PatchCleanser: Certifiably Robust Defense against Adversarial Patches for Any Image Classifier

Chong Xiang, Saeed Mahloujifar, and Prateek Mittal, *Princeton University*

https://www.usenix.org/conference/usenixsecurity22/presentation/xiang

This paper is included in the Proceedings of the
31st USENIX Security Symposium.

August 10–12, 2022 • Boston, MA, USA

978-1-939133-31-1

Open access to the Proceedings of the
31st USENIX Security Symposium is
sponsored by USENIX.

PatchCleanser: Certifiably Robust Defense against Adversarial Patches for Any Image Classifier

Chong Xiang
Princeton University

Saeed Mahloujifar
Princeton University

Prateek Mittal
Princeton University

Abstract

The adversarial patch attack against image classification models aims to inject adversarially crafted pixels within a restricted image region (i.e., a patch) for inducing model misclassification. This attack can be realized in the physical world by printing and attaching the patch to the victim object; thus, it imposes a real-world threat to computer vision systems. To counter this threat, we design PatchCleanser as a certifiably robust defense against adversarial patches. In PatchCleanser, we perform two rounds of *pixel masking* on the input image to neutralize the effect of the adversarial patch. This image-space operation makes PatchCleanser compatible with any state-of-the-art image classifier for achieving high accuracy. Furthermore, we can prove that PatchCleanser will always predict the correct class labels on certain images against any adaptive white-box attacker within our threat model, achieving certified robustness. We extensively evaluate PatchCleanser on the ImageNet, ImageNette, and CIFAR-10 datasets and demonstrate that our defense achieves similar clean accuracy as state-of-the-art classification models and also significantly improves certified robustness from prior works. Remarkably, PatchCleanser achieves 83.9% top-1 clean accuracy and 62.1% top-1 certified robust accuracy against a 2%-pixel square patch anywhere on the image for the 1000-class ImageNet dataset.[1]

1 Introduction

The adversarial patch attack [4, 21, 60] against image classification models aims to induce test-time misclassification. A patch attacker injects adversarially crafted pixels within a localized and restricted region (i.e., a patch) and can realize a physical-world attack by printing and attaching the patch to the victim object. The physically realizable nature of patch attacks imposes a significant threat to real-world computer vision systems.

To secure the deployment of critical computer vision systems, there has been an active research thread on certifiably robust defenses against adversarial patches [7, 25, 27, 33, 55, 61]. These defenses aim to provide a certifiable guarantee on making correct predictions on certain images, even in the presence of an adaptive white-box attacker. This strong robustness property provides a pathway towards ending the arms race between attackers and defenders.

Limitation of prior works: the dependence on specific model architectures. While prior works have made significant contributions to certifiable robustness, their defense performance is hindered by their dependence on specific model architectures. The most common architecture constraint of state-of-the-art certifiably robust defenses against patch attacks [25, 27, 33, 55, 61] is the dependence on small receptive fields (receptive field is the region of the input image that an extracted feature is looking at, or affected by). The small receptive field bounds the number of features that can be corrupted by the adversarial patch but also limits the information received by each feature. As a result, defenses with small receptive fields are limited in their classification accuracy: for example, the best top-1 clean accuracy on ImageNet [11] achieved by prior certifiably robust defenses is around 55% [27, 55] while state-of-the-art undefended classification models can attain an accuracy of 80%-90% [13, 22, 48, 52]. The poor clean accuracy discourages the real-world deployment of proposed defenses and also limits the achievable robustness (since the robust accuracy can be no higher than the clean accuracy).[2]

Limitation of prior works: abstention from predictions. Minority Reports (MR) [30] is the only certifiably robust defense with no assumption on the model architecture; however, it suffers from a weaker security guarantee of being able to only *detect* a patch attack (i.e., alert when an attack is de-

[1] Our source code is available at https://github.com/inspire-group/PatchCleanser.

[2] Chiang et al. [7] proposed the first certifiably robust defense against adversarial patches via Interval Bound Propagation [16, 34]. This defense does not rely on small receptive fields but requires extremely expensive model training. As a result, it is only applicable to small classification models (with limited performance) and low-resolution images.

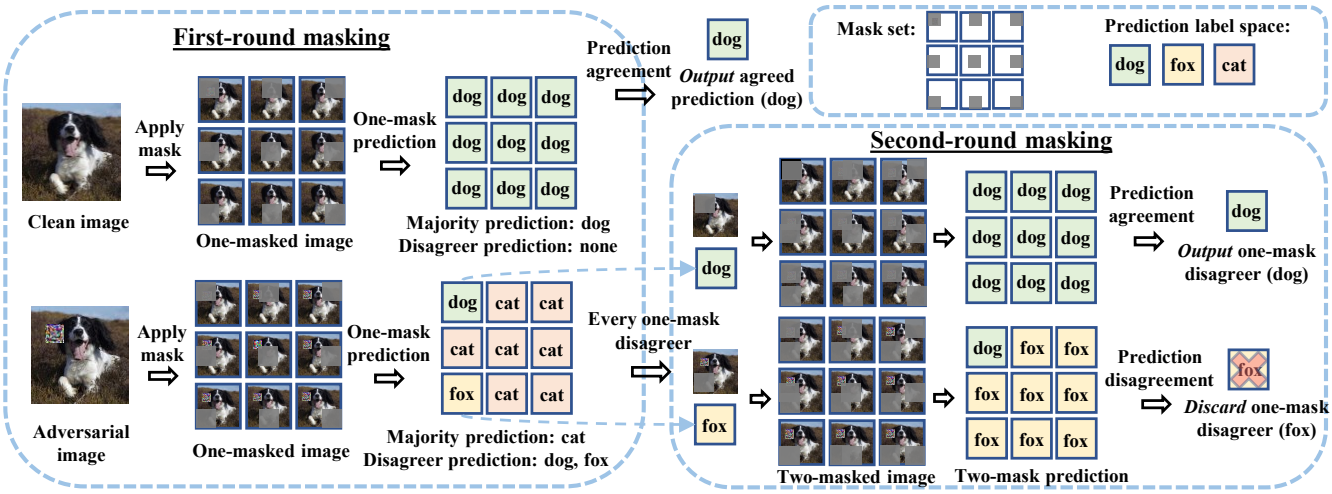

Figure 1: **Overview of double-masking defense.** The defense applies masks to the input image and evaluates model prediction on every masked image. *Clean image:* all one-mask predictions typically agree on the correct label ("dog"), our defense outputs the agreed prediction. *Adversarial image:* one-mask predictions have a disagreement; we aim to recover the benign prediction. We first categorize all one-mask predictions into the *majority prediction* (the one with the highest prediction label occurrence; the label "cat" in this example) and *disagreer predictions* (the ones that disagree with the majority; the labels "dog" and "fox"). For every mask that leads to a disagreer prediction, we add a set of second masks and evaluate two-mask predictions. If all two-mask predictions agree with this one-mask disagreer, we *output* its prediction label (the label "dog"; illustrated in the upper row of the second-round masking); otherwise, we *discard* it (the label "fox"; in the lower row of the second-round masking).

tected). As a result, an attacker can force the model to always alert and abstain from making a prediction. The inability of a model to make a prediction can compromise functionality in applications where human fallback is unavailable (e.g., level-5 autonomous vehicles without human drivers).

PatchCleanser: architecture-agnostic certifiably robust image classification (without abstention). In order to overcome the limitations of prior works, we propose PatchCleanser as a certifiably robust image classification (without any abstention) defense that is compatible with any image classifier. The high-level idea of PatchCleanser is to robustly remove/mask all adversarial pixels on the input image so that we can obtain accurate predictions (on the masked images) from *any* state-of-the-art image classifier.

However, the key question is: *How can we mask out the patch if the patch location is unknown?* An intuitive idea is to place a mask at all possible image locations and evaluate model predictions on every masked image. If the mask is large enough, then at least one masked image is benign (i.e., no adversarial pixels) and is likely to give a correct prediction (a similar intuition is used in MR for attack detection [30]). Unfortunately, despite the existence of one benign (and usually correct) masked prediction, it is challenging to robustly distinguish this benign prediction from other masked predictions that can be adversarially manipulated by an adaptive attacker. To solve this challenge, we propose a *double-masking* algorithm that achieves certifiable robustness.

We provide a defense overview in Figure 1. The double-masking algorithm involves two rounds of pixel masking. In the first round of masking (left of the figure), we apply every mask from a *mask set* to the input image and evaluate model predictions on *one-masked* images. The mask set is constructed in a way that at least one mask can remove the entire patch (regardless of the patch location) and give a benign (and usually correct) masked prediction. When the algorithm operates on a clean image, all one-mask predictions usually reach a unanimous agreement, and PatchCleanser will output the agreed label (top of Figure 1). On the other hand, for an adversarial image, since at least one mask can remove the patch and recover the benign prediction, we will see a disagreement between the benign prediction and malicious predictions (left bottom of Figure 1). To robustly identify the benign one-mask prediction, we perform a second round of masking: we apply a set of second masks to every *one-masked* image and use inconsistencies in model predictions on a set of *two-masked* images to filter out all malicious one-mask predictions (right of the figure). We will present the details of our double-masking defense in Section 3.2 and demonstrate that it provides certifiable robustness for certain images against *any patch attacker within our threat model* in Section 3.3.

Evaluation: state-of-the-art clean accuracy and certified robust accuracy. We instantiate PatchCleanser with three representative state-of-the-art architectures for image classification: ResNet [19], Vision Transformer (ViT) [13], and ResMLP [48]. We evaluate our defense performance on three image datasets: ImageNet [11], ImageNette [14],

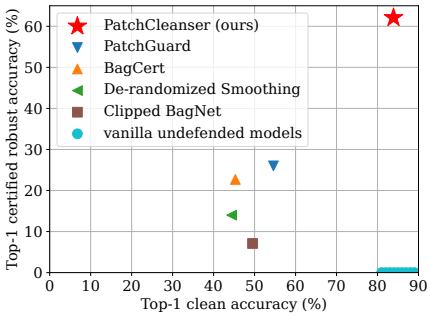

Figure 2: ImageNet clean and certified robust accuracy of PatchCleanser and prior defenses [25, 33, 55, 61]; the certified robust accuracy is evaluated against a 2%-pixel square patch.

CIFAR-10 [23]. We demonstrate that PatchCleanser achieves state-of-the-art (clean) classification accuracy and also greatly improves the certified robust accuracy from prior works [7, 25, 33, 55, 61]. In Figure 2, we plot the clean accuracy and certified robust accuracy of different defenses on the ImageNet dataset [11] to visualize our significant performance improvements. Our contributions can be summarized as follows:

- We present PatchCleanser's double-masking defense that is compatible with any image classifier to mitigate the threat of adversarial patch attacks.
- We formally prove the certifiable robustness of PatchCleanser for certain images against any adaptive white-box attacker within our threat model.
- We evaluate PatchCleanser on three state-of-the-art classification models and three benchmark datasets and demonstrate the significant improvements in clean accuracy and certified robust accuracy (e.g., Figure 2).

2 Problem Formulation

In this section, we formulate image classification models, attack threat models, and our defense objectives.

2.1 Image Classification Model

In this paper, we focus on the image classification problem. We use $\mathcal{X} \subset [0,1]^{W \times H \times C}$ to denote the image space, where each image has width W, height H, number of channels C, and the pixels are re-scaled to $[0,1]$. We further denote the label space as $\mathcal{Y}$. An image classification model is denoted as $\mathbb{F}: \mathcal{X} \to \mathcal{Y}$, which takes an image $\mathbf{x} \in \mathcal{X}$ as input and predicts the class label $y \in \mathcal{Y}$.

