Towards Verifying Robustness of Neural Networks Against A Family of Semantic Perturbations

Jeet Mohapatra[1], Tsui-Wei Weng[1], Pin-Yu Chen[2], Sijia Liu[2] and Luca Daniel[1]

[1]MIT EECS, [2]MIT-IBM Watson AI Lab, IBM Research

Abstract

Verifying robustness of neural networks given a specified threat model is a fundamental yet challenging task. While current verification methods mainly focus on the ℓ_p-norm threat model of the input instances, robustness verification against semantic adversarial attacks inducing large ℓ_p-norm perturbations, such as color shifting and lighting adjustment, are beyond their capacity. To bridge this gap, we propose Semantify-NN, a model-agnostic and generic robustness verification approach against semantic perturbations for neural networks. By simply inserting our proposed semantic perturbation layers (SP-layers) to the input layer of any given model, Semantify-NN is model-agnostic, and any ℓ_p-norm based verification tools can be used to verify the model robustness against semantic perturbations. We illustrate the principles of designing the SP-layers and provide examples including semantic perturbations to image classification in the space of hue, saturation, lightness, brightness, contrast and rotation, respectively. In addition, an efficient refinement technique is proposed to further significantly improve the semantic certificate. Experiments on various network architectures and different datasets demonstrate the superior verification performance of Semantify-NN over ℓ_p-norm-based verification frameworks that naively convert semantic perturbation to ℓ_p-norm. The results show that Semantify-NN can support robustness verification against a wide range of semantic perturbations.

1. Introduction

As deep neural networks (DNNs) become prevalent in machine learning and achieve the best performance in many standard benchmarks, their unexpected vulnerability to adversarial examples has spawned a wide spectrum of research disciplines in adversarial robustness, spanning from effective and efficient methods to find adversarial examples for causing model misbehavior (i.e., attacks), to detect adversar-

ial inputs and become attack-resistant (i.e., defenses), and to formally evaluate and quantify the level of vulnerability of well-trained models (i.e., robustness verification [10], certification [20], or evaluation [21]).

Given a data sample x and a trained DNN, the primary goal of verification tools is to provide a "robustness certificate" for verifying its properties in a specified threat model. For image classification tasks, the commonly used threat model is an ℓ_p-norm bounded perturbation to x, where p usually takes the value $p \in \{1, 2, \infty\}$, approximating the similarity measure of visual perception between x and its perturbed version. The robustness property to be verified is the consistent decision making of the DNN of any sample drawn from an ℓ_p-norm ball centered at x with radius ϵ. In other words, verification methods aim to verify whether the DNN can give the same top-1 class prediction to all samples in the ϵ-ball centered at x. Note that verification is attack-agnostic as it does not incur any attack methods for verification. Moreover, if the ϵ-ball robustness certificate for x is verified, it assures no adversarial attacks using the same threat model can alter the top-1 prediction of the DNN for x. Although finding the maximal verifiable ϵ value (i.e. the minimum distortion) is computationally intractable for DNNs [10], recent verification methods have developed efficient means for computing an lower bound on minimum distortion as a verifiable ϵ-ball certificate [11, 22, 20, 4, 23, 16, 19, 15, 3].

Beyond the ℓ_p-norm bounded threat model, recent works have shown the possibility of generating *semantic* adversarial examples based on semantic perturbation techniques such as color shifting, lighting adjustment and rotation [8, 13, 2, 9, 6, 5]. We refer the readers to Figure 1 for the illustration of some semantic perturbations for images. Notably, although semantically similar, these semantic adversarial attacks essentially consider different threat models than ℓ_p-norm bounded attacks in the RGB space. Therefore, semantic adversarial examples usually incur large ℓ_p-norm perturbations to the original data sample and thus exceed the verification capacity of ℓ_p-norm based verification methods. To bridge this gap and with an endeavor to render robustness ver-

ification methods more inclusive, we propose *Semantify-NN*, a model-agnostic and generic robustness verification against semantic perturbations. *Semantify-NN* is model-agnostic because it can apply to any given trained model by simply inserting our designed *semantic perturbation layers* (SP-layers). It is also generic since after adding SP-layers, one can apply any ℓ_p-norm based verification tools for certifying semantic perturbations. In other words, our proposed SP-layers work as a carefully designed converter that transforms semantic threat models to ℓ_p-norm threat models. As will be evident in the experiments, *Semantify-NN* yields substantial improvement over ℓ_p-norm based verification methods that directly convert semantic perturbations to the equivalent ℓ_p norm perturbations in the RGB space.

Our main contributions are summarized as below:

- We propose *Semantify-NN*, a model-agnostic and generic robustness verification toolkit for semantic perturbations. *Semantify-NN* can be viewed as a powerful extension module consisting of novel *semantic perturbation layers* (SP-layers) and is compatible to existing ℓ_p-norm based verification tools. The results show that *Semantify-NN* can support robustness verification against a wide range of semantic perturbations.

- We elucidate the design principles of our proposed SP-layers for a variety of semantic attacks, including hue/saturation/lightness change in color space, brightness and contrast adjustment, rotation, translation and occlusion. We also propose to use input space refinement and splitting methods to further improve the performance of robustness verification. In addition, we illustrate the need and importance of robustness verification for continuously parameterized perturbations.

- We propose an efficient refinement technique, *input splitting*, that can further tighten the semantic certificate delivered by *Semantify-NN*. Our extensive experiments evaluated on the rich combinations of three datasets (MNIST, CIFAR-10 and GTSRB) and five different network architectures (MLPs and CNNs) corroborate the superior verification performance of *Semantify-NN* over naive ℓ_p-norm based verification methods. In particular, our method without further refinement can already achieve around 2-3 orders of magnitude larger (tighter) semantic robustness certificate than the baselines that directly uses the same ℓ_p-norm verification methods to handle semantic perturbations. With our *input splitting* technique, the semantic robustness certificate can be further improved by 100-300% .

