be handled carefully. At the same time, Liang et al. (2022a) show that the attack objective for the object detection needs to account for the problem specifics. We see a lot of promise in enhancing context-awareness (Cai et al., 2022b) of black-box attacks on the dense prediction tasks as well as making it more time and memory efficient. We would also suggest further research of black-box attacks with provable success guarantees (Hong and Hong, 2023), in particular for other computer vision tasks than image classification.

A general trend shows more interest in making black-box attacks closer to the real-world scenarios. Following the tremendous success of the methods using relatively rich model output such as confidence for all classes (Andriushchenko et al., 2020), there are a lot of unanswered questions on how well black-box attacks perform in more realistic scenarios and how this performance can be further improved. In particular, there is growing interest in decision-based attacks (Brendel et al., 2018; Huang et al., 2022; Vo et al., 2022) using only the predicted labels. Besides, there are some interesting recent works on applying black-box setting for the adversarial patch optimization (Wei et al., 2022; Lapid and Sipper, 2023). For details on adversarial patch see Section 2.2.8.

Considering black-box attacks with other non-ℓ_p threat models such as unrestricted color attacks might be promising as well (Yuan et al., 2022). Another exciting direction recently proposed in the literature is identifying systematic errors and bugs in the model predictions (Metzen et al., 2023; Wiles et al., 2023). Since these methods typically use only querying the model output with the inputs generated from the textual prompts, it would be interesting to look at them from the black-box attack perspective and study how the number of queries required to identify systematic errors of the black-box models can be decreased. Alternatively, one could study the transferability of such systematic errors and consider transfer-based or combined approaches to this task.

Adversarial attacks can be researched not only for perception tasks such as object detection, image classification or segmentation but for other problems such as generative ones. For example, for text-to-image generative models that became popular recently such as diffusion models (Sohl-Dickstein et al., 2015b; Rombach et al., 2022). One could formalize and study perturbations in textual prompt, in particular, in black-box setting to better understand the robustness of such approaches (Zhuang et al., 2023; Maus et al., 2023).

Another promising direction for adversarial attacks going beyond vision is studying the robustness of multi-modal models such as CLIP (Radford et al., 2021). Zou et al. (2023) have considered adversarial attacks on aligned language model. Novel adversarial attacks on large language models would be valuable for the community to emphasize their vulnerability in presence of malicious attacker.

6.2 Adversarial patches

We consider exploring new masking schemes (Xiang and Mittal, 2021a; Balasubramanian and Feizi, 2022) for the existing smoothing-based patch certification approaches to be an important open problem. In particular, for the dense prediction tasks where the input information available after masking plays the key role (Yatsura et al., 2023). Learning the masking schemes in a data-driven manner instead of hand-crafting them might be a promising direction.

Certified patch defences were proposed for various computer vision tasks such as classification, object detection or semantic segmentation (Wei et al., 2023; Xiang et al., 2023). However, there are still many important unsolved problems. For example, obtaining provable guarantees for the tasks with the temporal component such as video classification or object tracking as well as regression-based problems such as depth estimation or key points prediction. Using inpainting based methods for this can be a promising direction (Yatsura et al., 2023), however

the aggregation and certification steps (Figure 5.2) require novel design adapted for these tasks and the time efficiency needs to be further improved. Future work may also include extending the number of metrics that can be certified for the dense prediction tasks for example mIoU or mean class precision for semantic segmentation. As outlined by Xiang et al. (2023), generalization of the provable patch robustness to complex end-to-end AI systems would be a promising direction since it would allow practical certification for autonomous driving.

The patch threat model (Brown et al., 2017) should be studied further as well, in particular the case of several disconnected patches. It was shown that although existing certified defences based on the reduced field of view can provide guarantees against several patches at once (Xiang et al., 2022a), they often suffer significant reduction in the certification efficiency or performance speed, especially in the dense prediction tasks (Xiang et al., 2023). For example, see Table 5.11 where we certify semantic segmentation against two tiny patches (0.1% of the image surface) and already see suboptimal certification performance. We find both multi-patch attack and multi-patch defences to be promising directions.

On the attack side, one could consider the ”sleeping patch” scenario in which an adversarial patch located somewhere in the scene is ”hiding” i. e. not affecting the model performance on its own but leverages the full multi-patch performance once another patch appears in the scene. Other synergy mechanisms between different patches would be of interest as well. Another understudied line of research is optimizing the patch position and content simultaneously (Wei et al., 2022). Since doing this directly would be infeasible, one would require novel approaches for the position-content optimization. As was already mentioned above, exploring adversarial patch optimization in the black-box setting has extreme practical importance (Lapid and Sipper, 2023). Since adversarial patches pose a significant threat for autonomously moving agents e. g. robots in robotics or vehicles in autonomous driving (Ranjan et al., 2019b), it would be interesting to consider the temporal modification of the patches e. g. the patches getting bigger as the agent approaches them. Existing methods such as Expectation over Transformation (Athalye et al., 2018b) might be too general for this specific task and not take the temporal dependencies into account.

Chapter 7

Conclusion

In this work, we have presented several novel approaches for evaluating and improving the robustness of deep learning models. We have summarized the advancements in the quickly evolving field of deep learning robust to physically-realizable attacks and emphasized the utmost importance of this research field.

We show that meta-learning the search distribution of black-box random search based adversarial attacks allows to remove the significant amount of manual design in black-box robustness evaluation and find adversarial examples more efficiently for different query budgets (Yatsura et al., 2021). We implement and investigate this method for Square Attack with ℓ_∞ and ℓ_2 perturbations. Our experimental results show that learned adaptive controllers improve attack performance across different query budgets and generalize to new datasets as well as targeted attacks. This work was accepted at the Thirty-fifth Conference on Neural Information Processing Systems, NeurIPS 2021.

In this work, we propose BagCert (Metzen and Yatsura, 2021) which is a novel approach to certifying deep learning robustness against patches based on a specific architecture and end-to-end model training. It allows to reduce the amount of manually-chosen parameters of the model and improve the robustness evaluation efficiency. The main contributions are a model architecture based on a CNN with small receptive field, certification conditions that are applicable to a broad range of models, and a margin-loss based objective that is derived from the certification condition. The resulting model achieves high certified robustness against patches with a broad range of sizes, aspect ratios, and locations on CIFAR10 and ImageNet. This work was accepted at the Ninth International Conference on Learning Representations, ICLR 2021.

We propose Demasked Smoothing (Yatsura et al., 2023) as the first approach to certify robust-

ness against patch attacks for an important semantic segmentation task. We have introduced specific masking schemes tailored to the dense prediction task as well as adding image inpainting into the patch robustness certification pipeline. This work was accepted at the Eleventh International Conference on Learning Representations, ICLR 2023.

We see numerous exciting possibilities in the field of adversarial robustness such as further automation of robustness evaluation and adversarial defences and finding new connections between model robustness and other deep learning fields. We hope that future research in these directions will make deep learning more trustworthy and explainable and will facilitate its application in different fields.

Bibliography

Manjushree B. Aithal and Xiaohua Li. Boundary defense against black-box adversarial attacks, 2022.