We do not make any assumption on the architecture of the image classification model $\mathbb{F}$. Our defense is compatible with any popular model such as ResNet [19], Vision Transformer [13], and ResMLP [48].

2.2 Threat Model

Attack objective. We focus on test-time evasion attacks. Given a model $\mathbb{F}$, an image $\mathbf{x}$, and its true class label y, the attacker aims to find an image $\mathbf{x}' \in \mathcal{A}(\mathbf{x}) \subset \mathcal{X}$ satisfying a constraint $\mathcal{A}$ such that $\mathbb{F}(\mathbf{x}') \neq y$. The constraint $\mathcal{A}$ is defined by the attacker's threat model, which we discuss next.

Attacker capability. The patch attacker has *arbitrary* control over the image pixels in a *restricted* region, and this region can be *anywhere* on the image. Formally, we use a binary tensor $\mathbf{r} \in \{0,1\}^{W \times H}$ to represent the restricted region, where the *pixels within the region are set to* 0 and others are set to 1. We further use $\mathcal{R}$ to denote a set of regions $\mathbf{r}$ (i.e., a set of patches at different locations). Then, we can express the patch attacker's constraint set $\mathcal{A}_{\mathcal{R}}(\mathbf{x})$ as $\{\mathbf{r} \odot \mathbf{x} + (\mathbf{1} - \mathbf{r}) \odot \mathbf{x}' \mid \mathbf{x}, \mathbf{x}' \in \mathcal{X}, \mathbf{r} \in \mathcal{R}\}$, where $\odot$ refers to the element-wise multiplication operator. When clear from the context, we drop $\mathcal{R}$ and use $\mathcal{A}$ instead of $\mathcal{A}_{\mathcal{R}}$.

An open research question: one single square patch at any image location. In this paper, we primarily focus on a popular open research question where $\mathbf{r}$ represents *one square region that can be anywhere on the image* and the defender has a *conservative estimation of the patch size*.[3] This enables a performance comparison with prior works that also focus on this setting [7, 25, 33, 55, 61] (Section 4). Moreover, we note that designing high-performance certifiably robust defenses under this setting is extremely challenging due to attacker's *arbitrary control* over the patch location and patch content as well as *full knowledge* of the defense setup.

Flexibility of PatchCleanser. Nevertheless, our defense design is general and can be easily adapted for even stronger attackers. In addition to evaluating our defense under the setting of *one single square patch* in Section 4, we also quantitatively analyze our defense against attackers who can use *a set of different patch shapes* (e.g., all possible rectangle shapes covering a certain area at any image location) and who can apply *multiple patches* (e.g., two patches at any image location) in Section 5.1.

2.3 Defense Objective

We design PatchCleanser with three major objectives.

Robust classification. We aim to build a defended model $\mathbb{D}$ for *robust classification*. That is, we want to have $\mathbb{D}(\mathbf{x}') = \mathbb{D}(\mathbf{x}) = y$ for a clean data point $(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}$ and any adversarial example $\mathbf{x}' \in \mathcal{A}(\mathbf{x})$. Note that we aim to *recover the correct prediction without any abstention*, which is harder than merely detecting an attack (e.g., Minority Reports [30]).

Certifiable robustness. We aim to design defenses with certifiable robustness [7, 25, 33, 55, 61]: given a clean data point $(\mathbf{x}, y)$, the defended model can always make a correct

[3] We note that similar assumptions on defender's knowledge are also commonly used in defenses against conventional global L_p perturbations. For example, verifiably robust network training [16, 34] and empirical adversarial training [15, 29] need to know the norm and magnitude of the perturbations.

prediction for any adversarial example within the threat model, i.e., $\mathbb{D}(\mathbf{x}') = \mathbb{D}(\mathbf{x}) = y,\ \forall\ \mathbf{x}' \in \mathcal{A}(\mathbf{x})$. We will design a robustness certification procedure, which takes a clean data point $(\mathbf{x}, y)$ and threat model $\mathcal{A}$ as inputs, to check if the robustness can be certified. The certification procedure should account for all possible attackers within the threat model $\mathcal{A}$, who could have full knowledge of our defense and full access to our model parameters. We note that the certification provides a *provable lower bound* for model robustness against adaptive attacks. This is a significant improvement over traditional empirical defenses [10, 18, 35, 36, 40, 54], whose robustness could be undermined by an adaptive attacker.

We note that we only discuss robustness certification for *labeled* images because the certification procedure needs ground-truth labels to check the correctness of model predictions. In our evaluation, we apply our certification procedure to labeled test sets and calculate the fraction of certified images, termed as *certified robust accuracy*, as our robustness metric. This accuracy indicates the estimated robustness (against the strongest adaptive attacks) when we deploy the defense; we do not aim to guarantee robustness/correctness for individual images in the wild.

Compatibility with any model architecture. As discussed in Section 1, prior works [7, 25, 33, 55, 61] on certifiably robust image classification suffer from their dependence on the model architecture (e.g., small receptive fields). Such dependence limits the model performance and hinders the practical deployment of the defense. In PatchCleanser, we aim to design a defense that is compatible with any state-of-the-art model architecture to achieve high defense performance (recall Figure 2) and benefit from any advancement in image classification research.

3 PatchCleanser Design

In this section, we introduce our PatchCleanser defense, which is agnostic to model architectures and achieves certifiable robustness. PatchCleanser performs two rounds of pixel masking (i.e., *double-masking)* on the input image to neutralize the effect of the adversarial patch (without knowing the location and content of the patch). We present our formulation of pixel masks in Section 3.1 and then discuss the details of double-masking algorithm in Section 3.2. We prove the robustness of our double-masking defense for certain images in Section 3.3. Finally, we discuss implementation details and present an end-to-end PatchCleanser defense pipeline in Section 3.4. We provide a summary of important notation in Table 1.

3.1 Pixel Mask Set

PatchCleanser aims to mask out the entire patch on the image and obtain accurate predictions from any state-of-the-art classification model. In this subsection, we introduce the concept of a mask set used in our masking operations.

Table 1: Summary of important notation

Notation	Description	Notation	Description
$\mathbb{F}$	Undefended model	$\bar{p}, \bar{p}_0 \times \bar{p}_1$	Estimated patch size
$\mathbf{x} \in \mathcal{X}$	Input image	$p, p_0 \times p_1$	Actual patch size
$y, \bar{y} \in \mathcal{Y}$	Class label	$k, k_0 \times k_1$	Budget of #masks
$\mathbf{m} \in \mathcal{M}$	Pixel mask	$m, m_0 \times m_1$	Mask size
$\mathbf{r} \in \mathcal{R}$	Patch region	$s, s_0 \times s_1$	Mask stride
$\mathcal{P} \subset \mathcal{M} \times \mathcal{Y}$	Masked prediction set	$n, n_0 \times n_1$	Image size

Mask set formulation. We represent each mask as a binary tensor $\mathbf{m} \in \{0,1\}^{W \times H}$ in the same shape as the $W \times H$ images; *the elements within the mask take values of* 0, and others are 1. We further denote a set of masks as $\mathcal{M}$ (these are similar to the definitions of $\mathbf{r}$ and $\mathcal{R}$). We require the mask set $\mathcal{M}$ to have the $\mathcal{R}$-covering property as defined below.

Definition 1 ($\mathcal{R}$-covering)**.** *A mask set $\mathcal{M}$ is $\mathcal{R}$-covering if, for any patch in the patch region set $\mathcal{R}$, at least one mask from the mask set $\mathcal{M}$ can cover the entire patch, i.e.,*

$$\forall\ \mathbf{r} \in \mathcal{R},\ \exists\ \mathbf{m} \in \mathcal{M}\ \ \text{s.t.}\ \ \mathbf{m}[i,j] \leq \mathbf{r}[i,j],\ \forall (i,j)$$

For a particular patch region set $\mathcal{R}$, there are multiple valid $\mathcal{R}$-covering mask sets $\mathcal{M}$ with a variable number of masks and different mask sizes/shapes. We will discuss a general approach for $\mathcal{R}$-covering mask set generation in Section 3.4. In the next subsection, we introduce how to perform our double-masking defense with a $\mathcal{R}$-covering mask set.

3.2 Double-masking for Robust Prediction

The double-masking algorithm is the core module of PatchCleanser; it performs two rounds of masking with an $\mathcal{R}$-covering mask set to robustly recover the correct prediction label. Our defense is based on the intuition that model predictions on images without adversarial pixels are generally correct and invariant to the masking operation: in Figure 1, we can visually recognize the dog even with one or two masks on the image.[4] In this subsection, we first introduce the high-level defense design and then explain the algorithm details.

First-round masking: detecting a prediction disagreement. Recall that Figure 1 gives an overview of our double-masking algorithm. In the first round of masking, we apply every mask $\mathbf{m}$ from the $\mathcal{R}$-covering mask set $\mathcal{M}$ to the input image and evaluate all *one-mask predictions* (left of Figure 1). In the clean setting, all one-mask predictions are likely to reach a unanimous agreement on the correct label, and we will output the agreed prediction (top of Figure 1). In the adversarial setting, at least one mask will remove all adversarial pixels; thus, at least one one-mask prediction is benign and likely to be correct (bottom left of Figure 1). In this case, we will detect a disagreement in one-mask predictions (benign

[4] A similar intuition is used in existing works [8, 18, 30], but we are the first to design a certifiably robust image classification defense without abstention.

versus malicious); we will then perform a second round of masking to settle this disagreement.

Second-round masking: settling the prediction disagreement. We first divide all one-mask prediction labels into two groups: the *majority prediction* (the prediction label with the highest occurrence) and the *disagreer predictions* (other labels that disagree with the majority). We need to decide which prediction label to trust (i.e., the majority or one of the disagreers). To solve this problem, we iterate over every *disagreer* prediction, get its corresponding first-round mask, and add a second mask from our mask set $\mathcal{M}$ to compute a set of *two-mask predictions* (right of Figure 1). If the first-round disagreer mask removes the patch, every second-round mask is applied to a "clean" image, and thus all two-mask predictions (evaluated with one first-round mask and different second-round masks) are likely to have a unanimous agreement. We can trust and return this agreed prediction. On the other hand, if the first-round disagreer mask does not remove the patch, the one-masked image is still "adversarial", and the second-round mask will cause a disagreement in two-mask predictions (when one of the second masks covers the patch). In this case, we discard this one-mask disagreer. Finally, if we try all one-mask disagreer predictions and no prediction label is returned, we trust and return the one-mask majority prediction as the default exit case.

Algorithm details. We provide the defense pseudocode in Algorithm 1. The defense takes an image $\mathbf{x}$, an undefended model $\mathbb{F}$, and an $\mathcal{R}$-covering mask set $\mathcal{M}$ as inputs and outputs a robust prediction $\bar{y}$. Line 2-5 illustrates the first-round masking; Line 6-11 demonstrates the second-round masking.

Details of first-round masking. In Algorithm 1, we first call the masking sub-procedure MASKPRED$(\cdot)$ using the mask set $\mathcal{M}$ (Line 2). The mask set $\mathcal{M}$ needs to ensure that at least one mask can remove the entire patch (i.e., $\mathcal{R}$-covering); we will discuss the mask set generation approach in Section 3.4.