2. Background and Related Works.

ℓ_p-**norm based verification**   For ℓ_p-norm bounded threat models, current robustness verification methods are mainly
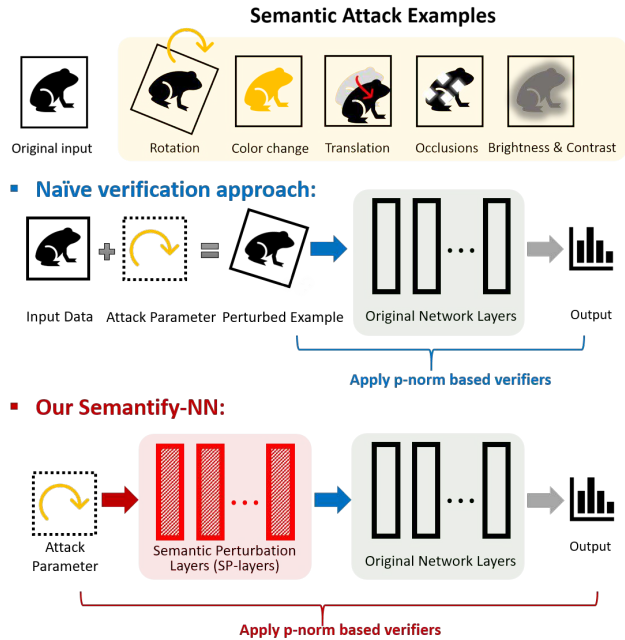

Figure 1: Schematic illustration of our proposed *Semantify-NN* robustness verification framework. Given a semantic attack threat model, *Semantify-NN* designs the corresponding semantic perturbation layers (SP-layers) and inserts them to the input layer of the original network for verification. With SP-layers, *Semantify-NN* can use any ℓ_p-norm based verification method for verifying semantic perturbations.

based on solving a convex relaxation problem [11, 22, 4], devising tractable linear bounds on activation functions and layer propagation [20, 23, 19, 17, 3, 18]. We refer readers to the prior arts and the references therein for more details. The work in [19] considers brightness and contrast in the *linear transformation* setting, which still falls under the ℓ_p norm threat model. The work in [18] has scratched the surface of semantic robustness verification by considering rotations attacks with the ℓ_p-norm based methods. However, we show that with our carefully designed refinement techniques, the robustness certificate can be significantly improved around 50-100% in average. Moreover, we consider a more general and challenging setting than [19] where the color space transformation can be *non-linear* and hence directly applying ℓ_p-norm based method could result in a very loose semantic robustness certificate. A recent work [1] considers geometric image transformations (i.e. spatial transformations such as rotation, translation, scaling, shearing), brightness and contrast, and interpolation, and their technique is based on sampling and requires solving a linear programming problem. In contrast, our framework is more general and efficient, because the main idea of our work is to describe non-linear semantic transformations with SP-Layers and apply a novel

refinement technique. The idea of SP-Layers enables the use of any state-of-the-art ℓ_p-norm based robustness verification tools [11, 22, 4, 20, 23, 19, 17, 3, 18] and our proposed refinement technique has very light computational overhead on the linear-bounding based verifiers [20, 23, 19, 17, 3, 18] while obtaining significant performance improvement. The work in [7] proposes to use semantic maps to evaluate semantic robustness, but how to apply this analysis to develop semantic robustness verification is beyond their scope.

Semantic adversarial attack. In general, semantic adversarial attacks craft adversarial examples by tuning a set of parameters governing semantic manipulations of data samples, which are either explicitly specified (e.g. rotation angle) or implicitly learned (e.g. latent representations of generative models). In [8], the HSV (hue, saturation and value) representation of the RGB (red, green and blue) color space is used to find semantic adversarial examples for natural images. To encourage visual similarity, the authors propose to fix the value, minimize the changes in saturation, and fully utilize the hue changes to find semantic adversarial examples. In [13], the authors present a physically-based differentiable renderer allowing propagating pixel-level gradients to the parametric space of lightness and geometry. In [2], the authors introduce texture and colorization to induce semantic perturbation with a large ℓ_p norm perturbation to the raw pixel space while remaining visual imperceptibility. In [9], an adversarial network composed of an encoder and a generator conditioned on attributes are trained to find semantic adversarial examples. In [6, 5], the authors show that simple operations such as image rotation or object translation can result in a notable mis-classification rate.

Continuously parameterized semantic perturbations cannot be enumerated. We note that for the semantic perturbations that are *continuously parameterized* (such as Hue, Saturation, Lightness, Brightness, Contrast and Rotations that are considered in Section 3.2), it is *not* possible to enumerate all possible values even if we only perturb one single parameter (e.g. rotation angle). The reason is that these parameters take *real values* in the continuous space, hence it is not possible to finitely enumerate all possible values, unlike its discrete parameterized counterpart (e.g. translations and occlusions have finite enumerations). Take the rotation angle for example, an attacker can try to do a grid search by sweeping rotation angle θ from $0°$ to $90°$ with a uniform 10^6 grids. However, if the attacks are not successful at $\theta = 30°$ and $30.00009°$, it does not eliminate the possibility that there could exist some θ' that could "fool" the classifier where $30° < \theta' < 30.00009°$. This is indeed the motivation and necessity to have the robustness verification algorithm for semantic perturbations as proposed in this paper – with a proper semantic robustness verification algorithm, we can

guarantee that neural networks will have a consistent prediction on the given image for any $\theta < a$, where a is the semantic robustness certificate (of the image) delivered by our algorithm.