A. Al-Dujaili and U.-M. O'Reilly. There are no bit parts for sign bits in black-box attacks. In *ICLR*, 2020.

M. Alzantot, Y. Sharma, S. Chakraborty, and M. Srivastava. Genattack: practical black-box attacks with gradient-free optimization. *Genetic and Evolutionary Computation Conference (GECCO)*, 2019.

Maksym Andriushchenko and Nicolas Flammarion. Understanding and improving fast adversarial training. In *NeurIPS*, 2020.

Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square attack: a query-efficient black-box adversarial attack via random search. In *ECCV*, 2020.

Marcin Andrychowicz, Misha Denil, Sergio Gomez Colmenarejo, Matthew W. Hoffman, D. Pfau, Tom Schaul, and N. D. Freitas. Learning to learn by gradient descent by gradient descent. In *NIPS*, 2016.

Anurag Arnab, Ondrej Miksik, and Philip HS Torr. On the robustness of semantic segmentation models to adversarial attacks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 888–897, 2018.

Anish Athalye, Nicholas Carlini, and David A. 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)*, 2018a. URL `http://proceedings.mlr.press/v80/athalye18a.html`.

Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. In *ICML*, 2018b.

Maximilian Augustin, Alexander Meinke, and Matthias Hein. Adversarial robustness on in- and out-distribution improves explainability. In *ECCV*, 2020.

Yutong Bai, Jieru Mei, Alan L Yuille, and Cihang Xie. Are transformers more robust than cnns? *Advances in Neural Information Processing Systems*, 34:26831–26843, 2021.

Sriram Balasubramanian and Soheil Feizi. Towards better input masking for convolutional neural networks, 2022.

Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. BEit: BERT pre-training of image transformers. In *International Conference on Learning Representations (ICLR)*, 2022. URL `https://openreview.net/forum?id=p-BhZSz59o4`.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation. *arXiv e-prints*, art. arXiv:1308.3432, August 2013.

Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 387–402. Springer, 2013.

Wieland Brendel and Matthias Bethge. Approximating CNNs with bag-of-local-features models works surprisingly well on ImageNet. In *International Conference on Learning Representations*, 2019.

Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. In *ICLR*, 2018.

Tom Brown, Dandelion Mane, Aurko Roy, Martin Abadi, and Justin Gilmer. Adversarial patch. In *Conference on Neural Information Processing System (NIPS)*, 2017. URL `https://arxiv.org/pdf/1712.09665.pdf`. arXiv: 1712.09665.

Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Coco-stuff: Thing and stuff classes in context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

Zikui Cai, Chengyu Song, Srikanth Krishnamurthy, Amit Roy-Chowdhury, and M. Salman Asif. Blackbox attacks via surrogate ensemble search. In *NeurIPS*, 2022a.

Zikui Cai, Xinxin Xie, Shasha Li, Mingjun Yin, Chengyu Song, Srikanth V. Krishnamurthy, Amit K.

Roy-Chowdhury, and M. Salman Asif. Context-aware transfer attacks for object detection. In *AAAI*, 2022b.

Zikui Cai, Yaoteng Tan, and M. Salman Asif. Ensemble-based blackbox attacks on dense prediction. In *ICML*, 2023.

Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM workshop on artificial intelligence and security*, pages 3–14, 2017a.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. Ieee, 2017b.

Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian Goodfellow, Aleksander Madry, and Alexey Kurakin. On evaluating adversarial robustness. *arXiv preprint arXiv:1902.06705*, 2019.

Nicholas Carlini, Florian Tramer, J Zico Kolter, et al. (certified!!) adversarial robustness for free! *arXiv preprint arXiv:2206.10550*, 2022.

Yair Carmon, Aditi Raghunathan, Ludwig Schmidt, Percy Liang, and John C. Duchi. Unlabeled data improves adversarial robustness. In *NeurIPS*, 2019.

J. Chen, M. I Jordan, and Wainwright M. J. HopSkipJumpAttack: a query-efficient decision-based attack. In *IEEE Symposium on Security and Privacy (SP)*, 2019.

Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018.

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*, pages 15–26, 2017a.

Sizhe Chen, Zhehao Huang, Qinghua Tao, Yingwen Wu, Cihang Xie, and Xiaolin Huang. Adversarial attack on attackers: Post-process to mitigate black-box score-based query attacks. In *NeurIPS*, 2022.

Tianlong Chen, Xiaohan Chen, Wuyang Chen, Howard Heaton, Jialin Liu, Zhangyang Wang, and Wotao Yin. Learning to optimize: A primer and a benchmark. *arXiv: 2103.12828*, 2021.

Yutian Chen, Matthew W. Hoffman, Sergio Gomez Colmenarejo, Misha Denil, Timothy P. Lillicrap, Matt Botvinick, and Nando de Freitas. Learning to learn without gradient descent by gradient descent. In *ICML*, 2017b.

S. Cheng, Y. Dong, T. Pang, H. Su, and J. Zhu. Improving black-box adversarial attacks with a transfer-based prior. In *NeurIPS*, 2019.

Zhiyuan Cheng, James Liang, Hongjun Choi, Guanhong Tao, Zhiwen Cao, Dongfang Liu, and Xiangyu Zhang. Physical attack on monocular depth estimation with optimal adversarial patches. *ArXiv*, abs/2207.04718, 2022.

Ping-yeh Chiang, Renkun Ni, Ahmed Abdelkader, Chen Zhu, Chris Studor, and Tom Goldstein. Certified defenses for adversarial patches. In *International Conference on Learning Representations*, 2020.

Aran Chindaudom, Prarinya Siritanawan, Karin Sumongkayothin, and Kazunori Kotani. Surreptitious adversarial examples through functioning QR code. *J. Imaging*, 8(5):122, 2022. doi: 10.3390/jimaging8050122. URL `https://doi.org/10.3390/jimaging8050122`.

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, pages 1310–1320. PMLR, 2019.

MMSegmentation Contributors. MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark. `https://github.com/open-mmlab/mmsegmentation`, 2020.

Francesco Croce and Matthias Hein. Sparse and imperceivable adversarial attacks. In *ICCV*, 2019.

Francesco Croce and Matthias Hein. Minimally distorted adversarial examples with a fast adaptive boundary attack. In *International Conference on Machine Learning*, pages 2196–2205. PMLR, 2020a.

Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, 2020b.

Francesco Croce and Matthias Hein. Mind the box: l_1-apgd for sparse adversarial attacks on image classifiers. In *ICML*, 2021.

Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. Robustbench: a standardized adversarial robustness benchmark. *arXiv:2010.09670*, 2020a.

Francesco Croce, Maksym Andriushchenko, Naman D. Singh, Nicolas Flammarion, and Matthias Hein. Sparse-rs: a versatile framework for query-efficient sparse black-box adversarial attacks. *arXiv:2006.12834*, 2020b.

Francesco Croce, Maksym Andriushchenko, Naman D Singh, Nicolas Flammarion, and Matthias Hein. Sparse-rs: a versatile framework for query-efficient sparse black-box adversarial attacks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 6437–6445, 2022.

Franklin C. Crow. Summed-area tables for texture mapping. In *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '84, page 207–212, New York, NY, USA, 1984. Association for Computing Machinery. ISBN 0897911385. doi: 10.1145/800031.808600. URL https://doi.org/10.1145/800031.808600.