In MASKPRED$(\cdot)$, we aim to collect all masked predictions and determine the majority prediction label (i.e., the label with the highest occurrence) as well as disagreer predictions (i.e., other predictions). We first generate a set $\mathcal{P}$ for holding all mask-prediction pairs. Next, for each mask $\mathbf{m}$ in the mask set $\mathcal{M}$, we evaluate the masked prediction via $\bar{y} \leftarrow \mathbb{F}(\mathbf{x} \odot \mathbf{m})$; here $\odot$ is the element-wise multiplication operator. We then add the mask-prediction pair $(\mathbf{m}, \bar{y})$ to the set $\mathcal{P}$. After gathering all masked predictions, we identify the label with the highest prediction occurrence as majority prediction $\bar{y}_{\text{maj}}$ (Line 20). Furthermore, we construct a disagreer prediction set $\mathcal{P}_{\text{dis}}$, whose elements are disagreer mask-prediction pairs (Line 21). Finally, we return the majority prediction label $\bar{y}_{\text{maj}}$ and the disagreer prediction set $\mathcal{P}_{\text{dis}}$.

After the first call of MASKPRED$(\cdot)$, we check if one-mask predictions reach a unanimous agreement (i.e., the disagreer prediction set $\mathcal{P}_{\text{dis}}$ is empty; Line 3). If $\mathcal{P}_{\text{dis}}$ is empty, we consider the input image likely as a clean image and return the agreed/majority prediction (*Case I: agreed prediction*; Line 4).

Algorithm 1 Double-masking defense of PatchCleanser

Input: Image $\mathbf{x}$, vanilla prediction model $\mathbb{F}$, mask set $\mathcal{M}$
Output: Robust prediction $\bar{y}$

1: **procedure** DOUBLEMASKING($\mathbf{x}, \mathbb{F}, \mathcal{M}$)
2: $\quad \bar{y}_{\text{maj}}, \mathcal{P}_{\text{dis}} \leftarrow$ MASKPRED$(\mathbf{x}, \mathbb{F}, \mathcal{M})$ ▷ First-rnd. mask
3: $\quad$ **if** $\mathcal{P}_{\text{dis}} = \varnothing$ **then**
4: $\quad\quad$ **return** $\bar{y}_{\text{maj}}$ ▷ *Case I: agreed prediction*
5: $\quad$ **end if**
6: $\quad$ **for** each $(\mathbf{m}_{\text{dis}}, \bar{y}_{\text{dis}}) \in \mathcal{P}_{\text{dis}}$ **do** ▷ Second-rnd. mask
7: $\quad\quad \bar{y}', \mathcal{P}' \leftarrow$ MASKPRED$(\mathbf{x} \odot \mathbf{m}_{\text{dis}}, \mathbb{F}, \mathcal{M})$
8: $\quad\quad$ **if** $\mathcal{P}' = \varnothing$ **then**
9: $\quad\quad\quad$ **return** $\bar{y}_{\text{dis}}$ ▷ *Case II: disagreer prediction*
10: $\quad\quad$ **end if**
11: $\quad$ **end for**
12: $\quad$ **return** $\bar{y}_{\text{maj}}$ ▷ *Case III: majority prediction*
13: **end procedure**

14: **procedure** MASKPRED($\mathbf{x}, \mathbb{F}, \mathcal{M}$)
15: $\quad \mathcal{P} \leftarrow \varnothing$ ▷ A set for mask-prediction pairs
16: $\quad$ **for** $\mathbf{m} \in \mathcal{M}$ **do** ▷ Enumerate every mask $\mathbf{m}$
17: $\quad\quad \bar{y} \leftarrow \mathbb{F}(\mathbf{x} \odot \mathbf{m})$ ▷ Evaluate masked prediction
18: $\quad\quad \mathcal{P} \leftarrow \mathcal{P} \bigcup \{(\mathbf{m}, \bar{y})\}$ ▷ Update set $\mathcal{P}$
19: $\quad$ **end for**
20: $\quad \bar{y}_{\text{maj}} \leftarrow \arg\max_{y^*} |\{(\mathbf{m}, \bar{y}) \in \mathcal{P} \mid \bar{y} = y^*\}|$ ▷ Majority
21: $\quad \mathcal{P}_{\text{dis}} \leftarrow \{(\mathbf{m}, \bar{y}) \in \mathcal{P} \mid \bar{y} \neq \bar{y}_{\text{maj}}\}$ ▷ Disagreers
22: $\quad$ **return** $\bar{y}_{\text{maj}}, \mathcal{P}_{\text{dis}}$
23: **end procedure**

On the other hand, a non-empty disagreer set implies a first-round prediction disagreement, and the algorithm proceeds to the second-round masking to settle the disagreement.

Details of second-round masking. The pseudocode of the second-round masking is in Line 6-11. We will look into *every* one-mask disagreer prediction in $\mathcal{P}_{\text{dis}}$ (Line 6). For each $(\mathbf{m}_{\text{dis}}, \bar{y}_{\text{dis}})$, we apply the disagreer mask $\mathbf{m}_{\text{dis}}$ to the image and feed the masked image $\mathbf{x} \odot \mathbf{m}_{\text{dis}}$ to the masking sub-procedure MASKPRED$(\cdot)$ for the second-round masking (Line 7). If all two-mask predictions reach a unanimous agreement (i.e., $\mathcal{P}' = \varnothing$), we consider that the first-round mask $\mathbf{m}_{\text{dis}}$ has already removed the adversarial perturbations. Our algorithm returns this one-mask disagreer prediction (*Case II: disagreer prediction*; Line 9). On the other hand, if two-mask predictions disagree (i.e., $\mathcal{P}' \neq \varnothing$), we consider that the disagreer mask $\mathbf{m}_{\text{dis}}$ has not removed the patch. In this case, we discard this one-mask disagreer prediction and move to the next one. In the end, if we return no prediction in the second-round masking, we trust and return the one-mask majority prediction $\bar{y}_{\text{maj}}$ (*Case III: majority prediction*; Line 12).

Remark: defense complexity. When the number of disagreer predictions is bounded by a small constant C (which is the usual case in the clean setting), the defense complexity is $O(|\mathcal{M}| + C \cdot |\mathcal{M}|)$. However, its worst-case complexity is $O(|\mathcal{M}|^2)$ (doing all two-mask predictions). In Appendix C,

we discuss and evaluate another defense algorithm that has the same robustness guarantees and a better worst-case inference complexity $O(|\mathcal{M}|)$, at the cost of a drop in clean accuracy.

3.3 Robustness Certification for Double-Masking Defense

In this subsection, we discuss how to certify the robustness of our double-masking algorithm for a given image. Recall that we say our defense is certifiably robust for a given image if our model prediction is always correct against any adaptive white-box attacker within our threat model $\mathcal{A}_{\mathcal{R}}$. The certification only applies to labeled images since we need ground-truth labels to check the prediction correctness.

First, we define a concept of *two-mask correctness*, which we claim is a sufficient condition for certified robustness.

Definition 2 (two-mask correctness)**.** *A model $\mathbb{F}$ has two-mask correctness for a mask set $\mathcal{M}$ and a clean image data point $(\mathbf{x}, y)$, if model predictions on all possible two-masked images are correct, i.e.,*

$$\mathbb{F}(\mathbf{x} \odot \mathbf{m}_0 \odot \mathbf{m}_1) = y \ , \forall\, \mathbf{m}_0 \in \mathcal{M}, \forall\, \mathbf{m}_1 \in \mathcal{M}$$

Next, we present our theorem stating that two-mask correctness for a clean image (and an $\mathcal{R}$-covering mask set) implies the certifiable robustness of our defense to adversarial patches (constrained by $\mathcal{A}_{\mathcal{P}}$) on that image.

Theorem 1. *Given a clean data point $(\mathbf{x}, y)$, a classification model $\mathbb{F}$, a mask set $\mathcal{M}$, and the threat model $\mathcal{A}_{\mathcal{R}}$, if $\mathcal{M}$ is $\mathcal{R}$-covering and $\mathbb{F}$ has two-mask correctness for $\mathcal{M}$ and $(\mathbf{x}, y)$, then our double-masking defense (Algorithm 1) will always return a correct label, i.e.,* $\textsc{DoubleMasking}(\mathbf{x}', \mathbb{F}, \mathcal{M}) = y,\ \forall\, \mathbf{x}' \in \mathcal{A}_{\mathcal{R}}(\mathbf{x})$.

Proof. First, we present three useful claims for our proof.

Claim. *Given the same conditions of Theorem 1 ($\mathcal{R}$-covering and two-mask correctness), we have:*

1. *There is at least one correct one-mask prediction in the first-round masking (Line 2 of Algorithm 1).*
2. *There is at least one correct two-mask prediction in every iteration of the second-round masking (Line 7 of Algorithm 1).*
3. *If a first-round mask removes the patch, then all its second-round two-mask predictions (Line 7 of Algorithm 1) are correct.*

Proof. The proof of three claims follows from the definitions of $\mathcal{R}$-covering and two-mask correctness.

1. The first claim holds since at least one first-round mask removes the patch (due to $\mathcal{R}$-covering) and recovers the correct prediction (due to two-mask correctness; note that two-mask correctness reduces to "one-mask" correctness when two masks are at the same locations).
2. The second claim holds since at least one second-round mask removes the patch (due to $\mathcal{R}$-covering) and recovers the correct prediction (due to two-mask correctness).
3. The third claim holds since all the two-mask predictions are benign and correct when the patch is removed by the first-round mask (due to two-mask correctness). □

Next, we use these three claims to prove three lemmas, which together show that our double-masking algorithm (Algorithm 1) will never return an incorrect prediction (given $\mathcal{R}$-covering and two-mask correctness). *All lemmas are under the same conditions of Theorem 1.*

Lemma 1. *Algorithm 1 will never return an incorrect label via Case I (Line 4 of Algorithm 1).*

Proof. If Algorithm 1 returns an incorrect label ($\bar{y}_{\text{maj}} \neq y$) via *Case I*, it means that $\mathcal{P}_{\text{dis}} = \varnothing$ and all one-mask predictions in the first-round masking are incorrect as $\bar{y}_{\text{maj}}$. This leads to a contradiction because our first claim indicates that at least one one-mask prediction is correct. □

Lemma 2. *Algorithm 1 will never return an incorrect label via Case II (Line 9 of Algorithm 1).*

Proof. If Algorithm 1 returns an incorrect label ($\bar{y}_{\text{dis}} \neq y$) via *Case II*, it means that $\mathcal{P}' = \varnothing$ and all two-mask predictions for the first-round disagreer $(\mathbf{m}_{\text{dis}}, \bar{y}_{\text{dis}})$ in the second-round masking are incorrect as $\bar{y}_{\text{dis}}$. This leads to a contradiction because our second claim indicates that at least one two-mask prediction is correct in any iteration of the second-round masking. □

Lemma 3. *Algorithm 1 will never return an incorrect label via Case III (Line 12 of Algorithm 1).*

Proof. If Algorithm 1 returns an incorrect label via *Case III*, we have $\bar{y}_{\text{maj}} \neq y$. This implies that the correct label y is a disagreer (recall that at least one first-round mask removes the patch and gives the correct one-mask prediction label y). From our third claim, we know that for this one-mask disagreer (whose first-round mask removes the patch), all its two-mask predictions are correct due to two-mask correctness. Therefore, we have $\mathcal{P}' = \varnothing$, and Algorithm 1 will return this disagreer $\bar{y}_{\text{dis}} = y$ via *Case II*. This contradicts with Algorithm 1 returning a label via *Case III*. □

Putting things together, we prove in the above three lemmas that our double-masking algorithm (Algorithm 1) will never return an incorrect label. Since Algorithm 1 will always return a prediction label, we have proved that our defense will always return a correct label under the conditions of Theorem 1. □

Robustness certification procedure. From Theorem 1, we can certify the robustness of our defense on a clean/test image by checking if our model has two-mask correctness on that image. The pseudocode for our robust certification is

Algorithm 2 Robustness certification for PatchCleanser

Input: Image $\mathbf{x}$, ground-truth label y, vanilla prediction model $\mathbb{F}$, mask set $\mathcal{M}$, threat model $\mathcal{A}_{\mathcal{R}}$
Output: Whether $\mathbf{x}$ has certified robustness

1: **procedure** CERTIFICATION($\mathbf{x}, y, \mathbb{F}, \mathcal{M}, \mathcal{A}_{\mathcal{R}}$)
2: **if** $\mathcal{M}$ is not $\mathcal{R}$-covering **then** ▷ Insecure mask set
3: **return** `False`
4: **end if**
5: **for** every $(\mathbf{m}_0, \mathbf{m}_1) \in \mathcal{M} \times \mathcal{M}$ **do**
6: $\bar{y}' \leftarrow \mathbb{F}(\mathbf{x} \odot \mathbf{m}_0 \odot \mathbf{m}_1)$ ▷ Two-mask pred.
7: **if** $\bar{y}' \neq y$ **then**
8: **return** `False` ▷ Possibly vulnerable
9: **end if**
10: **end for**
11: **return** `True` ▷ Certified robustness!
12: **end procedure**

presented in Algorithm 2. First, the certification procedure checks if mask set $\mathcal{M}$ is $\mathcal{R}$-covering (Line 2). Next, it evaluates all possible two-mask predictions (Line 5-10). If any of the two-mask predictions is incorrect, the algorithm returns `False` (possibly vulnerable). On the other hand, if all two-mask predictions match the ground-truth y, we have certified robustness for this image, and the algorithm returns `True`.