3. Semantify-NN: A Semantic Robustness Verification Framework

In this section, we formally introduce *Semantify-NN*, a general robustness verification framework for semantic perturbations. We begin by providing a general problem formulation in Sec. 3.1 and discuss the semantic perturbations that fall in this family in Sec. 3.2. Next, we elucidate the two key components of Semantify-NN: *SP-Layers* in Sec. 3.2 and a novel refinement technique *Implicit Input Splitting* in Sec. 3.3. These two components are the core of Semantify-NN to enable the use of any ℓ_p-norm based verifiers and further boost verification performance with light computation overhead compared to existing refinement techniques.

3.1. Problem Formulation

Let $\mathcal{A}$ be a general threat model. For an input data sample x, we define an associated space of perturbed images x', denoted as the *Attack Space* $\Omega_{\mathcal{A}}(x)$, which is equipped with a distance function $d_{\mathcal{A}}$ to measure the magnitude of the perturbation. The robustness certification problem under the threat model $\mathcal{A}$ is formulated as follow: given a trained K-class neural network function f, input data sample x, we aim to find the largest δ such that

$$\min_{x' \in \Omega_{\mathcal{A}}(x), d_{\mathcal{A}}(x',x) \leq \delta} \left(\min_{j \neq c} f_c(x') - f_j(x') \right) > 0, \quad (1)$$

where $f_j(\cdot)$ denotes the confidence (or logit) of the j-th class, $j \in \{1, 2, \ldots, K\}$, and c is the predicted class of un-perturbed input data x.

3.2. Semantic Perturbation

We consider semantic threat models that target semantically meaningful attacks, which are usually beyond the coverage of conventional ℓ_p-norm bounded threat models in the pixel space. For an attack space $\Omega_{\mathcal{A}}^k$, there exists a function $g : X \times (I_1 \times I_2 \times \ldots \times I_k) \to X$ such that

$$\Omega_{\mathcal{A}}^k(x) = \{g(x, \epsilon_1, \ldots, \epsilon_k) \mid \epsilon_i \in I_i\}, \quad (2)$$
$$d_{\mathcal{A}}(g(x, \epsilon_1, \ldots, \epsilon_k), x) = ||(\epsilon_1, \ldots, \epsilon_k)||_p,$$

where X is the pixel space (the raw image), I_i denotes a set of feasible semantic operations, and $\|\cdot\|_p$ denotes ℓ_p norm. The parameters ϵ_i specify semantic operations selected from I_i. For example, ϵ_i can describe some human-interpretable characteristic of an image, such as translations shift, rotation angle, etc. For convenience we define $\boldsymbol{\epsilon}^k = (\epsilon_1, \epsilon_2, \ldots, \epsilon_k)$ and $I^k = I_1 \times \ldots \times I_k$ where k denotes the dimension

Table 1: Semantic perturbations considered in Section 3. All the listed attacks can be described in Equation (2) with $p = \infty$.

Semantic perturbation	Discretely parameterized		Continuously parameterized				
	Translation	Occlusion	Hue	Saturation	Lightness	Brightness & Contrast	Rotation
# of attack parameter k	2	3	1	1	1	2	1
SP-Layer	Direct enumeration		Yes	Yes	Yes	Yes	Yes

of the semantic attack. In other words, we show that it is possible to define an explicit function g for all the semantic perturbations considered in this work, including translations, occlusions, color space transformations, and rotations, and we then measure the ℓ_p norm of the semantic perturbations on the space of semantic features ϵ^k rather than the raw pixel space. Notice that the conventional ℓ_p norm perturbations on the raw RGB pixels is a special case under this definition: by letting I_i equal to a bounded real set (i.e. $x'_i - x_i$, all possible difference between i-th pixel) and k be the dimension of input vector x, we recover $d_{\mathcal{A}} = \|x' - x\|_p$.

Based on the definition above, semantic attacks can be divided into two categories: discretely parameterized perturbations (i.e. I_k is a discrete set) including translation and occlusions and continuously parameterized perturbations (i.e. I_k is a continuous set) including color changes, brightness, contrast, and spatial transformations (e.g. rotations).

Discretely Parameterised Semantic Perturbation

Translation Translation is a 2-dimensional semantic attack with the parameters being the relative position of left-uppermost pixel of perturbed image to the original image. Therefore, $I_1 = \{0, 1, \ldots, r\}, I_2 = \{0, 1, \ldots, t\}$ where r, t are the dimensions of width and height of our input image x. Note that any padding methods can be applied including padding with the black pixels or boundary pixels, etc.

Occlusion Similar to translation, occlusion attack can be expressed by 3-dimensional attack parameters: the coordinates of the left-uppermost pixel of the occlusion patch and the occlusion patch size[1]. Note that for discretely paramert-erised semantic perturbations, provided with sufficient computation resources, one could simply exhaustively enumerate all the possible perturbed images. At the scale of our considered image dimensions, we find that exhaustive enumeration can be accomplished within a reasonable computation time and the generated images can be used for direct verification. In this case, the SP-layers are reduced to enumeration operations given a discretely parameterized semantic attack threat model. Nonetheless, the computation complexity of exhaustive enumeration grows combinatorially when considering

a joint attack threat model consisting of multiple types of discretely parameterized semantic attacks.