Jiequan Cui, Shu Liu, Liwei Wang, and Jiaya Jia. Learnable boundary guided adversarial training. In *ICCV*, 2021.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

Gavin Weiguang Ding, Luyu Wang, and Xiaomeng Jin. AdverTorch v0.1: An adversarial robustness toolbox based on pytorch. *arXiv:1902.07623*, 2019.

Gavin Weiguang Ding, Yash Sharma, Kry Yik Chau Lui, and Ruitong Huang. MMA training: Direct input space margin maximization through adversarial training. In *ICLR*, 2020.

Qiaole Dong, Chenjie Cao, and Yanwei Fu. Incremental transformer structure enhanced image inpainting with masking positional encoding. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

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. URL https://openreview.net/forum?id=YicbFdNTTy.

Jiawei Du, Hu Zhang, Joey Tianyi Zhou, Yi Yang, and Jiashi Feng. Query-efficient meta attack to deep neural networks. In *ICLR*, 2020.

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.

Ruediger Ehlers. Formal verification of piece-wise linear feed-forward neural networks. In *International Symposium on Automated Technology for Verification and Analysis*, pages 269–286. Springer, 2017.

Logan Engstrom, Andrew Ilyas, Hadi Salman, Shibani Santurkar, and Dimitris Tsipras. Robustness (python library), 2019. URL `https://github.com/MadryLab/robustness`.

Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1625–1634, 2018.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017.

Junjie Fu, Jian Sun, and Gang Wang. Boosting black-box adversarial attacks with meta learning, 2022.

Xavier Gastaldi. Shake-Shake regularization. *arXiv e-prints*, art. arXiv:1705.07485, May 2017.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014a.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014b.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015.

Sven Gowal, Krishnamurthy Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Relja Arandjelovic, Timothy Mann, and Pushmeet Kohli. On the effectiveness of interval bound propagation for training verifiably robust models. *arXiv preprint arXiv:1810.12715*, 2018.

Sven Gowal, Krishnamurthy (Dj) Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Relja Arandjelovic, Timothy Mann, and Pushmeet Kohli. Scalable veri-

fied training for provably robust image classification. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

Sven Gowal, Chongli Qin, Jonathan Uesato, Timothy Mann, and Pushmeet Kohli. Uncovering the limits of adversarial training against norm-bounded adversarial examples. *arXiv: 2010.03593*, 2021.

Jindong Gu, Hengshuang Zhao, Volker Tresp, and Philip Torr. Adversarial examples on segmentation models can be easy to transfer, 2021.

C. Guo, J. R Gardner, Y. You, A. G. Wilson, and K. Q. Weinberger. Simple black-box adversarial attacks. In *ICML*, 2019.

Husheng Han, Kaidi Xu, Xing Hu, Xiaobing Chen, Ling Liang, Zidong Du, Qi Guo, Yanzhi Wang, and Yunji Chen. Scalecert: Scalable certified defense against adversarial patches with sparse superficial layers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Gavin Hartnett, Li Ang Zhang, Caolionn L O'Connell, Andrew J. Lohn, and Jair Aguirre. Empirical evaluation of physical adversarial patch attacks against overhead object detection models. *ArXiv*, abs/2206.12725, 2022.

Jamie Hayes. On visible adversarial perturbations & digital watermarking. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPR)*, 2018. URL `http://openaccess.thecvf.com/content_cvpr_2018_workshops/w32/html/Hayes_On_Visible_Adversarial_CVPR_2018_paper.html`.

Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016a.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016b.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition (CVPR)*, 2016c. URL `https://arxiv.org/abs/1512.03385`.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners, 2021. arXiv:2111.06377.

Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. Support vector machines. *IEEE Intelligent Systems and their applications*, 13(4):18–28, 1998.

Matthias Hein and Maksym Andriushchenko. Formal guarantees on the robustness of a classifier against adversarial manipulation. *Advances in neural information processing systems*, 30, 2017.

Dan Hendrycks, Kimin Lee, and Mantas Mazeika. Using pre-training can improve model robustness and uncertainty. In *ICML*, 2019.

Hanbin Hong and Yuan Hong. Certifiable black-box attack: Ensuring provably successful attack for adversarial examples, 2023.

Hanbin Hong, Binghui Wang, and Yuan Hong. Unicr: Universally approximated certified robustness via randomized smoothing. *arXiv preprint arXiv:2207.02152*, 2022.

Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *IEEE PAMI*, 2021.

Lang Huang, Chao Zhang, and Hongyang Zhang. Self-adaptive training: beyond empirical risk minimization. In *NeurIPS*, 2020.

Yiran Huang, Yexu Zhou, Michael Hefenbrock, Till Riedel, Likun Fang, and Michael Beigl. Universal distributional decision-based black-box adversarial attack with reinforcement learning, 2022.

Yuheng Huang and Yuanchun Li. Zero-shot certified defense against adversarial patches with vision transformers, 2021. arXiv:2111.10481.

Zhichao Huang and Tong Zhang. Black-box adversarial attack with transferable model-based embedding. In *ICLR*, 2020.

A. Ilyas, L. Engstrom, A. , and J. Lin. Black-box adversarial attacks with limited queries and information. *ICML*, 2018.

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *NeurIPS*, 2019.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 448–456. JMLR.org, 2015.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparametrization with gumbel-softmax. In *ICLR*, 2017.

Xiaojun Jia, Yong Zhang, Baoyuan Wu, Ke Ma, Jue Wang, and Xiaochun Cao. Las-at: Adversarial training with learnable attack strategy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13398–13408, 2022.

Kaixun Jiang, Zhaoyu Chen, Tony Huang, Jiafeng Wang, Dingkang Yang, Bo Li, Yan Wang, and Wenqiang Zhang. Efficient decision-based black-box patch attacks on video recognition, 2023.

Danny Karmon, Daniel Zoran, and Yoav Goldberg. LaVAN: Localized and visible adversarial noise. In *International Conference on Machine Learning (ICML)*, pages 2507–2515, 2018. URL `https://proceedings.mlr.press/v80/karmon18a.html`.

G. Katz, C. Barrett, D. Dill, K. Julian, and M. Kochenderfer. Reluplex: An efficient smt solver for verifying deep neural networks. In *CAV*, 2017.

Valentin Khrulkov and Ivan Oseledets. Art of singular vectors and universal adversarial perturbations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8562–8570, 2018.

Soma Kontár and András Horváth. On the feasibility and generality of patch-based adversarial attacks on semantic segmentation problems. In *ICPRAI*, 2022.

Alex Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto*, 05 2009.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *Artificial intelligence safety and security*, pages 99–112. Chapman and Hall/CRC, 2018.

Cassidy Laidlaw and Soheil Feizi. Functional adversarial attacks. In *NeurIPS*, 2019.

Raz Lapid and Moshe Sipper. Patch of invisibility: Naturalistic black-box adversarial attacks on object detectors, 2023.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

Mark Lee and J. Zico Kolter. On physical adversarial patches for object detection. *ArXiv*, abs/1906.11897, 2019a.

Mark Lee and J. Zico Kolter. On physical adversarial patches for object detection. *International Conference on Machine Learning (Workshop)*, 2019b. URL `http://arxiv.org/abs/1906.11897`.