In our evaluation (Section 4), we will apply Algorithm 2 to labeled datasets and report *certified robust accuracy*, the fraction of labeled test images for which Algorithm 2 returns `True`. We note that Theorem 1 ensures that this certified accuracy is the *lower bound* of model accuracy against any adaptive attacker within the threat model $\mathcal{A}_{\mathcal{R}}$. For example, a certified robust accuracy of 62.1% on the ImageNet [11] dataset implies that PatchCleanser can correctly classify 62.1% of the ImageNet test images, no matter how an adaptive attacker (within the threat model) generates and places the patch.

3.4 Implementation of the End-to-End PatchCleanser Defense

In this subsection, we first present an adaptive $\mathcal{R}$-covering mask set generation technique and then provide a complete view of our end-to-end PatchCleanser defense.

Adaptive mask set generation. In practice, a defense needs to operate within the constraints of available computational resources. This imposes a bound on the number of masked model predictions that we can evaluate.[5] Therefore, we need to carefully generate a mask set that meets the computation budget (i.e., the number of masks) while maintaining the security guarantee (i.e., $\mathcal{R}$-covering). We first present our approach for 1-D "images" (Figure 3 provides two visual examples) and then generalize it to 2-D.

[5] For example, there are 40k possible locations for a 24×24 mask on a 224×224 image, and it is computationally expensive to evaluate all 40k masked predictions.

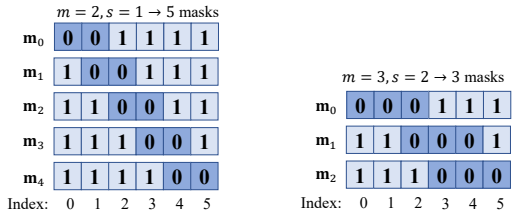

Figure 3: Visual examples for 1-D mask set generation. Both mask sets are $\mathcal{R}$-covering for a patch of estimated size $\bar{p} = 2$ on an image of size $n = 6$. *Left example:* smaller mask size $m = 2$, smaller mask stride $s = 1$, and larger mask set $I = \{0,1,2,3,4\}$. *Right example:* larger mask size $m = 3$, larger mask stride $s = 2$, and smaller mask set $I = \{0,2,3\}$.

Adjusting the mask set size. We generate a mask set via moving a mask over the input image. Consider a mask of width m over an image of size n, we first place the mask at the coordinate 0 (so that the mask covers the indices from 0 to $m-1$). Next, we move the mask with a stride of s across the image and gather a set of mask locations $\{0, s, 2s, \cdots, \lfloor \frac{n-m}{s} \rfloor s\}$. Finally, we place the last mask at the index of $n-m$ in case the mask at $\lfloor \frac{n-m}{s} \rfloor s$ cannot cover the last m pixels. We can define a mask set $\mathcal{M}_{m,s,n}$ as:

$$\mathcal{M}_{m,s,n} = \{\mathbf{m} \in \{0,1\}^n \mid \mathbf{m}[u] = 0, u \in [i, i+m); \\ \mathbf{m}[u] = 1, u \notin [i, i+m); i \in I\}$$
$$I = \{0, s, 2s, \cdots, \lfloor \frac{n-m}{s} \rfloor s\} \bigcup \{n-m\} \quad (1)$$

Furthermore, we can compute the mask set size as:

$$|\mathcal{M}_{m,s,n}| = |I| = \lceil \frac{n-m}{s} \rceil + 1 \quad (2)$$

This equation shows that we can adjust the mask set size via the mask stride s. In the example of Figure 3, we can reduce the mask set size from 5 to 3 by increasing the mask stride from 1 to 2. Next, we discuss the security property (i.e., $\mathcal{R}$-covering) of the mask set $\mathcal{M}$.

Ensuring the security guarantee. Using a large mask stride might leave "gaps" between two adjacent masks; therefore, we need to choose a proper mask size to cover these gaps to ensure that the mask set $\mathcal{M}$ is $\mathcal{R}$-covering. We present the following lemma discussing the $\mathcal{R}$-covering property of the mask set $\mathcal{M}_{m,s,n}$; we will present its proof in Appendix E.

Lemma 4. *The mask set $\mathcal{M}_{m,s,n}$ is $\mathcal{R}$-covering for a patch that is no larger than $p^* = m - s + 1$.*

Lemma 4 indicates that the mask size needs to be no smaller than $m^* = p + s - 1$ to ensure $\mathcal{R}$-covering.

Mask set generation. Armed with the ability to adjust the mask set size and to ensure the security guarantee (as discussed above), we now present our complete mask set generation approach. The procedure takes as inputs *the computation*

budget k (i.e., the number of masks), the *estimated* patch size $\bar{p}$ (i.e., the security parameter), and the image size n, and aims to generate a $\mathcal{R}$-covering set that satisfies the computational budget k. First, based on the the inputs $k, \bar{p}, n$, we derive mask stride s and mask size m using Lemma 4 ($\bar{p} = m - s + 1$) and Equation 2 ($k = \lceil \frac{n-m}{s} \rceil + 1$) as follows:

$$s = \lceil \frac{n - \bar{p} + 1}{k} \rceil$$
$$m = \bar{p} + s - 1 \tag{3}$$

Next, we generate the set $\mathcal{M}_{m,s,n}$ via Equation 1 accordingly. We note that when we have a different estimation for the patch size $\bar{p}$, we only need to adjust the mask stride s and mask size m according to Equation 3 while keeping the number of masks k unchanged.

Generalizing to 2-D images. We can easily generalize the 1-D mask set to 2-D by separately applying Equation 1 and Equation 3 to each of the two axes of the image. For $n_0 \times n_1$ images, $\bar{p}_0 \times \bar{p}_1$ patches, $k_0 \times k_1$ number of masks, we can calculate s_0, s_1, m_0, m_1 with Equation 3 and obtain I_0, I_1 with Equation 1. The mask set generation becomes $\mathcal{M}_{(m_0,m_1),(s_0,s_1),(n_0,n_1)} = \{\mathbf{m} \in \{0,1\}^{n_0 \times n_1} \mid \mathbf{m}[u,v] = 0, u \in [i, i+m_0), v \in [j, j+m_1), (i,j) \in I_0 \times I_1;\ \mathbf{m}[u,v] = 1, \text{otherwise}\}$.

Remark: trade-off between efficiency and accuracy. As shown in Equation 3, if we want to improve the efficiency (by having a smaller k), we will have to use a larger stride s and larger mask size m. Intuitively, the model prediction is less accurate for a larger mask; thus, the improvement in efficiency can be at the cost of model accuracy. Our mask set generation approach allows us to balance this trade-off between efficiency and accuracy in the real-world deployment. We will study this trade-off in Section 4.3.

End-to-end PatchCleanser pipeline. With the mask set generation technique, we can summarize the end-to-end PatchCleanser pipeline as follows:

1. First, we perform *adaptive mask set generation* to obtain a secure $\mathcal{R}$-covering mask set $\mathcal{M}$ that satisfies a certain computation budget (number of masks $k_0 \times k_1$).
2. Second, we perform *double-masking* (Algorithm 1) with the model $\mathbb{F}$ and the mask set $\mathcal{M}$ for robust classification.
3. Third, we can use our certification procedure (Algorithm 2) to certify the robustness of PatchCleanser on a given labeled image against any adaptive white-box attacker within the threat model $\mathcal{A}_{\mathcal{R}}$. We will evaluate the fraction of labeled test images that can be certified across multiple datasets in the next section.

4 Evaluation

We instantiate PatchCleanser with three different classification models, and extensively evaluate the defense using three different datasets. We will demonstrate state-of-the-art clean accuracy and certified robust accuracy of PatchCleanser compared with prior works [7, 25, 33, 55, 61] and provide detailed analysis of our defense under different settings.

In this section, we primarily focus on a single square patch that can have arbitrary content and that can be anywhere on the image. This setting is currently an open research question in the field, and also allows for a fair comparison with prior works [7, 25, 33, 55, 61]. We will show the flexibility of PatchCleanser by demonstrating its generalization to a set of different patch shapes and multiple patches in Section 5.1.

4.1 Setup

In this subsection, we briefly introduce our evaluation setup. We provide additional details in our technical report [56]. Our source code is available at https://github.com/inspire-group/PatchCleanser.

Datasets. We choose three popular image classification benchmark datasets for evaluation: ImageNet [11], ImageNette [14], CIFAR-10 [23].

ImageNet and ImageNette. ImageNet [11] is a challenging image classification dataset which has 1.3M training images and 50k validation images from 1000 classes. ImageNette [14] is a 10-class subset of ImageNet with 9469 training images and 3925 validation images. ImageNet/ImageNette images have a high resolution, and we resize and crop them to 224×224 before feeding them to different models.

CIFAR-10. CIFAR-10 [23] is a benchmark dataset for low-resolution image classification. CIFAR-10 has 50k training images and 10k test images from 10 classes. Each image is in the resolution of 32×32. We resize them to 224×224 via bicubic interpolation for a better classification performance.

Models. We choose three representative image classification models to build PatchCleanser. We provide model training details in Appendix A.

ResNet. ResNet [19] is a classic Convolutional Neural Network (CNN) model. It uses layers of convolution filters and residual blocks to extract features for image classification. We use ResNetV2-50x1 and its publicly available weights trained for ImageNet [11]. We finetune the model for other different datasets used in our evaluation.

Vision Transformer (ViT). ViT [13] is adapted from NLP Transformer [50] for the image classification task. It divides an image into disjoint pixel blocks, and uses self-attention architecture to extract features across different pixel blocks for classification. We use ViT-B16-224 [13] trained for ImageNet and finetune it on other datasets.