Continuously Parameterized Semantic Perturbation

Most of the semantic perturbations fall under the framework where the parameters are continuous values, i.e., $I^k \subset \mathbb{R}^k$. We propose the idea of adding *semantic perturbation layers* (SP-layers) to the input layer of any given neural network model for efficient robustness verification, as illustrated in Figure 1. By letting $g_x(\epsilon^k) = g(x, \epsilon^k)$, the verification problem for neural network f formulated in (1) becomes

$$\min_{\epsilon^k \in I^k, d_{\mathcal{A}}(g_x(\epsilon^k), x) \leq \delta} \left(\min_{j \neq c} f_c(g_x(\epsilon^k)) - f_j(g_x(\epsilon^k)) \right) > 0. \tag{3}$$

If we consider the new network as $f^{sem} = f \circ g_x$, then we have the following problem:

$$\min_{\epsilon^k \in I^k, \|\epsilon^k\|_p \leq \delta} \left(\min_{j \neq c} f_c^{sem}(\epsilon^k) - f_j^{sem}(\epsilon^k) \right) > 0, \tag{4}$$

which has a similar form to ℓ_p-norm perturbations but now on the semantic space I^k. The proposed SP-layers allow us to explicitly define the dimensionality of our perturbations and put explicit dependence between the manner and the effect of the semantic perturbation on different pixels of the image. In other words, one can view our proposed SP-layers as a parameterized input transformation function from the semantic space to RGB space and $g(x, \epsilon^k)$ is the perturbed input in the RGB space which is a function of perturbations in the semantic space. Our key idea is to express g in terms of commonly-used activation functions and thus g is in the form of neural network and can be easily incorporated into the original neural network classifier f. Note that g can be arbitrarily complicated to allow for general transformations for SP-layers; nevertheless, it does not result in any difficulties to apply the conventional ℓ_p-norm based methods such as [23, 19, 16, 3, 20], as we only require the activation functions to have custom linear bounds and do not need them to be continuous or differentiable. Below we specify the explicit form of SP-layers corresponding to five different semantic perturbations using (i) hue, (ii) saturation, (iii) lightness, (iv) brightness and contrast, and (v) rotation.

---

[1] we use squared patch, but it can be rectangular in general.

Color space transformation   We consider color transformations parameterized by the hue, saturation and lightness (HSL space). Unlike RGB values, HSL form a more intuitive basis for understanding the effect of the color transformation as they are semantically meaningful. For each of the basis, we can define the following functions for g:

- *Hue*   This dimension corresponds to the position of a color on the color wheel. Two colors with the same hue are generally considered as different shades of a color, like blue and light blue. The hue is represented on a scale of 0-360° which we have rescaled to the range $[0, 6]$ for convenience. Therefore, we have $g(R, G, B, \epsilon_h) = (d \cdot \phi_R^h(h') + m, d \cdot \phi_G^h(h') + m, d \cdot \phi_B^h(h') + m)$, where $d = (1 - |2l - 1|)s, m = l - \frac{d}{2}$ and $h' = (h + \epsilon_h)$ mod 6 are functions of R, G, B independent of ϵ_h and

$$(\phi_R^h(h'), \phi_G^h(h'), \phi_B^h(h')) = \begin{cases} (1, V, 0) & 0 \leq h' \leq 1 \\ (V, 1, 0) & 1 \leq h' \leq 2 \\ (0, 1, V) & 2 \leq h' \leq 3 \\ (0, V, 1) & 3 \leq h' \leq 4 \\ (V, 0, 1) & 4 \leq h' \leq 5 \\ (1, 0, V) & 5 \leq h' \leq 6 \end{cases}$$

where $V = (1 - |(h' \mod 2) - 1|)$.

  For $0 \leq h' \leq 6$ the above can be reduced to the following in the ReLU form ($\sigma_i(x) = \text{ReLU}(x - i)$) and hence can be seen as one hidden layer with ReLU activation connecting from hue space to original RGB space:

$$\phi_R^h(h') = 1 + \sigma_2(h') + \sigma_4(h') - (\sigma_5(h') + \sigma_1(h'))$$
$$\phi_G^h(h') = \sigma_0(h') + \sigma_4(h') - (\sigma_1(h') + \sigma_3(h')) \qquad (5)$$
$$\phi_B^h(h') = \sigma_2(h') + \sigma_6(h') - (\sigma_5(h') + \sigma_3(h'))$$

- *Saturation*   This corresponds to the colorfulness of the picture. At saturation 0, we get grey-scale images; while at a saturation of 1, we see the colors pretty distinctly. We have $g(R, G, B, \epsilon_s) = (d_R \cdot \phi^s(s') + l, d_G \cdot \phi^s(s') + l, d_B \cdot \phi^s(s') + l)$ where $s' = s + \epsilon_s, d_R = \frac{R-l}{s}, d_G = \frac{G-l}{s}$ and $d_B = \frac{B-l}{s}$ are functions of R, G, B independent of ϵ_s and

$$\phi^s(s') = \min(\max(s', 0), 1) = \sigma_0(s') - \sigma_1(s') \qquad (6)$$

- *Lightness*   This property corresponds to the perceived brightness of the image where a lightness of 1 gives us white and a lightness of 0 gives us black images. In this case, $g(R, G, B, \epsilon_l) = (d_R \cdot \phi_1^l(l') + \phi_2^l(l'), d_G \cdot \phi_1^l(l') + \phi_2^l(l'), d_B \cdot \phi_1^l(l') + \phi_2^l(l'))$ where $l' = l + \epsilon_l$, $d_R = \frac{R-l}{1-|2l-1|}, d_G = \frac{G-l}{1-|2l-1|}$, and $d_B = \frac{B-l}{1-|2l-1|}$ are functions of R, G, B independent of ϵ_l and

$$\phi_1^l(l') = 1 - |2 \cdot \min(\max(l', 0), 1) - 1|$$
$$= -\sigma_0(2 \cdot l') - \sigma_2(2 \cdot l') + 2 \cdot \sigma_1(2 \cdot l') + 1 \quad (7)$$
$$\phi_2^l(l') = \min(\max(l', 0), 1) = \sigma_0(l') - \sigma_1(l')$$

Brightness and contrast   We also use the similar technique as HSL color space for multi-parameter transformations such as brightness and contrast: the attack parameters are ϵ_b for brightness perturbation and ϵ_c for contrast perturbation, and we have

$$g(x, \epsilon_b, \epsilon_c) = \min(\max((1 + \epsilon_c) \cdot x + \epsilon_b, 0), 1) \qquad (8)$$
$$= \sigma_0((1 + \epsilon_c) \cdot x + \epsilon_b) - \sigma_1((1 + \epsilon_c) \cdot x + \epsilon_b)$$

Therefore, g can be expressed as one additional ReLU layer before the original network model, which is the proposed SP Layers in Figure 1.