Alexander Levine and Soheil Feizi. (De)Randomized Smoothing for Certifiable Defense against Patch Attacks. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6465–6475. Curran Associates, Inc., 2020.

Chu-Tak Li, Wan-Chi Siu, Zhi-Song Liu, Li-Wen Wang, and Daniel Pak-Kong Lun. Deepgin: Deep generative inpainting network for extreme image inpainting, 2020a.

Huichen Li, Xiaojun Xu, Xiaolu Zhang, Shuang Yang, and Bo Li. Qeba: Query-efficient boundary-based blackbox attack. In *CVPR*, 2020b.

Juncheng Billy Li, Frank R. Schmidt, and J. Zico Kolter. Adversarial camera stickers: A physical camera-based attack on deep learning systems. *ArXiv*, abs/1904.00759, 2019.

Shasha Li, Abhishek Aich, Shitong Zhu, M. Salman Asif, Chengyu Song, Amit K. Roy-Chowdhury, and Srikanth V. Krishnamurthy. Adversarial attacks on black box video classifiers: Leveraging the power of geometric transformations. In *NeurIPS*, 2021.

Siyuan Liang, Longkang Li, Yanbo Fan, Xiaojun Jia, Jingzhi Li, Baoyuan Wu, and Xiaochun Cao. A large-scale multiple-objective method for black-box attack against object detection. In *ECCV*, 2022a.

Siyuan Liang, Baoyuan Wu, Yanbo Fan, Xingxing Wei, and Xiaochun Cao. Parallel rectangle flip attack: A query-based black-box attack against object detection, 2022b.

Wan-Yi Lin, Fatemeh Sheikholeslami, jinghao shi, Leslie Rice, and J Zico Kolter. Certified robustness against physically-realizable patch attack via randomized cropping, 2021. URL `https://openreview.net/forum?id=vttv9ADGuWF`.

Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks, 2017.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *International Conference on Computer Vision (ICCV)*, 2021.

Nicholas A. Lord, Romain Mueller, and Luca Bertinetto. Attacking deep networks with surrogate-based adversarial black-box methods is easy. In *ICLR*, 2022.

I. Loshchilov and F. Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations (ICLR) 2017 Conference Track*, April 2017a.

Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with warm restarts. In *ICLR*, 2017b.

Giulio Lovisotto, Nicole Finnie, Mauricio Munoz, Chaithanya Kumar Mummadi, and Jan Hendrik Metzen. Give me your attention: Dot-product attention considered harmful for adversarial patch robustness. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15234–15243, 2022.

Jiajun Lu, Hussein Sibai, Evan Fabry, and David Forsyth. No need to worry about adversarial examples in object detection in autonomous vehicles. *arXiv preprint arXiv:1707.03501*, 2017.

Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables. In *ICLR*, 2017.

A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Valdu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018a.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018b. URL `https://arxiv.org/abs/1706.06083`.

Natalie Maus, Patrick Chao, Eric Wong, and Jacob Gardner. Black box adversarial prompting for foundation models, 2023.

Michael McCoyd, Won Park, Steven Chen, Neil Shah, Ryan Roggenkemper, Minjune Hwang, Jason Xinyu Liu, and David Wagner. Minority reports defense: Defending against adversarial patches, 2020. arXiv:2004.13799.

Jan Hendrik Metzen and Maksym Yatsura. Efficient certified defenses against patch attacks on image classifiers. In *International Conference on Learning Representations (ICLR)*, 2021. URL `https://openreview.net/forum?id=hr-3PMvDpil`.

Jan Hendrik Metzen, Mummadi Chaithanya Kumar, Thomas Brox, and Volker Fischer. Universal adversarial perturbations against semantic image segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 2755–2764, 2017a.

Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. *arXiv preprint arXiv:1702.04267*, 2017b.

Jan Hendrik Metzen, Nicole Finnie, and Robin Hutmacher. Meta adversarial training against universal patches. *arXiv preprint arXiv:2101.11453*, 2021.

Jan Hendrik Metzen, Robin Hutmacher, N. Grace Hua, Valentyn Boreiko, and Dan Zhang. Identification of systematic errors of image classifiers on rare subgroups, 2023.

L. Meunier, J. Atif, and O. Teytaud. Yet another but more efficient black-box adversarial attack: tiling and evolution strategies. *arXiv preprint, arXiv:1910.02244*, 2019.

Matthew Mirman, Timon Gehr, and Martin Vechev. Differentiable abstract interpretation for provably robust neural networks. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 3578–3586. PMLR, 10–15 Jul 2018. URL `https://proceedings.mlr.press/v80/mirman18b.html`.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 86–94, 2017.

Konda Reddy Mopuri, Utkarsh Ojha, Utsav Garg, and R Venkatesh Babu. Nag: Network for adversary generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 742–751, 2018.

Marius Mosbach, Maksym Andriushchenko, Thomas Trost, Matthias Hein, and Dietrich Klakow. Logit pairing methods can fool gradient-based attacks. *arXiv preprint arXiv:1810.12042*, 2018.

Muzammal Naseer, Salman Khan, and Fatih Porikli. Local gradients smoothing: Defense against localized adversarial attacks. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2019. URL `https://doi.org/10.1109/WACV.2019.00143`.

Gaurav Kumar Nayak, Inder Khatri, Shubham Randive, Ruchit Rawal, and Anirban Chakraborty. Data-free defense of black box models against adversarial attacks, 2022.

Federico Nesti, Giulio Rossolini, Saasha Nair, Alessandro Biondi, and Giorgio C. Buttazzo. Evaluating the robustness of semantic segmentation for autonomous driving against real-

world adversarial patch attacks. *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2826–2835, 2022.

Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv:1803.02999*, 2018.

Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *2017 ACM Asia Conference on Computer and Communications Security*, 2017.

Svetlana Pavlitskaya, Jonas Hendl, Sebastian Kleim, Leopold Müller, Fabian Wylczoch, and Johann Marius Zöllner. Suppress with a patch: Revisiting universal adversarial patch attacks against object detection. *ArXiv*, abs/2209.13353, 2022.

J. Ross Quinlan. Simplifying decision trees. *Int. J. Man Mach. Stud.*, 27(3):221–234, 1987. doi: 10.1016/S0020-7373(87)80053-6. URL https://doi.org/10.1016/S0020-7373(87)80053-6.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

Anurag Ranjan, Joel Janai, Andreas Geiger, and Michael J. Black. Attacking optical flow. In *ICCV*, 2019a.

Anurag Ranjan, Joel Janai, Andreas Geiger, and Michael J. Black. Attacking optical flow. In *International Conference on Computer Vision (ICCV)*, 2019b.

Leslie Rice, Eric Wong, and J. Zico Kolter. Overfitting in adversarially robust deep learning. In *ICML*, 2020.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

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 *CVPR*, 2019.

Yangjun Ruan, Yuanhao Xiong, Sashank Reddi, Sanjiv Kumar, and Cho-Jui Hsieh. Learning to learn by zeroth-order oracle. In *ICLR*, 2020.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

Aniruddha Saha, Akshayvarun Subramanya, Koninika Patil, and Hamed Pirsiavash. Adversarial Patches Exploiting Contextual Reasoning in Object Detection. *arXiv e-prints*, art. arXiv:1910.00068, Sep 2019.