Multi-layer Perceptron (MLP). There have been recent advances in leveraging MLP-only architectures for image classification (e.g., MLP-mixer [47], ResMLP [48]). These architectures take pixel blocks as input and "mix" features/pixels across locations and channels for predictions. We choose

Table 2: Clean accuracy and certified robust accuracy for different defenses and datasets[†]

Dataset	ImageNette [14]						ImageNet [11]						CIFAR-10 [23]			
Patch size	1% pixels		2% pixels		3% pixels		1% pixels		2% pixels		3% pixels		0.4% pixels		2.4% pixels	
Accuracy (%)	clean	robust	clean	robust	clean	robust	clean	robust	clean	robust	clean	robust	clean	robust	clean	robust
PC-ResNet	**99.6**	96.4	**99.6**	94.4	**99.5**	93.5	81.7	58.4	81.6	53.0	81.4	50.0	98.0	88.5	97.8	78.8
PC-ViT	**99.6**	**97.5**	**99.6**	**96.4**	**99.5**	**95.3**	**84.1**	**66.4**	**83.9**	**62.1**	**83.8**	**59.0**	**99.0**	**94.3**	**98.7**	**89.1**
PC-MLP	99.4	96.8	99.3	95.3	99.4	94.6	79.6	58.4	79.4	53.8	79.3	50.7	97.4	86.1	97.0	78.0
IBP [7]	computationally infeasible												65.8	51.9	47.8	30.8
CBN [61]	94.9	74.6	94.9	60.9	**94.9**	45.9	49.5	13.4	49.5	7.1	49.5	3.1	84.2	44.2	84.2	9.3
DS [25]	92.1	82.3	92.1	79.1	92.1	75.7	44.4	17.7	44.4	14.0	44.4	11.2	83.9	68.9	83.9	56.2
PG-BN [55]	**95.2**	**89.0**	**95.0**	**86.7**	94.8	**83.0**	**55.1**	**32.3**	**54.6**	**26.0**	**54.1**	**19.7**	84.5	63.8	83.9	47.3
PG-DS [55]	92.3	83.1	92.1	79.9	92.1	76.8	44.1	19.7	43.6	15.7	43.0	12.5	84.7	69.2	84.6	57.7
BagCert[‡][33]	–	–	–	–	–	–	45.3	27.8	45.3	22.7	45.3	18.0	**86.0**	**72.9**	**86.0**	**60.0**

[†] We mark the best result for PatchCleanser models and the best result for prior works in bold.
[‡] The BagCert numbers are provided by the authors [33] through personal communication since the source code is unavailable; results for ImageNette are not provided.

ResMLP-S24-224 [48] in our evaluation. We take the pre-trained model for ImageNet, and finetune it for other datasets.

Adversarial patches. Following prior works [7, 25, 33, 55], we report defense performance against a square patch that takes 1%, 2%, and 3% of input image pixels for ImageNet/ImageNette and a square patch with 0.4% and 2.4% pixels for CIFAR-10 images. We allow these patches to have *arbitrary* content and be *anywhere* on the image. In Section 4.3, we also report results for larger patch sizes (ranging from 2% to 62% image pixels) to understand the limit of our defense. In Section 5.1, we quantitatively discuss the implications of using a set of different rectangle patch shapes as well as multiple patches.

Defenses. We build three defense instances PC-ResNet, PC-ViT, PC-MLP using three vanilla models of ResNet, ViT, and MLP. In Section 4.3, we will analyze the effect of different defense setups (i.e., the number of masks). In our default setup, we set the number of masks $k_0 \times k_1 = k^2 = 6 \times 6$, which has high certified robustness and moderate computational overhead. We then generate the $\mathcal{R}$-covering mask set $\mathcal{M}$ as discussed in Section 3.4.

We also report defense performance of prior works Interval Bound Propagation based defense (IBP) [7], Clipped BagNet (CBN) [61], De-randomized Smoothing (DS) [25], PatchGuard (PG) [55], and BagCert [33] for comparison. We use the optimal defense settings stated in their respective papers.

Evaluation Metrics. We report clean accuracy and certified robust accuracy as our evaluation metrics. The *clean accuracy* is defined as the fraction of clean test images that can be correctly classified by our defended model. The *certified robust accuracy* is the fraction of test images for which Algorithm 2 returns `True` (certifies the robustness for this image), i.e., no adaptive white-box attacker can bypass our defense. In addition to accuracy metrics, we also use *per-example inference time* to analyze the computational overhead.

Remark: no need to implement adaptive attacks. As discussed in Section 3.3, certified robust accuracy is the lower bound of model accuracy against any adaptive attacker within the threat model. Therefore, it is not necessary to empirically evaluate robustness using any concrete adaptive attack strategy: empirical robust accuracy is always higher than certified accuracy.

Table 3: Clean accuracy of vanilla models

	ImageNette	ImageNet	CIFAR-10
ResNet [19]	99.8%	82.3%	98.3%
ViT [13]	99.8%	84.8%	99.0%
MLP [48]	99.5%	80.2%	97.8%

4.2 State-of-the-art Clean Accuracy and Certified Robust Accuracy across All Datasets

We report our main evaluation results for PatchCleanser in Table 2 and compare defense performance with prior works.

State-of-the-art clean accuracy. As shown in Table 2, PatchCleanser achieves high clean accuracy. Take PC-ViT as an example, PatchCleanser achieves 99.5+% clean accuracy for 10-class ImageNette, 83.8+% for 1000-class ImageNet, and 98.7+% for CIFAR-10. We further report the accuracy of state-of-the-art vanilla classification models in Table 3. From these two tables, we can see that the clean accuracy of our defense is very close to the state-of-the-art undefended models (the difference is smaller than 1%). The high clean accuracy can foster real-world deployment of our defense.

High certified robustness. In addition to state-of-the-art clean accuracy achieved by our defense, we can see from Table 2 that PatchCleanser also has very high certified robust accuracy. For ImageNette, our PC-ViT has a certified robust accuracy of 97.5% against a 1% square patch. *That is, for 97.5% of the test images, no strong adaptive white-box attacker who uses a 1%-pixel square patch anywhere on the image can induce misclassification of our defended model.* Furthermore, we can also see high certified robust accuracy for ImageNet and CIFAR-10, e.g., 66.4% certified robust accuracy for a 1%-pixel patch on ImageNet and 94.3% certified

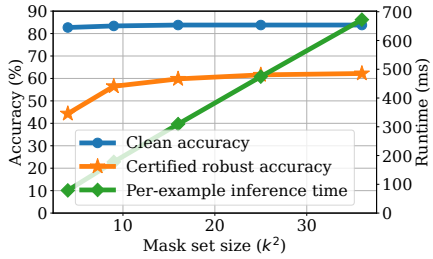

Figure 4: The effect of mask set size on defense performance (ImageNet)

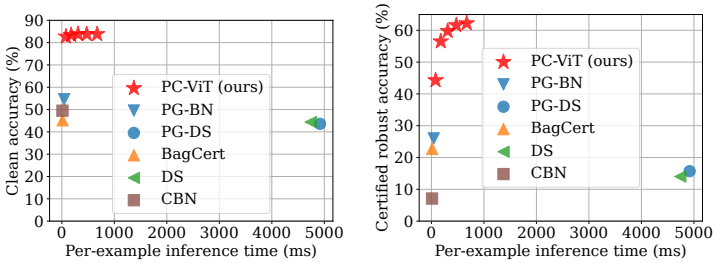

Figure 5: Trade-off between overhead and accuracy on ImageNet (left: clean accuracy; right: certified robust accuracy)

robust accuracy for a 0.4%-pixel patch on CIFAR-10.

Significant improvements in clean accuracy and certified robust accuracy from prior works. We compare our defense performance with all prior certifiably robust defenses. From Table 2, we can see that all our defense instances (i.e., PC-ResNet, PC-ViT, and PC-MLP) significantly outperform all prior works in terms of both clean accuracy and certified robust accuracy. Notably, for a 2%-pixel patch on ImageNet, PC-ViT improves the clean accuracy from 54.6% to 83.9% (29.3% gain in top-1 accuracy) and boosts the certified robust accuracy from 26.0% to 62.1% (the accuracy gain is 36.1%; the improvement is more than 2 times). Moreover, we can see that *the certified robust accuracy of PC-ViT is even higher than the clean accuracy of all prior works*. These significant improvements are due to PatchCleanser's compatibility with state-of-the-art classification models, while previous works are fundamentally incompatible with them (recall that PG [55], DS [25], BagCert [33], CBN [61] are all limited to models with a small receptive field).

We can also see large improvements across datasets including ImageNette and CIFAR-10. For a 2%-pixel patch on ImageNette, PC-ViT improves clean accuracy from 95.0% to 99.6% and certified robust accuracy from 86.7% to 96.4%. For a 2.4%-pixel patch on CIFAR-10, PC-ViT improves clean accuracy from 86.0% to 98.7% (12.7% gain) and certified robust accuracy from 60.0% to 89.1% (29.1% gain).

Takeaways. In this subsection, we demonstrate that PatchCleanser has similar clean accuracy as vanilla state-of-the-art models, as well as high certified robust accuracy. In comparison with prior certifiably robust defenses, we demonstrate significant improvements in both clean accuracy and certified robust accuracy. These improvements showcase the strength of defenses that are compatible with any state-of-the-art model.

4.3 Detailed Analysis of PatchCleanser

In this subsection, we provide a detailed analysis of PatchCleanser models. We will discuss the trade-off between defense performance and defense overhead, study the implications of over-estimated patch sizes, and finally explore the limit of PatchCleanser against larger patches.

There is a trade-off between defense performance and defense overhead (balanced by the number of masks). In this analysis, we use PC-ViT against a 2%-pixel patch on 5000 randomly selected ImageNet test images to study the trade-off between defense performance and defense overhead. In Figure 4, we report the clean accuracy, certified robust accuracy, and per-example inference time (evaluated using a batch size of one) for PC-ViT configured with different computation budgets (i.e., number of masks k^2). As shown in the figure, as we increase the number of masks, the certified robust accuracy first significantly improves and then gradually saturates. This is because a larger k^2 gives a smaller mask stride and leads to a smaller mask size, which enhances the robustness certification. However, we also observe that the per-example inference time greatly increases as we are using a larger number of masks. Therefore, we need to carefully choose a proper mask set size to balance the trade-off between defense performance and defense overhead. In our default setting, we prioritize the defense performance and use a mask set size of $6^2 = 36$.

We further visualize the defense overhead and defense performance (in terms of clean accuracy and certified robust accuracy) for different defenses in Figure 5. As shown in the figure, CBN [61] (12.0ms), PG-BN [55] (44.2ms), and BagCert [33] (14.0ms) have a very small runtime since they only requires one-time model feed-forward inference. For PC-ViT, we report the performance trade-off under different mask set sizes ranging from 4 to 36 (we omit PC-ResNet and PC-MLP for simplicity). We can see that when PC-ViT is optimized for classification accuracy, we have 83.8% clean accuracy and 62.2% certified robust accuracy with a moderate defense overhead (672.4ms). On the other hand, when PC-ViT is optimized for defense efficiency, we achieve a small per-example inference time (78.8ms) while still significantly outperforming prior works in terms of clean accuracy (82.7%) and certified robust accuracy (44.3%). Furthermore, we note that prior works such as DS [25] and PG-DS [55] have a much larger defense overhead on ImageNet (4740.0 ms and 4918.0ms, respectively).