Rotation   We have 1-dimensional semantic attack parameterized by the rotation angle θ, and we consider rotations at the center of the image with the boundaries being extended to the area outside the image. We use the following interpolation to get the values $x'_{i,j}$ of output pixel at position (i, j) after rotation by θ. Let $i' = i\cos\theta - j\sin\theta, j' = j\cos\theta + i\sin\theta$, then

$$x'_{i,j} = \frac{\sum_{k,l} x_{k,l} \cdot \max(0, 1 - \sqrt{(k - i')^2 + (l - j')^2})}{\sum_{k,l} \max(0, 1 - \sqrt{(k - i')^2 + (l - j')^2})} \qquad (9)$$

where k, l range over all possible values. For individual pixels at position (k, l) of the original image, the scaling factors for its influence on the output pixel at position (i, j) is given by the following function:

$$m_{(k,l),(i,j)}(\theta) = \frac{\max(0, 1 - \sqrt{(k - i')^2 + (l - j')^2})}{\sum_{k',l'} \max(0, 1 - \sqrt{(k' - i')^2 + (l' - j')^2})} \qquad (10)$$

which is highly non-linear. It is 0 for most θ and for a very small range of θ, it takes non-zero values which can go up to 1. Thus, it makes naive verification infeasible. One idea is to use *Explicit Input Splitting* in Sec. 3.3 to split the input the range of θ into smaller parts and certify all parts, which will give a tighter bound since in smaller ranges the bounds are tighter. However, the required number of splits in *Explicit Input Splitting* may become too large, making it computationally infeasible. To balance this trade-off, we propose a new refinement technique named as *implicit input splitting* in the following section, which has light computational overhead and helps substantial boost in verification performance.

3.3. Input Space Refinement for Semantify-NN

To better handle highly non-linear functions that might arise from the general activation functions in the SP-layers, we propose two types of input-level refinement strategies. For linear-relaxation based verification methods, the following Theorem holds with the proof in the appendix.

Theorem 3.1. *Given an image x, if we can verify that a set S of perturbed images x' is correctly classified for a threat*

model using one certification cycle, then we can verify that every perturbed image x' in the convex hull of S is also correctly classified, where the convex hull in the pixel space.

Here one certification cycle means one pass through the certification algorithm sharing the same linear relaxation values. Although ℓ_p-norm balls are convex regions in pixel space, other threat models (especially semantic perturbations) usually do not have this property. This in turn poses a big challenge for semantic verification.

Remark 3.2. *For some non-convex attack spaces embedded in high-dimensional pixel spaces, the convex hull of the attack space associated with an image can contain images belonging to a different class (an example of rotation is illustrated in Figure 3 in the appendix). Thus, one cannot certify large intervals of perturbations using a single certification cycle of linear relaxation based verifiers.*

Explicit Input Splitting As we cannot certify large ranges of perturbation simultaneously, input-splitting is essential for verifying semantic perturbations. It reduces the gap between the linear bounds on activation functions and yields tighter bounds, as illustrated in appendix. We observe that

$$\min_{\boldsymbol{\epsilon}^k \in (I_1^k \cup I_2^k), h(\boldsymbol{\epsilon}^k) \leq \delta} \min_{j \neq c} f_c^{sem}(\boldsymbol{\epsilon}^k) - f_j^{sem}(\boldsymbol{\epsilon}^k) =$$

$$\min_{l \in \{1,2\}} \min_{\boldsymbol{\epsilon}^k \in I_l^k, h(\boldsymbol{\epsilon}^k) \leq \delta} \min_{j \neq c} f_c^{sem}(\boldsymbol{\epsilon}^k) - f_j^{sem}(\boldsymbol{\epsilon}^k)$$

If $\min_{\boldsymbol{\epsilon}^k \in I_l^k, h(\boldsymbol{\epsilon}^k) \leq \delta} (\min_j f_c^{sem}(\boldsymbol{\epsilon}^k) - f_j^{sem}(\boldsymbol{\epsilon}^k)) > 0$ holds for both parameter dimensions $l = \{1, 2\}$, then we have $\min_{\boldsymbol{\epsilon}^k \in I_1^k \cup I_2^k, h(\boldsymbol{\epsilon}^k) \leq \delta} (\min_j f_c^{sem}(\boldsymbol{\epsilon}^k) - f_j^{sem}(\boldsymbol{\epsilon}^k)) > 0$. As a result, we can split the original interval into smaller parts and certify each of them separately in order to certify the larger interval. The drawback of this procedure is that the computation time scales linearly with the number of divisions as one has to run the certification for every part. However, for the color space experiments we find that a few partitions are already sufficient for tight certificate.