Hadi Salman, Greg Yang, Huan Zhang, Cho-Jui Hsieh, and Pengchuan Zhang. A convex relaxation barrier to tight robustness verification of neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

Hadi Salman, Andrew Ilyas, Logan Engstrom, Ashish Kapoor, and Aleksander Madry. Do adversarially robust imagenet models transfer better? In *NeurIPS*, 2020a.

Hadi Salman, Mingjie Sun, Greg Yang, Ashish Kapoor, and J Zico Kolter. Denoised smoothing: A provable defense for pretrained classifiers. *Advances in Neural Information Processing Systems*, 33:21945–21957, 2020b.

Hadi Salman, Saachi Jain, Eric Wong, and Aleksander Madry. Certified patch robustness via smoothed vision transformers, 2021. arXiv:2110.07719.

Arthur L. Samuel. Some studies in machine learning using the game of checkers. *IBM J. Res. Dev.*, 3(3):210–229, 1959. doi: 10.1147/rd.33.0210. URL `https://doi.org/10.1147/rd.33.0210`.

Takami Sato, Junjie Shen, Ningfei Wang, Yunhan Jia, Xue Lin, and Qi Alfred Chen. Dirty road can attack: Security of deep learning based automated lane centering under Physical-World attack. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 3309–3326. USENIX Association, August 2021. ISBN 978-1-939133-24-3. URL `https://www.usenix.org/conference/usenixsecurity21/presentation/sato`.

Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61: 85–117, 2015.

Lukas Schott, Jonas Rauber, Matthias Bethge, and Wieland Brendel. Towards the first adversarially robust neural network model on mnist. *arXiv preprint arXiv:1805.09190*, 2018.

M. Seungyong, A. Gaon, and O. S. Hyun. Parsimonious black-box adversarial attacks via efficient combinatorial optimization. In *ICML*, 2019.

Yucheng Shi, Yahong Han, Yu an Tan, and Xiaohui Kuang. Decision-based black-box attack against vision transformers via patch-wise adversarial removal, 2022.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587): 484–489, 2016.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

Chawin Sitawarin, Supriyo Chakraborty, and David Wagner. Improving adversarial robustness through progressive hardening. *arXiv:2003.09347*, 2020.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015a.

Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015b.

Chenghao Sun, Yonggang Zhang, Wan Chaoqun, Qizhou Wang, Ya Li, Tongliang Liu, Bo Han, and Xinmei Tian. Towards lightweight black-box attacks against deep neural networks. In *NeurIPS*, 2022.

Roman Suvorov, Elizaveta Logacheva, Anton Mashikhin, Anastasia Remizova, Arsenii Ashukha, Aleksei Silvestrov, Naejin Kong, Harshith Goka, Kiwoong Park, and Victor S. Lempitsky. Resolution-robust large mask inpainting with fourier convolutions. *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 3172–3182, 2022.

C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *ICLR*, 2014.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016.

Yusuke Tashiro, Yang Song, and Stefano Ermon. Diversity can be transferred: Output diversification for white- and black-box attacks. In *NeurIPS*, 2020.

Simen Thys, Wiebe Van Ranst, and Toon Goedemé. Fooling automated surveillance cameras: Adversarial patches to attack person detection. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPR)*, 2019.

Vincent Tjeng, Kai Y Xiao, and Russ Tedrake. Evaluating robustness of neural networks with mixed integer programming. In *International Conference on Learning Representations*, 2018.

Florian Tramer. Detecting adversarial examples is (nearly) as hard as classifying them. In *International Conference on Machine Learning*, pages 21692–21702. PMLR, 2022.

Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. In *NeurIPS*, 2020.

J. Uesato, B. O'Donoghue, A. Van den Oord, and P. Kohli. Adversarial risk and the dangers of evaluating against weak attacks. In *ICML*, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Viet Quoc Vo, Ehsan Abbasnejad, and Damith C. Ranasinghe. Query efficient decision based sparse attacks against black-box deep learning models. In *ICLR*, 2022.

Wenxuan Wang, Xuelin Qian, Yanwei Fu, and Xiangyang Xue. Dst: Dynamic substitute training for data-free black-box attack. In *CVPR*, 2022.

Yisen Wang, Difan Zou, Jinfeng Yi, James Bailey, Xingjun Ma, and Quanquan Gu. Improving adversarial robustness requires revisiting misclassified examples. In *ICLR*, 2020.

Xingxing Wei, Ying Guo, Jie Yu, and Bo Zhang. Simultaneously optimizing perturbations and positions for black-box adversarial patch attacks. In *TPAMI*, 2022.

Xingxing Wei, Bangzheng Pu, Jiefan Lu, and Baoyuan Wu. Visually adversarial attacks and defenses in the physical world: A survey, 2023.

Olivia Wiles, Isabela Albuquerque, and Sven Gowal. Discovering bugs in vision models using off-the-shelf image generation and captioning, 2023.

Eric Wong and J. Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In Jennifer G. Dy and Andreas Krause, editors, *Proceed-*

ings of the 35th International Conference on Machine Learning (ICML), 2018. URL http://proceedings.mlr.press/v80/wong18a.html.

Eric Wong, Frank R. Schmidt, Jan Hendrik Metzen, and J. Zico Kolter. Scaling provable adversarial defenses. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018 (NeurIPS)*, 2018. URL http://papers.nips.cc/paper/8060-scaling-provable-adversarial-defenses.

Eric Wong, Leslie Rice, and J. Zico Kolter. Fast is better than free: Revisiting adversarial training. In *ICLR*, 2020.

Dongxian Wu, Shu tao Xia, and Yisen Wang. Adversarial weight perturbation helps robust generalization. In *NeurIPS*, 2020.

Han Wu, Sareh Rowlands, and Johan Wahlstrom. Distributed black-box attack against image classification cloud services, 2022.

Chong Xiang and Prateek Mittal. Detectorguard: Provably securing object detectors against localized patch hiding attacks. In *ACM Conference on Computer and Communications Security (CCS)*, 2021a.

Chong Xiang and Prateek Mittal. Patchguard++: Efficient provable attack detection against adversarial patches, 2021b. arXiv:2104.12609.

Chong Xiang, Arjun Nitin Bhagoji, Vikash Sehwag, and Prateek Mittal. Patchguard: A provably robust defense against adversarial patches via small receptive fields and masking. In *30th USENIX Security Symposium (USENIX Security)*, 2021.

Chong Xiang, Saeed Mahloujifar, and Prateek Mittal. Patchcleanser: Certifiably robust defense against adversarial patches for any image classifier. In *31st USENIX Security Symposium (USENIX Security)*, 2022a.

Chong Xiang, Alexander Valtchanov, Saeed Mahloujifar, and Prateek Mittal. Objectseeker: Certifiably robust object detection against patch hiding attacks via patch-agnostic masking, 2022b. arXiv:2202.01811.

Chong Xiang, Chawin Sitawarin, Tong Wu, and Prateek Mittal. Short: Certifiably robust perception against adversarial patch attacks: A survey. *Proceedings Inaugural International Symposium on Vehicle Security & Privacy*, 2023.