From this analysis, we demonstrate that there is a trade-off between defense strength and defense efficiency. In PatchCleanser, we can tune mask set size to balance this trade-off. In contrast, while prior works like PG-BN [55] and

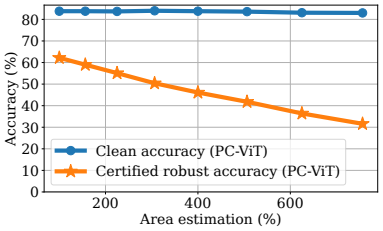

Figure 6: Effect of over-estimated patch size for a 32×32 patch (ImageNet)

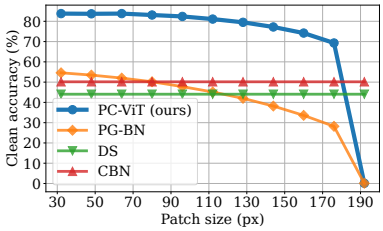

Figure 7: Clean accuracy for defense setups against different patch sizes (ImageNet)

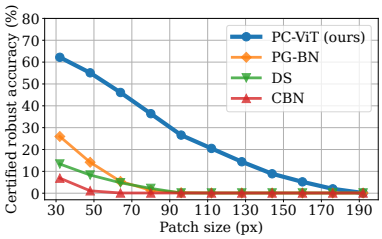

Figure 8: Certified robust accuracy against different patch sizes (ImageNet)

BagCert [33] have a smaller inference time, they cannot further improve their defense performance regardless of additionally available computation resources. Finally, we argue that our defense can be applied to time-sensitive applications like video analysis by performing the defense on a subset of frames. We also note that we can significantly reduce the empirical inference time by running the masked prediction evaluation (i.e., MASKPRED$(\cdot)$ in Algorithm 1) in parallel when multiple GPUs are available. With the improvement in computation resources and the development of high-performance lightweight models, we expect the computational cost to be mitigated in the future.

Over-estimation of patch sizes has a small impact on the defense performance. PatchCleanser requires a conservative estimation of the patch size (as a security parameter) to generate a proper mask set (the dependence on patch size estimation is similar to several prior works [30,55,58]). In this analysis, we aim to study the defense performance when we over-estimate the patch size. In Figure 6, we plot the defense performance as a function of an estimated patch area (i.e., the number of pixels) on the ImageNet dataset (the actual patch has 32×32 pixels on the 224×224 image)). The x-axis denotes the ratio of the estimated patch area to the actual patch area; 100% implies no over-estimation. As shown in the figure, as the over-estimation becomes greater, the clean accuracy of PC-ViT is barely affected while the certified robust accuracy gradually drops. We note that even when the estimation of the patch area is conservatively set to 4 times the actual area of the patch, PC-ViT still significantly outperforms all prior works.

Understanding the limit of our defense with larger patch sizes. In Figure 7 and 8, we report the defense performance of PC-ViT against different patch sizes on the 224×224 ImageNet test images. This analysis helps us to understand the limit of PatchCleanser when facing extremely large adversarial patches. Figure 7 shows that, as we increase the patch size, the clean accuracy of PC-ViT slowly decreases. For example, even when the patch size is 112×112 (on the 224×224 image), the clean accuracy is still above 80%. The clean accuracy finally deteriorates to 0.1% (random guess) when the patch is extremely large as 192×192. Figure 8 shows that the certified robust accuracy also decreases when a larger patch is used. When a large patch of 64×64 is used, we have 46.1% certified robust accuracy; when the patch is as large as 112×112 (half of the image size), we still have a non-trivial top-1 certified robust accuracy of 20.5% for 1000-class classification. We note that we use a fixed number of masks ($k^2 = 36$) for this analysis; this shows that PatchCleanser performs well across different patch sizes when having a fixed computational budget.

We further plot the clean accuracy and certified accuracy of prior defenses [25, 55, 61] in Figure 7 and 8. We can see that the certified robust accuracy of prior works drops quickly to zero when we consider a larger patch, while PatchCleanser achieves a much higher robust accuracy across all patch sizes. We note that the clean accuracy of CBN [61] and DS [25] does not change due to their fixed defense parameters. When the certified robust accuracy of DS and CBN reduces to zero (at a patch size of 96px), PatchCleanser still has a much higher clean accuracy and certified robust accuracy.

5 Discussion

In this section, we quantitatively discuss the implications of multiple patch shapes and multiple patches, the Minority Reports defense [30], limitations and future work directions.

5.1 PatchCleanser against Multiple Patch Shapes and Multiple Patches

In Section 4, we primarily focus on the scenario of one *square* patch. In this subsection, we further demonstrate the compatibility of our defense with (1) a set of different patch shapes as well as (2) multiple patches.

Intuition. The key requirement of PatchCleanser is using an $\mathcal{R}$-covering mask set $\mathcal{M}$. Therefore, to counter an attacker who can use a patch from a set of different patch shapes or who can use multiple patches, we only need to consider a mask set that includes masks of different shapes or multiple masks to ensure $\mathcal{R}$-covering and plug the mask set into our double-masking algorithm.

Different patch shapes. First, we consider a scenario where an attacker can use *any* rectangle shape that cov-

Table 4: Defense against different patch shapes (*all possible rectangle shapes that consist of 1% image pixels*) and multiple patches (*two 1%-pixel square patches*).

Dataset	ImageNette		ImageNet		CIFAR-10	
Accuracy (%)	clean	robust	clean	robust	clean	robust
Any 1% rectangle	99.2	91.8	85.4	49.8	99.4	82.6
1% square (baseline)	99.6	96.6	84.2	68.2	99.0	92.6
Two 1% square patches	98.8	89.2	83.8	45.8	98.6	76.6
One 2% square patch	99.2	95.6	83.8	63.2	99.0	91.2

ers at most 1% pixels of the 224×224 image (502 pixels), which includes thousands of shapes ranging from 1×224 to 22×22. To counter this strong attacker, we consider a shape set $\mathcal{S} = \{5\times224, 12\times83, 23\times38, 39\times20, 84\times12, 224\times5\}$; we claim that 6 shapes in $\mathcal{S}$ together can *cover any 1%-pixel rectangle shape* (more details in Appendix B). We then generate masks for 6 different rectangles and use these masks in the double-masking algorithm. We report the defense performance in the upper half of Table 4, which shows that our defense has high clean performance and certified robust accuracy. We note that the reported certified robust accuracy accounts for a much stronger attacker that can use any rectangle shape covering at most 1% pixels, explaining a drop in certified robustness from the baseline. Nevertheless, our defense performance (while considering a stronger adversary) is still much better than those of prior works against a 1%-pixel square patch (recall Table 2).

Multiple patches. To handle multiple (K) patches, we can generate a mask set that includes all possible K-mask combinations; the certification needs to check $2K$-mask correctness (more details in Appendix B). In the lower part of Table 4, we report defense performance against two 1%-pixel patches. Our defense achieves good defense performance against two patches (e.g., 98.8% clean accuracy and 89.2% certified robust accuracy for two 1% square patches anywhere on the ImageNette images). Moreover, we compare the defense performance against a 2%-pixel square patch (which has the same number of adversarial pixels). We can see that our defense performance for two patches is reduced compared to that for one-patch; however, the numbers are still much higher than prior works against a 2%-pixel square patch in Table 2.

5.2 PatchCleanser and Minority Reports

In this paper, we propose PatchCleanser for certifiably robust prediction without abstention. In contrast, another pixel-masking defense, Minority Reports (MR) [30], only achieves a weaker robustness notion for attack *detection*: an attacker can force MR to always abstain from prediction. Though the certified robust accuracy for these two defenses with different robustness notions are not directly comparable, we implement MR and PatchCleanser (using the same number of masks) and report their defense performance against a 2%-pixel patch on ImageNet in Figure 9.

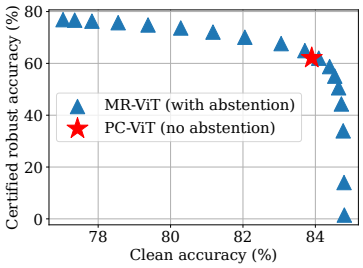

Figure 9: Defense performance of PC-ViT and MR-ViT on ImageNet; note that robustness notions are different (robust prediction for PatchCleanser vs. attack detection for MR).

First, we observe that MR can balance the trade-off between clean accuracy and certified robust accuracy. Second, MR achieves the highest certified robust accuracy (76.9%) due to the easier certification for a weaker robustness notion. Third, PatchCleanser achieves a similar value of certified robust accuracy (∼62%) to MR when their clean accuracy is around 84%. This is remarkable given that PatchCleanser eliminates the issue of abstentions/alerts.

5.3 Limitation and Future Work

Improving defense efficiency. Compared to some prior works [33, 55, 61], PatchCleanser achieves better performance at the cost of efficiency (recall Figure 5). In Section 4.3 (Figure 4), we also see a trade-off between defense performance and efficiency. How to improve the efficiency of the underlying model (e.g., EfficientNet [46]) and the algorithm (e.g., our alternative inference algorithm in Appendix C) is interesting to study. We note that PatchCleanser's runtime can be improved by evaluating masked predictions in parallel with multiple GPUs.

Relaxing the prior estimation of the patch shape and patch size. PatchCleanser requires a conservative estimation of the patch shape/size as the security parameters to generate the mask set. This dependence on the prior knowledge is similar to that of verifiably robust network training [16, 34] and empirical adversarial training [15, 29] against global perturbations [6, 15, 45], which need to know the norm and magnitude of the perturbations. This limitation is also shared by masking-based defenses [30, 55, 58]; an underestimated patch size/shape will undermine the robustness. How to relax the dependence on this prior knowledge is important to study. In Section 5.1, we demonstrate how to mitigate the dependence on prior knowledge of patch shape by considering all possible patch shapes and using a union of different mask shapes.

Handling potential semantic changes caused by masks. PatchCleanser uses masks to achieve substantial robustness against adversarial patches, and we have demonstrated its effectiveness on common image datasets. However, the masking operation might lead to semantic changes for special classi-

fication tasks (e.g., a classifier trained to recognize masks). In these special cases, we could use colored masks for PatchCleanser and further train the classifier to distinguish between vanilla masks and PatchCleanser masks. We leave further explorations for future work.

6 Related Work

6.1 Adversarial Patch Attacks

The adversarial patch attack was first introduced by Brown et al. [4]; this attack focused on generating universal adversarial patches to induce model misclassification. Brown et al. [4] demonstrated that the patch attacker can realize a physical-world attack by printing and attaching the patch to the victim objects. A concurrent paper on the Localized and Visible Adversarial Noise (LaVAN) attack [21] aimed at inducing misclassification in the digital domain. Both of these papers operated in the white-box threat model, with access to the internals of the classifier under attack. PatchAttack [60], on the other hand, proposed a reinforcement learning based attack for generating adversarial patches in the black-box setting.

There have been adversarial patch attacks proposed in other domains such as object detection [28], semantic segmentation [43], and network traffic analysis [44]. In this paper, we focus on test-time attacks against image classification models.

6.2 Adversarial Patch Defenses

To counter the threat of adversarial patches, heuristic-based empirical defenses, Digital Watermark (DW) [18] and Local Gradient Smoothing (LGS) [36], were first proposed. However, Chiang et al. [7] had shown that these defenses were ineffective against an adaptive attacker with the knowledge of the defense algorithm and model parameters [7].

The ineffectiveness of empirical defenses has inspired many certifiably robust defenses. Chiang et al. [7] proposed the first certifiably robust defense against adversarial patches via Interval Bound Propagation (IBP) [16, 34], which conservatively bounded the activation values of neurons to derive a robustness certificate. This defense requires expensive training and does not scale to large models and high-resolution images. Zhang et al. [61] proposed Clipped BagNet (CBN) to clip features of BagNet (a classification model with small receptive fields) for certified robustness. Levine et al. [25] proposed De-randomized Smoothing, which fed small image regions to a classification model and performed majority voting for the final prediction. Xiang et al. [55] proposed PatchGuard as a general defense framework with two key ideas: the use of small receptive fields and secure feature aggregation. Metzen et al. [33] proposed BagCert, a variant of BagNet with majority voting, for certified robustness.

A key takeaway from our paper is that the dependence of prior works on specific model architectures (e.g., small receptive fields [25, 27, 33, 55, 61]) greatly limits the defense performance; in contrast, the compatibility of PatchCleanser with any model architecture leads to state-of-the-art clean accuracy and certified robust accuracy.