Implicit Input Splitting As a motivating example, in Figure 4a in the appendix, we give the form of the activation function for rotation. Even in a small range of rotation angle θ ($2°$), the function is quite non-linear resulting in very loose linear bounds. As a result, we find that we are unable to get good verification results for datasets like MNIST and CIFAR-10 without increasing the number of partitions to very large values ($\approx 40,000$). This makes verification methods computationally infeasible. We aim at reducing the cost in *explicit splitting* by combining the intermediate bounds used by linear relaxation methods ([15]-[23]) to compute the suitable relaxation for the layer-wise non-linear activations. The idea is to use the shared linear bounds among

all sub-problems, and hence we only need to construct the matrices (Def 3.3-Cor 3.7 in [20]) $\mathbf{A}^{(k)}, \mathbf{T}^{(k)}, \mathbf{H}^{(k)}$ once for all S-subproblems instead of having different matrices for each subproblem. This helps to reduce the cost significantly from a factor of S to 1 when S is large (which is usually the case in order to get good refinement).

For the implementation, we split the original problem into S subproblem. To derive bounds on the output of a neuron at any given layer l, we calculate the pre-activation range for every subproblem. Then we merge the intervals of each neuron among all the subproblems (e.g. set $\boldsymbol{u}_r = \max_i \boldsymbol{u}_{r,\text{sub}i}, \boldsymbol{l}_r = \min_i \boldsymbol{l}_{r,\text{sub}i}$ in [20]) to construct the linear relaxation, $\mathbf{A}^{(k)}, \mathbf{T}^{(k)}, \mathbf{H}^{(k)}$ for the post-activation output of layer l. Continuing this procedure till the last layer gives the bounds on the output of the whole neural network.

Remark. Our experiments demonstrate that, for semantic perturbations, the refinement on the input space of the semantic space can significantly boost up the tightness of robustness certificate, as shown in Section 4. Although additional improvement can be made by refining the pre-activation bounds of each layer through solving a Linear Programming problem or Mixed integer optimization problem similar to the works in [19] and [17] for the ℓ_p-norm input perturbation, we observed that our proposed approach with input space refinement has already delivered a certified lower bound (of the minimal successful semantic perturbation) very close to the attack results (which is an upper bound), suggesting that the feature layer refinement will only have minor improvement while at the cost of much larger computation overhead (grows exponentially with number of nodes to be refined).

4. Experiments

We conduct extensive experiments for all the continuously-parametrized semantic attack threat models presented in the paper. The verification of discretely-parametrized semantic perturbations can be straightforward using enumeration, as discussed in Section 3.2. By applying our proposed method, Semantify-NN, one can leverage L_p-norm verification algorithms including [20, 23, 19, 16, 3]. We use verifiers proposed in [23] and [3] to certify multilayer perceptron (MLP) models and convolutional neural network (CNN) models as they are open-sourced, efficient, and support general activations on MLP and CNN models.

- **Baselines**. We calculate the upper bound and lower bound for possible value ranges of each pixel x_i of the original image given perturbation magnitude in the semantic space. Then, we use L_∞-norm based verifier to perform bisection on the perturbation magnitude and report its value. It is shown that directly converting the perturbation range from semantic space to original RGB space and then apply L_p-norm based verifiers give very poor results in all Tables.

Table 2: Evaluation of averaged bounds on HSL space perturbation. SPL denotes our proposed SP-layers. SPL + Refine refers to certificate obtained after using explicit splitting. Grid search on parameter space is used for attack. The results demonstrate the significance of using SPL layers for certification.

Network	Certified Bounds				Ours Improvement (vs Weighted)		Attack
	Naive	Weighted	SPL	SPL + Refine	w/o refine	w/ refine	Grid
Experiment (I)-A: Hue							
CIFAR, MLP 6×2048	0.00316	0.028	0.347	0.974	11.39x	51.00x	1.456
CIFAR, CNN 5×10	0.0067	0.046	0.395	1.794	7.58x	38.00x	1.964
GTSRB, MLP 4×256	0.01477	0.091	0.771	2.310	8.47x	22.31x	2.388
GTSRB MLP 4×256 sem adv	0.01512	0.092	0.785	2.407	8.53x	26.16x	2.474
Experiment (I)-B: Saturation							
CIFAR, MLP 6×2048	0.00167	0.004	0.101	0.314	24.25x	77.50x	0.342
CIFAR, CNN 5×10	0.00348	0.019	0.169	0.389	7.89x	19.47x	0.404
GTSRB, MLP 4×256	0.00951	0.020	0.38	0.435	19.00x	21.75x	0.444
GTSRB MLP 4×256 sem adv	0.00968	0.020	0.431	0.458	21.55x	22.90x	0.467
Experiment (I)-C: Lightness							
CIFAR, MLP 6×2048	0.00043	0.001	0.047	0.244	46.00x	243.00x	0.263
CIFAR, CNN 5×10	0.00096	0.002	0.080	0.273	39.00x	135.50x	0.303
GTSRB, MLP 4×256	0.0025	0.005	0.134	0.332	26.80x	66.40x	0.365
GTSRB MLP 4×256 sem adv	0.00268	0.005	0.148	0.362	29.80x	72.40x	0.398

We also give a weighted-eps version where we allow for different levels of perturbation for different pixels.

- **Attack**. We use a grid-search attack with the granularity of the order of the size of the sub-intervals after input splitting. Although this is not the optimal attack value, it is indeed an upper bound for the perturbation. Increasing the granularity would only result in a tighter upper bound and does not affect the lower bound (the certificate we deliver). We would like to highlight again that even though the threat models are very low dimensional, they are continuously parametrized and cannot be certified against by enumeration as discussed in Sec 2.

- **Semantify-NN**: We implement both SP-layers (**SPL**) and with refinement (**SPL+Refine**) described in Section 3.3.