Cihang Xie, Jianyu Wang, Zhishuai Zhang, Yuyin Zhou, Lingxi Xie, and Alan Yuille. Adversar-

ial examples for semantic segmentation and object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 1369–1378, 2017.

Shangyu Xie, Han Wang, Yu Kong, and Yuan Hong. Universal 3-dimensional perturbations for black-box attacks on video recognition systems. In *Oakland*, 2021.

Yuanhao Xiong and Cho-Jui Hsieh. Improved adversarial training via learned optimizer. In *ECCV*, 2020.

Changming Xu and Gagandeep Singh. Robust universal adversarial perturbations. *ArXiv*, abs/2206.10858, 2022.

Chengyuan Yao, Pavol Bielik, Petar Tsankov, and Martin Vechev. Automated discovery of adaptive attacks on adversarial defenses. *arXiv:2102.11860*, 2021.

Maksym Yatsura, Jan Metzen, and Matthias Hein. Meta-learning the search distribution of black-box random search based adversarial attacks. *Advances in Neural Information Processing Systems*, 34:30181–30195, 2021.

Maksym Yatsura, Kaspar Sakmann, N Grace Hua, Matthias Hein, and Jan Hendrik Metzen. Certified defences against adversarial patch attacks on semantic segmentation. In *International Conference on Learning Representations (ICLR)*, 2023.

D Yin, R. G. Lopes, J. Shlens, E. D Cubuk, and J. Gilmer. A Fourier perspective on model robustness in computer vision. In *NeurIPS*, 2019.

Fei Yin, Yong Zhang, Baoyuan Wu, Yan Feng, Jingyi Zhang, Yanbo Fan, and Yujiu Yang. Generalizable black-box adversarial attack with meta learning. In *TPAMI*, 2022.

Shengming Yuan, Qilong Zhang, Lianli Gao, Yaya Cheng, and Jingkuan Song. Natural color fool: Towards boosting black-box unrestricted attacks. In *NeurIPS*, 2022.

Zheng Yuan, Jie Zhang, Yunpei Jia, Chuanqi Tan, Tao Xue, and Shiguang Shan. Meta gradient adversarial attack. In *ICCV*, 2021.

Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12104–12113, 2022.

Chaoning Zhang, Philipp Benz, Adil Karjauv, Jae Won Cho, Kang Zhang, and In So Kweon. Investigating top-k white-box and transferable black-box attack. In *ICLR*, 2022a.

Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. You only propagate once: Accelerating adversarial training via maximal principle. In *NeurIPS*, 2019a.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *International conference on machine learning*, pages 7472–7482. PMLR, 2019b.

Jingfeng Zhang, Xilie Xu, Bo Han, Gang Niu, Lizhen Cui, Masashi Sugiyama, and Mohan Kankanhalli. Attacks which do not kill training make adversarial learning stronger. In *ICML*, 2020a.

Jingfeng Zhang, Jianing Zhu, Gang Niu, Bo Han, Masashi Sugiyama, and Mohan Kankanhalli. Geometry-aware instance-reweighted adversarial training. In *ICLR*, 2021.

Yimeng Zhang, Yuguang Yao, Jinghan Jia, Jinfeng Yi, Mingyi Hong, Shiyu Chang, and Sijia Liu. How to robustify black-box ml models? a zeroth-order optimization perspective. In *ICLR*, 2022b.

Zhanyuan Zhang, Benson Yuan, Michael McCoyd, and David Wagner. Clipped BagNet: Defending Against Sticker Attacks with Clipped Bag-of-features. In *3rd Deep Learning and Security Workshop (DLS)*, 2020b.

Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Bolei Zhou, Aditya Khosla, Àgata Lapedriza, Antonio Torralba, and Aude Oliva. Places: An image database for deep scene understanding. *Journal of Vision*, 17, 10 2016. doi: 10.1167/17.10.296.

Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

Shuai Zhou, Chi Liu, Dayong Ye, Tianqing Zhu, Wanlei Zhou, and Philip S. Yu. Adversarial attacks and defenses in deep learning: From a perspective of cybersecurity. *ACM Comput. Surv.*, 55(8), dec 2022. ISSN 0360-0300. doi: 10.1145/3547330. URL `https://doi.org/10.1145/3547330`.

Haomin Zhuang, Yihua Zhang, and Sijia Liu. A pilot study of query-free adversarial attack against stable diffusion, 2023.

Andy Zou, Zifan Wang, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models, 2023.

Appendix A

Supplementary illustrations

A.1 Demasked Smoothing Visualisation

We provide additional illustrations for the Demasked Smoothing pipeline (Chapter 5 in case of 3-mask (Figure A.1) and 4-mask (Figure A.2.) See details in Section 5.2.1. We illustrate the steps of masking, demasking, segmentation and the aggregation results. We use ZITS demasking model (Dong et al., 2022) and BEIT-B segmentation model(Bao et al., 2022).

A.2 Examples of certification maps

In this section, provide examples of certification maps for certified recovery and certified detection with different images from ADE20K (Zhou et al., 2017) with Swin segmentation model (Liu et al., 2021) (Figure A.3, A.4). For each image we illustrate the recovery and detection maps as well as the ground truth.