Another line of certifiably robust research focuses on attack detection. Minority Reports (MR) [30] places a mask at all image locations and uses the inconsistency in masked prediction voting grids as an attack indicator. PatchGuard++ [58] performed a similar defense in the feature space. We note that the first-round masking of PatchCleanser is is similar to the masking operation of MR; we provided detailed comparison in Section 5.2. A concurrent work ScaleCert [17] uses superficial important neurons to detect a patch attack; we omit its detailed discussion due to different defense objectives.

Some other recent defenses focus on adversarial training and robust model architecture [10, 35, 40, 54], but they lack certifiable robustness guarantee. In other domains like object detection, empirical defenses [20, 26, 41] and certifiably robust defenses [57] have also been proposed. We omit a detailed discussion since PatchCleanser focuses on certifiably robust image classification.

6.3 Other Adversarial Example Attacks

In addition to adversarial patch attacks and defenses, there is a significant body of work on adversarial examples. Conventional adversarial attacks [2, 3, 6, 15, 29, 37, 45] aim to introduce a small global L_p perturbation to the image for model misclassification. Empirical defenses [31, 32, 38, 59] were first proposed to mitigate the threat of adversarial examples, but were later found vulnerable to a strong adaptive attacker with the knowledge of the defense setup [1, 5, 49]. The fragility of these heuristic-based defenses inspired a new research thread on developing certifiably robust defenses [9, 16, 24, 34, 39, 42, 53]. In contrast, we focus on adversarial patch attacks, whose perturbations are localized and thus are realizable in the physical world.

7 Conclusion

In this paper, we propose PatchCleanser for certifiably robust image classification against adversarial patch attacks. Notably, PatchCleanser is compatible with any state-of-the-art classification model (including ones with large receptive fields). PatchCleanser uses a double-masking algorithm to remove all adversarial pixels and recover the correct prediction without any abstention. Our evaluation shows that PatchCleanser outperforms all prior works by a large margin: it is the first certifiably robust defense that achieves clean accuracy comparable to state-of-the-art vanilla models while simultaneously achieving high certified robust accuracy. PatchCleanser thus represents a promising new direction in our quest for secure computer vision systems.

Acknowledgements

We are grateful to David Wagner for shepherding the paper and anonymous reviewers at USENIX Security for their valuable feedback. We are also grateful to Ashwinee Panda, Sihui Dai, Alexander Valtchanov, Xiangyu Qi, and Tong Wu for their insightful comments on the paper draft. This work was supported in part by the National Science Foundation under grants CNS-1553437 and CNS-1704105, the ARL's Army Artificial Intelligence Innovation Institute (A2I2), the Office of Naval Research Young Investigator Award, the Army Research Office Young Investigator Prize, Schmidt DataX award, and Princeton E-ffiliates Award.

References

[1] Anish Athalye, Nicholas Carlini, and David A. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, pages 274–283, 2018.

[2] Marco Barreno, Blaine Nelson, Anthony D Joseph, and J Doug Tygar. The security of machine learning. *Machine Learning*, 81(2):121–148, 2010.

[3] 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 *ECML PKDD*, pages 387–402. Springer, 2013.

[4] Tom B. Brown, Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer. Adversarial patch. In *NeurIPS Workshops*, 2017.

[5] Nicholas Carlini and David A. Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *AISec@CCS*, pages 3–14, 2017.

[6] Nicholas Carlini and David A. Wagner. Towards evaluating the robustness of neural networks. In *IEEE S&P*, pages 39–57, 2017.

[7] Ping-Yeh Chiang, Renkun Ni, Ahmed Abdelkader, Chen Zhu, Christoph Studor, and Tom Goldstein. Certified defenses for adversarial patches. In *ICLR*, 2020.

[8] Edward Chou, Florian Tramer, and Giancarlo Pellegrino. Sentinet: Detecting localized universal attacks against deep learning systems. In *IEEE S&P Workshops*, pages 48–54. IEEE, 2020.

[9] Jeremy M. Cohen, Elan Rosenfeld, and J. Zico Kolter. Certified adversarial robustness via randomized smoothing. In *ICML*, pages 1310–1320, 2019.

[10] Christian Cosgrove, Adam Kortylewski, Chenglin Yang, and Alan Yuille. Robustness out of the box: Compositional representations naturally defend against black-box patch attacks. *arXiv preprint arXiv:2012.00558*, 2020.

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. ImageNet: A large-scale hierarchical image database. In *CVPR*, pages 248–255, 2009.

[12] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[13] 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 *ICLR*, 2021.

[14] fast.ai. ImageNette: A smaller subset of 10 easily classified classes from imagenet. https://github.com/fastai/imagenette, 2020.

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

[16] Sven Gowal, Krishnamurthy Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Relja Arandjelovic, Timothy Arthur Mann, and Pushmeet Kohli. Scalable verified training for provably robust image classification. In *ICCV*, pages 4841–4850, 2019.

[17] 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 *NeurIPS*, 2021.

[18] Jamie Hayes. On visible adversarial perturbations & digital watermarking. In *CVPR Workshops*, pages 1597–1604, 2018.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[20] Nan Ji, YanFei Feng, Haidong Xie, Xueshuang Xiang, and Naijin Liu. Adversarial yolo: Defense human detection patch attacks via detecting adversarial patches. *arXiv preprint arXiv:2103.08860*, 2021.

[21] Danny Karmon, Daniel Zoran, and Yoav Goldberg. LaVAN: Localized and visible adversarial noise. In *ICML*, pages 2512–2520, 2018.

[22] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *ECCV*, 2020.

[23] Alex Krizhevsky. Learning multiple layers of features from tiny images. https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf, 2009.

[24] Mathias Lécuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. Certified robustness to adversarial examples with differential privacy. In *IEEE S&P*, pages 656–672, 2019.

[25] Alexander Levine and Soheil Feizi. (De)randomized smoothing for certifiable defense against patch attacks. In *NeurIPS*, 2020.

[26] Bin Liang, Jiachun Li, and Jianjun Huang. We can always catch you: Detecting adversarial patched objects with or without signature. *arXiv preprint arXiv:2106.05261*, 2021.

[27] Wan-Yi Lin, Fatemeh Sheikholeslami, jinghao shi, Leslie Rice, and J Zico Kolter. Certified robustness against physically-realizable patch attack via randomized cropping, 2021.

[28] Xin Liu, Huanrui Yang, Ziwei Liu, Linghao Song, Yiran Chen, and Hai Li. DPATCH: an adversarial patch attack on object detectors. In *AAAI workshops*, volume 2301, 2019.

[29] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

[30] Michael McCoyd, Won Park, Steven Chen, Neil Shah, Ryan Roggenkemper, Minjune Hwang, Jason Xinyu Liu, and David A. Wagner. Minority reports defense: Defending against adversarial patches. In *ACNS Workshops*, volume 12418, pages 564–582. Springer, 2020.

[31] Dongyu Meng and Hao Chen. Magnet: A two-pronged defense against adversarial examples. In *CCS*, pages 135–147, 2017.

[32] Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. In *ICLR*, 2017.

[33] Jan Hendrik Metzen and Maksym Yatsura. Efficient certified defenses against patch attacks on image classifiers. In *ICLR*, 2021.

[34] Matthew Mirman, Timon Gehr, and Martin T. Vechev. Differentiable abstract interpretation for provably robust neural networks. In *ICML*, pages 3575–3583, 2018.

[35] Norman Mu and David Wagner. Defending against adversarial patches with robust self-attention. In *ICML Workshops*, 2021.

[36] Muzammal Naseer, Salman Khan, and Fatih Porikli. Local gradients smoothing: Defense against localized adversarial attacks. In *WACV*, pages 1300–1307, 2019.

[37] Nicolas Papernot, Patrick D. McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *EuroS&P*, pages 372–387, 2016.

[38] Nicolas Papernot, Patrick D. McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *IEEE S&P*, pages 582–597, 2016.

[39] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. In *ICLR*, 2018.

[40] Sukrut Rao, David Stutz, and Bernt Schiele. Adversarial training against location-optimized adversarial patches. In *ECCV Workshops*, 2020.

[41] Aniruddha Saha, Akshayvarun Subramanya, Koninika Patil, and Hamed Pirsiavash. Role of spatial context in adversarial robustness for object detection. In *CVPR Workshops*, pages 784–785, 2020.

[42] Hadi Salman, Jerry Li, Ilya P. Razenshteyn, Pengchuan Zhang, Huan Zhang, Sébastien Bubeck, and Greg Yang. Provably robust deep learning via adversarially trained smoothed classifiers. In *NeurIPS*, pages 11289–11300, 2019.

[43] Vikash Sehwag, Chawin Sitawarin, Arjun Nitin Bhagoji, Arsalan Mosenia, Mung Chiang, and Prateek Mittal. Not all pixels are born equal: An analysis of evasion attacks under locality constraints. In *CCS posters*, pages 2285–2287, 2018.

[44] Shawn Shan, Arjun Nitin Bhagoji, Haitao Zheng, and Ben Y Zhao. A real-time defense against website fingerprinting attacks. In *AISec@CCS*, 2021.

[45] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *ICLR*, 2014.

[46] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *ICML*, volume 97, pages 6105–6114. PMLR, 2019.

[47] Ilya Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, et al. Mlp-mixer: An all-mlp architecture for vision. In *NeurIPS*, 2021.

[48] Hugo Touvron, Piotr Bojanowski, Mathilde Caron, Matthieu Cord, Alaaeldin El-Nouby, Edouard Grave, Armand Joulin, Gabriel Synnaeve, Jakob Verbeek, and Hervé Jégou. Resmlp: Feedforward networks for image classification with data-efficient training. *arXiv preprint arXiv:2105.03404*, 2021.

[49] Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. In *2020 USENIX Security and AI Networking Summit (ScAINet)*, 2020.

[50] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, pages 5998–6008, 2017.

[51] Ross Wightman. Pytorch image models. https://github.com/rwightman/pytorch-image-models, 2019.

[52] Ross Wightman, Hugo Touvron, and Hervé Jégou. Resnet strikes back: An improved training procedure in timm. *arXiv preprint arXiv:2110.00476*, 2021.

[53] Eric Wong and J. Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *ICML*, pages 5283–5292, 2018.

[54] Tong Wu, Liang Tong, and Yevgeniy Vorobeychik. Defending against physically realizable attacks on image classification. In *ICLR*, 2020.

[55] 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 *USENIX Security*, 2021.

[56] Chong Xiang, Saeed Mahloujifar, and Prateek Mittal. Patchcleanser: Certifiably robust defense against adversarial patches for any image classifier. *arXiv preprint arXiv:2108.09135*, 2021.

[57] Chong Xiang and Prateek Mittal. DetectorGuard: Provably securing object detectors against localized patch hiding attacks. In *CCS*, 2021.

[58] Chong Xiang and Prateek Mittal. Patchguard++: Efficient provable attack detection against adversarial patches. In *ICLR Workshops*, 2021.

Table 5: Effect of different models and masked model training

Dataset	ImageNette		ImageNet		CIFAR-10	
Accuracy (%)	clean	robust	clean	robust	clean	robust
PC-ResNet-vanilla	99.4	88.1	81.1	41.6	94.3	39.4
PC-ResNet-cutout	99.6	94.4	81.6	53.4	97.8	78.8
PC-ViT-vanilla	99.3	94.2	83.6	59.4	97.9	77.5
PC-ViT-cutout	**99.6**	**96.4**	**83.9**	**62.1**	**98.7**	**89.1**
PC-MLP-vanilla	98.6	91.3	79.6	53.1	95.6	62.0
PC-MLP-cutout	99.3	95.3	79.4	53.8	97.0	78.0

[59] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. In *NDSS*, 2018.