Implementations, Models and Datasets. In all of our experiments, we use a custom google cloud instance with 24 vCPUs (Intel Xeon CPU @ 2.30GHz) and 90GB RAM. The SP-layers are added as fully connected layers for MLP's and as modified convolution blocks for CNN models (we allow the filter weights and biases to be different for different neurons). We evaluate Semantify-NN and other methods on MLP and CNN models trained on the MNIST, CIFAR-10 and GTSRB (German Traffic Sign Benchmark datasets). We use the MNIST and CIFAR models released by [6] and their standard test accuracy of MNIST/CIFAR models are 98-99%/60-70%. We train the GTSRB models from scratch to have 94-95% test accuracies. All CNNs (LeNet) use 3-by-3 convolutions and two max pooling layers and along with filter size specified in the description for two convolution layers each. LeNet uses a similar architecture to LeNet-5 [12], with the no-pooling version applying the same convolutions over

larger inputs. We also have two kinds of adversarially trained models. The models (denoted as sem adv in the Table) are trained using data augmentation where we add perturbed images (according to the corresponding threat model) to the training data. The models denoted as l_∞ adv are trained using the L_∞ norm adversarial training method [14]. We evaluate all methods on 200 random test images and random targeted attacks. We train all models for 50 epochs and tune hyperparameters to optimize validation accuracy.

Experiment (I): HSL Space Perturbations. Table 2 demonstrates that using L_p-norm based verification results in extremely loose bounds because of the mismatch in the dimensionality of the semantic attack and dimensionality of the induced L_p-norm attack. Explicitly introducing this dimensionality constraint by augmenting the neural networks with our proposed SP-layers gives a significant increase in the maximum certifiable lower bound, resulting in $4 - 50\times$ larger bounds. However, there is still an apparent gap between the Semantify-NN's certified lower bound and attack upper bound. Notably, we observe that adding input-space refinements helps us to further tighten the bounds, yielding an extra $1.5 - 5\times$ improvement. This corroborates the importance of input splitting for the certification against semantic attacks. The transformations for HSL space attacks are fairly linear, so the gap between our certified lower bound and attack upper bound becomes quite small.

Experiment (II): Rotation Table 3 shows the results of rotation space verification. Rotation induces a highly non-linear transformation on the pixel space, so we use this to illustrate the use of refinement for certifying such functions.

Table 3: Evaluation of averaged bounds on rotation space perturbation. SPL denotes our proposed SP-layers. The certified bounds obtained from SPL+Refine are close to the upper bounds from grid attack.

Network	Certified Bounds (degrees)				Attack (degrees)
	Number of Implicit Splits			**SPL + Refine**	Grid Attack
	1 implicit No explicit	5 implicit No explicit	10 implicit No explicit	100 implicit + explicit intervals of $0.5°$	
Experiment (II): Rotations					
MNIST, MLP 2×1024	0.627	1.505	2.515	46.24	51.42
MNIST, MLP 2×1024 l_∞ adv	1.376	2.253	2.866	45.49	46.02
MNIST, CNN LeNet	0.171	0.397	0.652	43.33	48.00
CIFAR, MLP 5×2048	0.006	0.016	0.033	14.81	37.53
CIFAR, CNN 5×10	0.008	0.021	0.042	10.65	30.81
GTSRB, MLP 4×256	0.041	0.104	0.206	31.53	33.43

Table 4: Evaluation of averaged bounds and run time on translation space perturbation and occlusions.

	Translation		Occlusion	
Network	Number of Pixels	Runtime (sec)	Filter Size	Runtime (sec)
Experiment (III): Translation & Occlusion				
MNIST, MLP 3×1024	4.063	0.46	11.080	0.44
MNIST, MLP 4×1024	4.130	0.51	10.778	0.55
MNIST, CNN LeNet	4.852	0.44	10.560	0.64
MNIST, CNN 4×5	5.511	0.47	11.590	0.71
CIFAR, MLP 5×2048	3.251	1.22	3.264	1.11
CIFAR, MLP 6×2048	2.755	1.36	3.300	1.28
CIFAR, CNN 5×10	4.532	0.86	3.373	0.66
GTSRB, MLP 4×256	2.785	0.39	2.652	0.14

Figure 2: Semantic certification for brightness and contrast.

As the transforms are very non-linear, the linear bounds used by our SP-layers are very loose, yielding very small robustness certification. In this case, explicit input splitting is not a computationally-appealing approach as there are a huge amount of intervals to be certified. Table 3 shows how using implicit splits can increase the size of certifiable intervals to the point where the total number of intervals needed is manageably big. At this point we use explicit splitting to get tight bounds. For the results in **SPL + Refine**, we use intervals of size 0.5 at a time with 100 implicit splits for each interval. The effect on the number of implicit splitting is discussed in Appendix B.

Experiment (III): Brightness and Contrast For multi-dimensional semantic attacks (here a combination attack using both brightness and contrast), we can consider any L_p norm of the parameters to be our distance function. In Figure 2 we show the results for average lower bound for brightness perturbations while fixing the maximum perturbation for contrast parameter (ϵ) to be 0.01, 0.03 and 0.05.

Experiment (IV): Translation and Occlusion Table 4 shows that in these experiments, the run-times needed for translation and occlusion to perform exhaustive enumeration are quite affordable. For translation we consider the attack
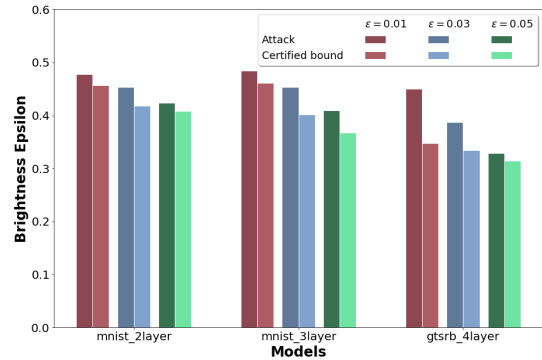
space with $I_1 = \{0, 1, \ldots, r\}$ and $I_2 = \{0, 1, \ldots, t\}$ where the images are of shape $r \times c$. The reported values are the average L_2 norm values for (ϵ_1, ϵ_2), the shift vector. For occlusion we consider the attack space with $I_1 = \{0, 1, \ldots, r\}$, $I_2 = \{0, 1, \ldots, t\}$ and $I_3 = \{1, 2, 3, \ldots, \max\{r, t\}\}$, where the images are of shape $r \times t$. The reported values are the average L_∞ norm values for ϵ_3 (the side-length of the square patch).