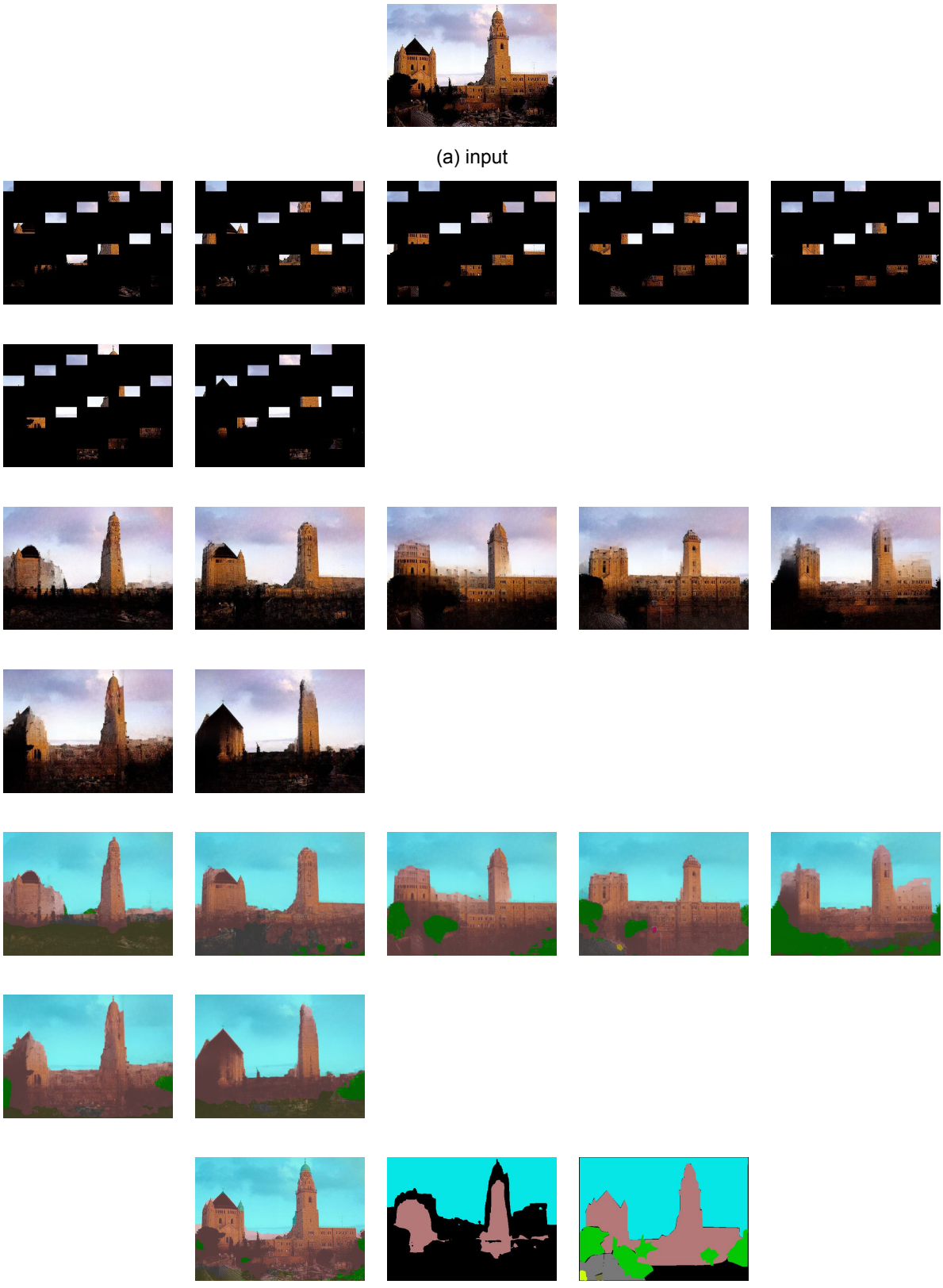


Figure A.1: Demasked Smoothing recovery with 3-mask (Section 5.2.1). Illustration for an image from ADE20K (Zhou et al., 2017) with ZITS demasking(Dong et al., 2022) and BEIT-B segmentation (Bao et al., 2022).