[60] Chenglin Yang, Adam Kortylewski, Cihang Xie, Yinzhi Cao, and Alan Yuille. Patchattack: A black-box texture-based attack with reinforcement learning. In *ECCV*, pages 681–698, 2020.

[61] Zhanyuan Zhang, Benson Yuan, Michael McCoyd, and David Wagner. Clipped bagnet: Defending against sticker attacks with clipped bag-of-features. In *Deep Learning and Security Workshop (DLS)*, 2020.

A Details of Experiment Setup

For all high-resolution images (i.e., ImageNet [11], ImageNette [14]), we resize and crop them into 224×224. For low-resolution images (i.e., CIFAR-10 [23]), we resize them to 224×224 (via bicubic interpolation) without cropping. We use `timm` library [51] to build all vanilla models and load weights trained for ImageNet [11]. In our default setting, we use Cutout data augmentation [12] for the model training. Specifically, we apply 2 masks of size 128×128 at random locations to the 224 × 224 training images; this training-time data augmentation can improve model prediction invariance to pixel masking. We note that the Cutout training is only an optional step in PatchCleanser pipeline.We further report the defense performance with and without Cutout training in Table 5.

We provide additional details of experiment setup in our technical report [56]. We release our source code at https://github.com/inspire-group/PatchCleanser.

B Additional Details for Defenses against Different Patch Shapes and Multiple Patches

In Section 5.1, we quantitatively discussed PatchCleanser against a stronger attacker who use a set of different patch shapes or use multiple patches. In this section, we provide additional details of our implementation and evaluation.

Different patch shapes. To generate a mask set that is robust to a patch that use *any* rectangle shape that covers at most 1% pixels of the 224×224 image (502 pixels), we first consider a rectangle shape set $\mathcal{S} = \{5 \times 224, 12 \times 83, 23 \times 38, 39 \times 20, 84 \times 12, 224 \times 5\}$. We claim that these 6 shapes in $\mathcal{S}$ together can *cover any 1%-pixel rectangle shape*.[6] To prove this covering property, we let a and b be the height and width of the rectangle patch, respectively. We know that $a \cdot b < 502$ (1% image pixels). If $a \leq 5$, then the patch is covered by the 5×224 rectangle. If $5 < a \leq 12$, then $b < 502/6 < 84$, and the patch is covered by the 12×83 rectangle. If $12 < a \leq 23$, then $b < 502/13 < 39$, and the patch is covered by the 23×38 rectangle. If $23 < a \leq 39$, then $b < 502/24 < 21$, and the patch is covered by the 39×20 rectangle. If $39 < a \leq 84$, then $b < 502/40 < 13$, and the patch is covered by the 84×12 rectangle. Finally, if $84 < a \leq 224$, then $b < 502/85 < 6$, and the patch is covered by 224×5 rectangle. Now we have considered all 1%-pixel rectangles and proved the covering property. Next, we can generate a mask set $\mathcal{M}'$ for every shape in $\mathcal{S}$ (as discussed in Section 3.4) and take the union of all $\mathcal{M}'$ as the $\mathcal{R}$-covering mask set $\mathcal{M}$.

We implement our strategy and report the defense performance for 500 randomly selected test images in Table 4. We additionally note that, in some cases, we can see a higher clean accuracy for PatchCleanser against all rectangle shapes, compared to PatchCleanser against the square patch. This is because we are using a larger number of masks, and PatchCleanser could become less likely to output an incorrect disagreer label in the clean setting.

Multiple patches. As discussed in Section 5.1, to defend against an attacker who can use K patches, We can generate a mask set with all possible K-mask combinations, at least one of which can remove all patches. We then apply our double-mask algorithm with this mask set for robust image classification. In order to certify the robustness of a given image, we need to check if the image predictions are correct for all $2K$-mask combinations.

In Table 4, we provide a proof of concept for our multiple-patch defense. We select 500 random test images from each dataset and report defense performance against two 1%-pixel patches.

C Challenger Masking: Improving Inference Complexity

Our double-masking defense (Algorithm 1 in Section 3.2) has inference complexity of $O(|\mathcal{M}|^2)$ in the worst case (doing all two-mask predictions). In this subsection, we introduce a new inference algorithm named challenger masking, which has better worst-case complexity of $O(|\mathcal{M}|)$, the same certified robust accuracy, but slightly lower clean accuracy. Similar to

[6] There are other valid shape sets $\mathcal{S}$. Here, we only provide one example as proof of concept.

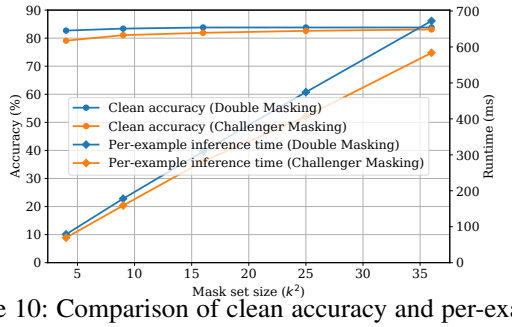

Figure 10: Comparison of clean accuracy and per-example inference time between two inference algorithms

our double-masking algorithm (Algorithm 1 in Section 3.2), the challenger masking involves two rounds of masking: if the first-round masking reaches a unanimous agreement on masked predictions, we return the agreed prediction label; otherwise, we play a challenger game (in the second-round masking) to settle the disagreement.

Challenger game. The high-level idea of the challenger game is to let different masked predictions challenge each other and output the game-winner as the final prediction. For two masks $\mathbf{m}_0, \mathbf{m}_1$ that give different masked predictions (i.e., $\bar{y}_0 \neq \bar{y}_1, \bar{y}_0 = \mathbb{F}(\mathbf{x} \odot \mathbf{m}_0), \bar{y}_1 = \mathbb{F}(\mathbf{x} \odot \mathbf{m}_1)$), we apply both two masks to the image and evaluate the two-mask prediction as $\hat{y} = \mathbb{F}(\mathbf{x} \odot \mathbf{m}_0 \odot \mathbf{m}_1)$. If the two-mask prediction $\hat{y}$ agrees with any of the one-mask prediction $\bar{y}_0$ or $\bar{y}_1$, we consider the agreed prediction as the winner of this challenger game. Our algorithm will discard a mask once it loses any challenger game and continue to play the game until there is only one label left (i.e., no challenger exists). Finally, we output the winner label as the robust prediction. Intuitively, if the first-round mask removes the patch, then adding a second mask is unlikely to give a different prediction (since the second mask is applied to a benign image). Therefore, the mask that removes the patch has a great chance to win this challenger game. We provide additional details of algorithm pseudocode in our technical report [56].

Robustness certification. The robustness certification condition for this challenger game is the same as our double-masking defense: two-mask correctness. This is because if a model has two-mask correctness, the first-round mask that removes the patch will never lose the challenger game.

Remark: defense complexity. The first-round masking needs $O(|\mathcal{M}|)$ masking operations evaluation. In the challenger game, every first-round mask will be used as a challenger for at most one time. Therefore, the complexity for the challenger game is also $O(|\mathcal{M}|)$. In summary, the algorithm has a complexity of $O(|\mathcal{M}|)$, in contrast to $O(|\mathcal{M}|^2)$ of our double-masking algorithm (Algorithm 1).

Performance evaluation. We note that challenger masking and double-masking (Algorithm 1) have the same certified robust accuracy (certified via two-mask correctness), but different clean accuracy and inference efficiency. In Fig-

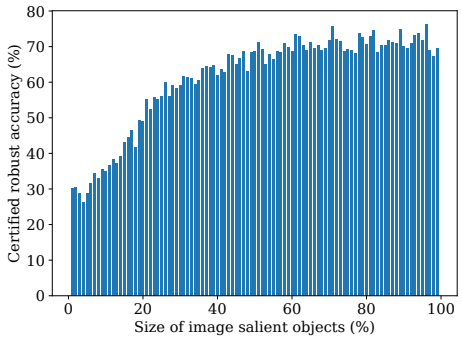

Figure 11: Certified robust accuracy for images with different salient object sizes (PC-ViT on ImageNet)

ure 10, we plot the per-example runtime (on clean images) and clean accuracy of two algorithms on the ImageNet dataset. As shown in the figure, the challenger masking algorithm has better defense efficiency but lower clean accuracy. We note that the double-masking algorithm has higher clean accuracy because it is more conservative in trusting a one-mask disagreer: double-masking requires all two-mask predictions in the second-round masking to give the same prediction label while challenger masking does not require this. As a result, the double-masking algorithm is less likely to return an incorrect disagreer prediction for clean images whose robustness cannot be certified. We prioritize the defense accuracy and choose the double-masking algorithm in the main body of the paper.

D PatchCleanser Robustness for Images with Different Object Sizes and Object Classes

In this subsection, we study how the certified robustness of PC-ViT is affected by object sizes and object classes on the ImageNet [11] dataset.

Object size. We take the annotations of object bounding boxes from the ImageNet [11] dataset to study how the object size affects the certified robustness. For each image, we count the number of pixels of the union of all bounding boxes as our measure of the salient object size. We plot the certified robust accuracy (against a 2%-pixel patch) of images with different salient object sizes (in the percentage of image pixels) in Figure 11. As shown in the figure, we can see that PatchCleanser generally has higher certified robust accuracy for larger objects. This is an expected result since small objects might be completely occluded by the adversarial patch (see Figure 12 for visual examples).

Object class. In Figure 13, we plot the distribution of certified robust accuracy for different image object classes. We can see that most classes have high certified robustness, but the certified robust accuracy can vary greatly across different object classes. For example, we can achieve 100% certified robust accuracy for some classes (e.g., classes "n02116738: African hunting dog", "n02342885: hamster", "n11879895: rapeseed", and "n12057211: yellow lady's slipper") while we only have 4% certified robust accuracy for some classes (e.g., classes "n02107908: Appenzeller" and "n04152593: screen"). Moreover, we plot the average object size (in the percentage of the image pixels) for each class. The result further demonstrates that the certified robust accuracy is affected by not only the object size (Figure 11), but also the object class. Further diagnosis on the classes with poor performance is one of our future work directions.

Figure 12: Visualization of 32×32 occlusion on 224×224 ImageNet images with small objects

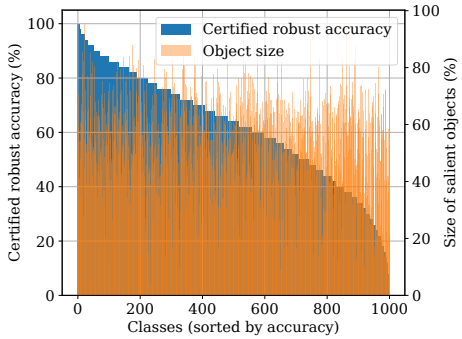

Figure 13: Certified robust accuracy and average object size across different classes (PC-ViT on ImageNet; the class indices are sorted based on the certified robust accuracy)

E Proof of Lemma 4

Lemma 4. The mask set $\mathcal{M}_{m,s,n}$ is $\mathcal{R}$-covering for a patch that is no larger than $p^* = m - s + 1$.

Proof. Without loss of generality, we consider the first two adjacent masks in the 1-D scenario, whose mask pixel index ranges are $[0, m-1]$ and $[s, s+m-1]$, respectively. Now let us consider an adversarial patch of size p^*. In order to avoid being completely masked by the first mask, the smallest index of the patch has to be no smaller than $j^* = (m-1) - (p^* - 1) + 1 = s$. However, we find that the second mask starts from the index s, so the patch that evades the first mask will be captured by the second mask. □