5. Conclusion

In this paper, we propose *Semantify-NN*, a model-agnostic and generic verification framework for neural network image classifiers against a broad spectrum of semantic perturbations. Semantify-NN exclusively features semantic perturbation layers (SP-layers) to expand the verification power of current verification methods beyond ℓ_p-norm bounded threat models. Based on a diverse set of semantic attacks, we demonstrate how the SP-layers can be implemented and refined for verification. Evaluated on various datasets, network architectures and semantic attacks, our experiments corroborate the effectiveness of *Semantify-NN* for semantic robustness verification.

References

[1] Mislav Balunovic, Maximilian Baader, Gagandeep Singh, Timon Gehr, and Martin Vechev. Certifying geometric robustness of neural networks. In *Advances in Neural Information Processing Systems*, pages 15287–15297, 2019. 2

[2] Anand Bhattad, Min Jin Chong, Kaizhao Liang, Bo Li, and David A Forsyth. Big but imperceptible adversarial perturbations via semantic manipulation. *arXiv preprint arXiv:1904.06347*, 2019. 1, 3

[3] Akhilan Boopathy, Tsui-Wei Weng, Pin-Yu Chen, Sijia Liu, and Luca Daniel. Cnn-cert: An efficient framework for certifying robustness of convolutional neural networks. In *AAAI*, Jan 2019. 1, 2, 3, 4, 6

[4] Krishnamurthy Dvijotham, Robert Stanforth, Sven Gowal, Timothy Mann, and Pushmeet Kohli. A dual approach to scalable verification of deep networks. *UAI*, 2018. 1, 2, 3

[5] Logan Engstrom, Brandon Tran, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. A rotation and a translation suffice: Fooling cnns with simple transformations. *arXiv preprint arXiv:1712.02779*, 2017. 1, 3

[6] Alhussein Fawzi and Pascal Frossard. Manitest: Are classifiers really invariant? *BMVC*, 2015. 1, 3

[7] Abdullah Hamdi and Bernard Ghanem. Towards analyzing semantic robustness of deep neural networks. *arXiv preprint arXiv:1904.04621*, 2019. 3

[8] Hossein Hosseini and Radha Poovendran. Semantic adversarial examples. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1614–1619, 2018. 1, 3

[9] Ameya Joshi, Amitangshu Mukherjee, Soumik Sarkar, and Chinmay Hegde. Semantic adversarial attacks: Parametric transformations that fool deep classifiers. *arXiv preprint arXiv:1904.08489*, 2019. 1, 3

[10] Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer. Reluplex: An efficient smt solver for verifying deep neural networks. In *International Conference on Computer Aided Verification*, pages 97–117. Springer, 2017. 1

[11] J Zico Kolter and Eric Wong. Provable defenses against adversarial examples via the convex outer adversarial polytope. *ICML*, 2018. 1, 2, 3

[12] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 7

[13] Hsueh-Ti Derek Liu, Michael Tao, Chun-Liang Li, Derek Nowrouzezahrai, and Alec Jacobson. Beyond pixel normballs: Parametric adversaries using an analytically differentiable renderer. *International Conference on Learning Representations*, 2019. 1, 3

[14] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *International Conference on Learning Representations*, 2018. 7

[15] Aditi Raghunathan, Jacob Steinhardt, and Percy S Liang. Semidefinite relaxations for certifying robustness to adversarial examples. In *Advances in Neural Information Processing Systems*, pages 10877–10887, 2018. 1, 6

[16] Gagandeep Singh, Timon Gehr, Matthew Mirman, Markus Püschel, and Martin Vechev. Fast and effective robustness certification. In *NeurIPS*, 2018. 1, 4, 6

[17] Gagandeep Singh, Timon Gehr, Markus Püschel, and Martin Vechev. Boosting robustness certification of neural networks. 2018. 2, 3, 6

[18] Gagandeep Singh, Timon Gehr, Markus Püschel, and Martin Vechev. An abstract domain for certifying neural networks. *Proceedings of the ACM on Programming Languages*, 3(POPL):41, 2019. 2, 3

[19] Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. Efficient formal safety analysis of neural networks. In *NeurIPS*, 2018. 1, 2, 3, 4, 6

[20] Tsui-Wei Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Duane Boning, Inderjit S Dhillon, and Luca Daniel. Towards fast computation of certified robustness for relu networks. *ICML*, 2018. 1, 2, 3, 4, 6

[21] Tsui-Wei Weng, Huan Zhang, Pin-Yu Chen, Jinfeng Yi, Dong Su, Yupeng Gao, Cho-Jui Hsieh, and Luca Daniel. Evaluating the robustness of neural networks: An extreme value theory approach. *arXiv preprint arXiv:1801.10578*, 2018. 1

[22] Eric Wong, Frank Schmidt, Jan Hendrik Metzen, and J Zico Kolter. Scaling provable adversarial defenses. *NeurIPS*, 2018. 1, 2, 3

[23] Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel. Efficient neural network robustness certification with general activation functions. In *NeurIPS*, dec 2018. 1, 2, 3, 4, 6