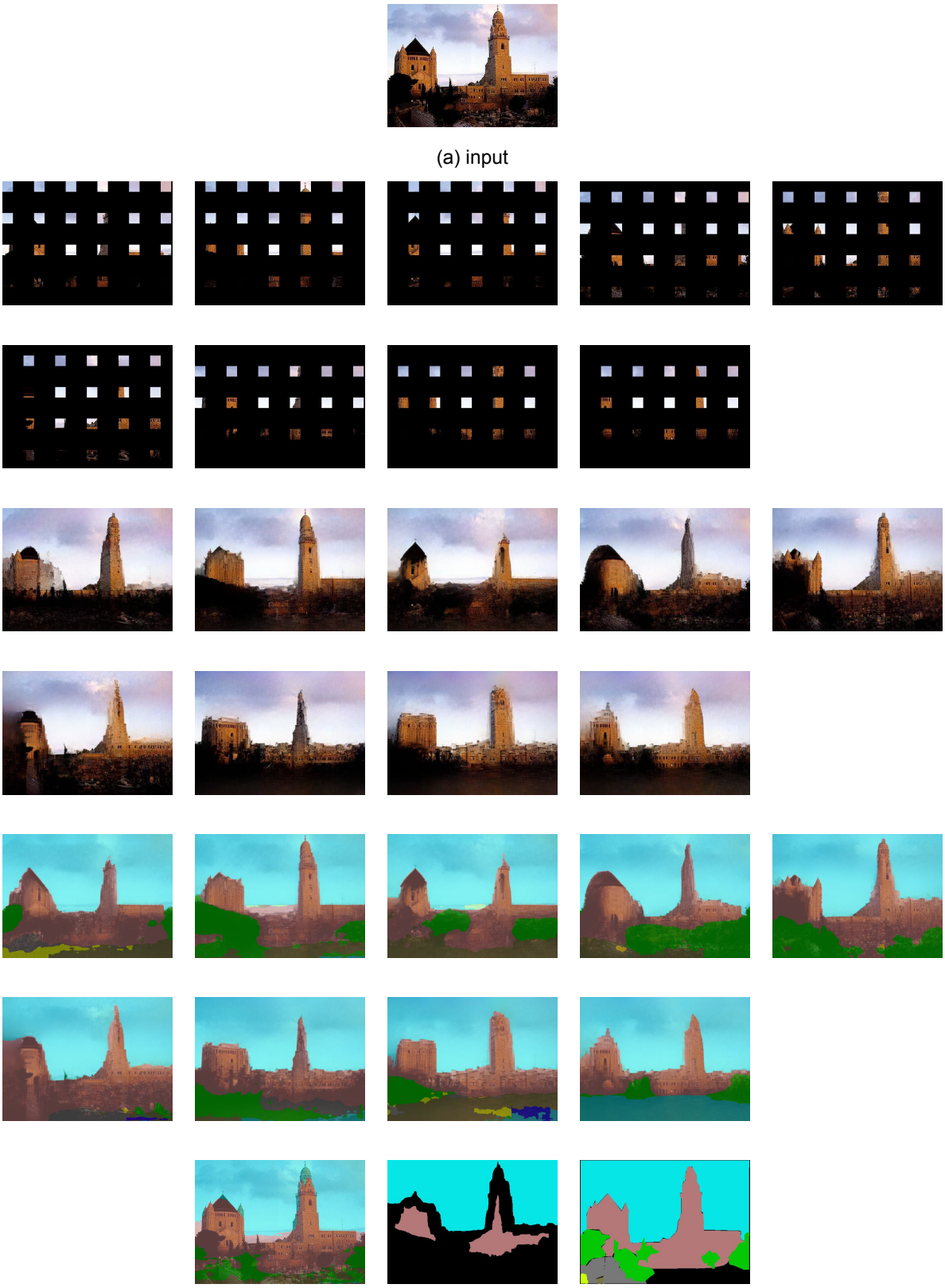
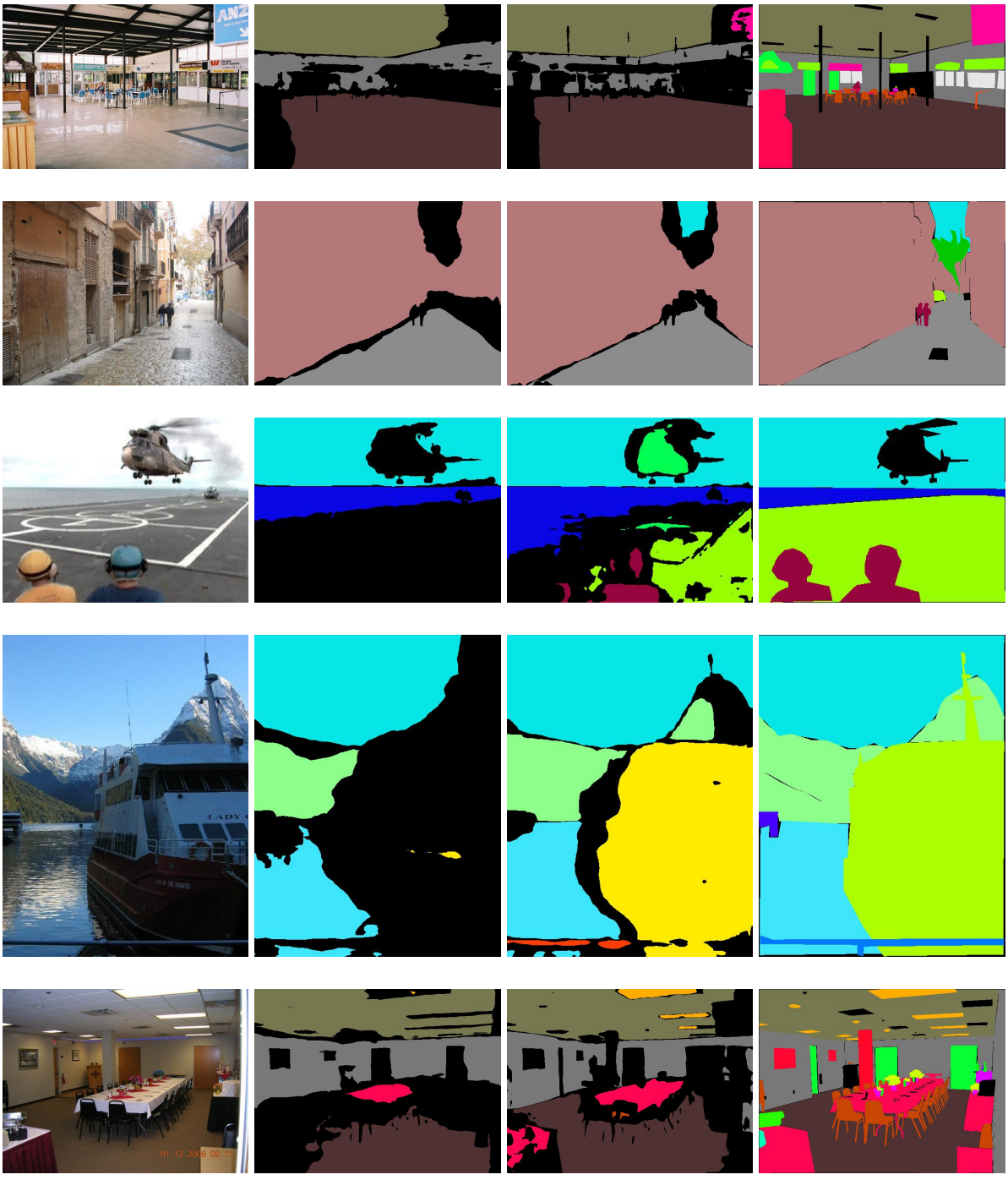
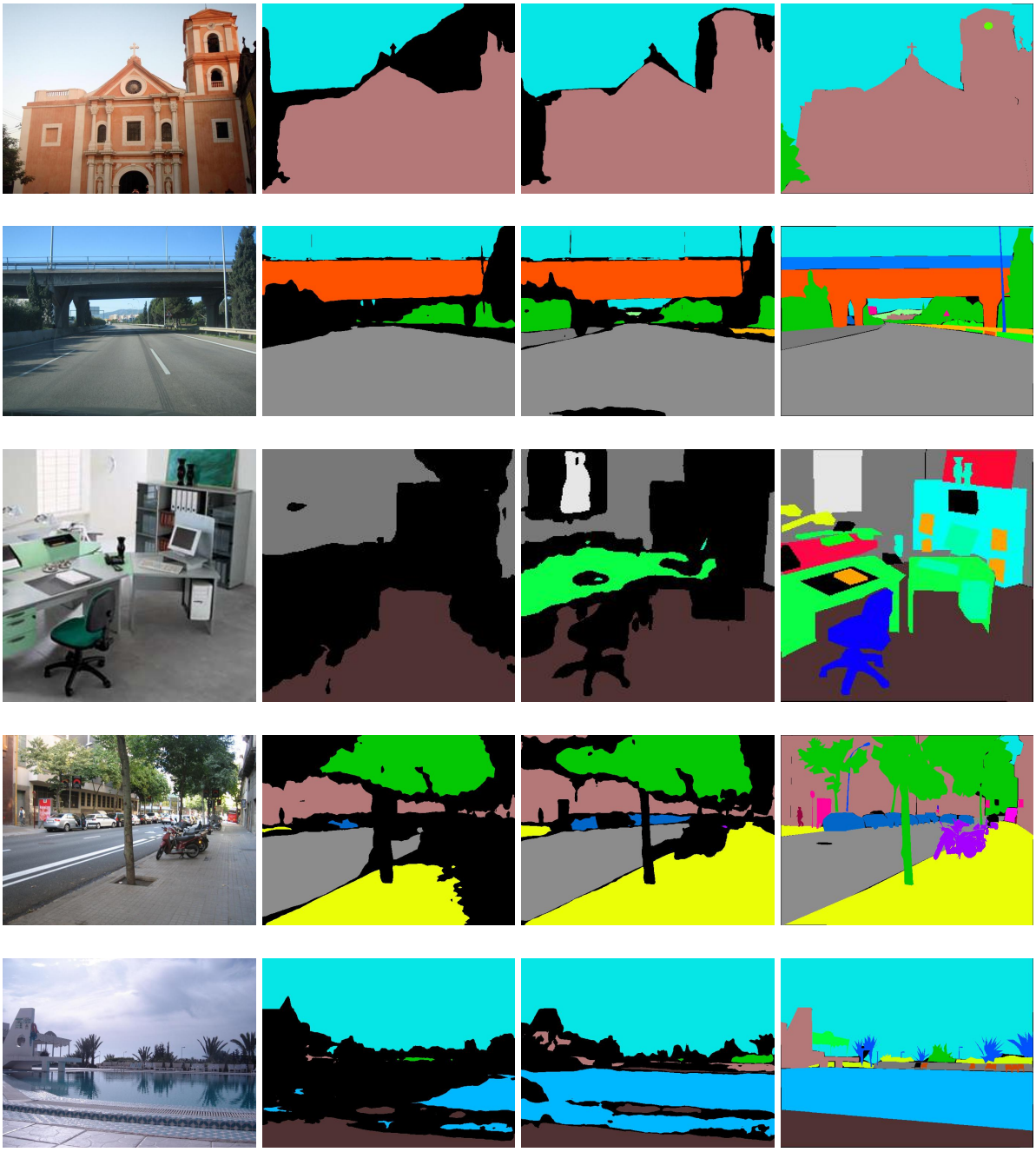


Figure A.2: Demasked Smoothing recovery with 4-mask (Section 5.2.1). Illustration for an image from ADE20K (Zhou et al., 2017) with ZITS demasking(Dong et al., 2022) and BEIT-B segmentation (Bao et al., 2022).



(a) original image (b) recovery map (c) detection map (d) ground truth

Figure A.3: Certification map examples on ADE20K (Zhou et al., 2017) with ZITS (Dong et al., 2022) and Swin Liu et al. (2021).



(a) original image (b) recovery map (c) detection map (d) ground truth

Figure A.4: Certification map examples on ADE20K (Zhou et al., 2017) with ZITS (Dong et al., 2022) and Swin Liu et al. (2021).