CHAPTER 10

Verification against semantic perturbations

Whereas 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, in this chapter, we introduce the *Semantify-NN* framework proposed by Mohapatra et al. (2020), a model-agnostic and generic robustness verification approach against semantic perturbations for neural networks. By simply inserting the 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 will 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. In addition, we discuss an efficient refinement technique to further improve the semantic certificate.

10.1 Semantic adversarial example

Beyond the ℓ_p-norm bounded threat model, some works have shown the possibility of generating *semantic adversarial examples* based on semantic perturbation techniques such as color shifting, lighting adjustment, and rotation (Hosseini and Poovendran, 2018; Liu et al., 2019b; Bhattad et al., 2019; Joshi et al., 2019; Fawzi and Frossard, 2015; Engstrom et al., 2017). We refer the readers to Fig. 10.1 for the illustration of some semantic perturbations for images. Notably, although semantically similar, these semantic adversarial attacks essentially consider threat models different from ℓ_p-norm bounded attacks in the RGB (red, green, and blue) 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.

In general, semantic adversarial attacks craft adversarial examples by tuning a set of parameters governing semantic manipulations of data sam-

Adversarial Robustness for Machine Learning
https://doi.org/10.1016/B978-0-12-824020-5.00020-X

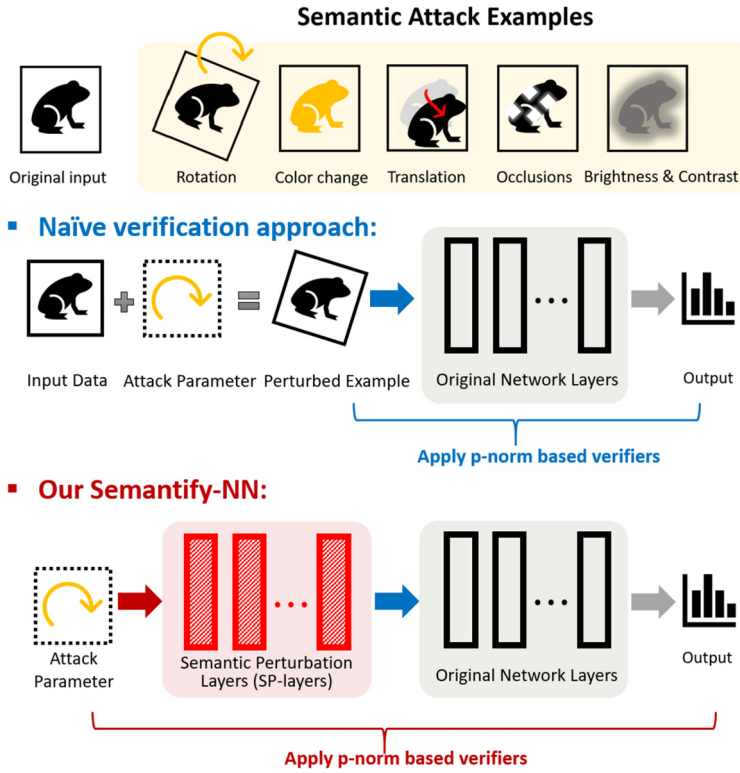


Figure 10.1 Schematic illustration of the *Semantify-NN* robustness verification framework in (Mohapatra et al., 2020). Given a semantic attack threat model, Semantify-NN designs the corresponding semantic perturbation layers (SP-layers) and inserts them to the input layer of the original network for verification. With SP-layers, Semantify-NN can use any ℓ_p-norm-based verification method for verifying semantic perturbations.

ples, which are either explicitly specified (e.g., rotation angle) or implicitly learned (e.g., latent representations of generative models). Hosseini and Poovendran (2018) use the HSV (hue, saturation, and value) representation of the RGB color space 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. Liu et al. (2019b) present a physically based differentiable renderer allowing propagating pixel-level gradients to the parametric space of lightness and geometry. Bhattad et al. (2019) introduce texture and colorization to induce semantic perturbation with a large ℓ_p norm perturbation to the raw pixel space while remaining visual

imperceptibility. Joshi et al. (2019) train an adversarial network composed of an encoder and a generator conditioned on attributes to find semantic adversarial examples. Fawzi and Frossard (2015) and Engstrom et al. (2017) show that simple operations such as image rotation or object translation can result in a notable misclassification rate.

We note that for the semantic perturbations that are *continuously parameterized* (such as hue, saturation, lightness, brightness, contrast, and rotations), 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, and hence it is not possible to finitely enumerate all possible values, unlike its discrete parameterized counterpart (e.g., translations and occlusions have finite enumerations). Taking 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 grid of 10^6 samples. However, if the attacks are not successful at $\theta = 30°$ and $30.00009°$, then it is possible that there exists 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 discussed in this chapter – with a proper semantic robustness verification algorithm, we aim to deliver a robustness *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.

10.2 Semantic perturbation layer

To bridge this gap and with an endeavor to render robustness verification methods more inclusive, Mohapatra et al. (2020) 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 the designed *semantic perturbation layers* (SP-layers). It is also generic since after adding SP-layers, we can apply any ℓ_p-norm-based verification tools for certifying semantic perturbations. In other words, the proposed SP-layers work as a carefully designed converter that transforms semantic threat models to ℓ_p-norm threat models.

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 follows: given a trained K-class neural net-

work function f and an input data sample x, we aim to find the largest δ such that

$$\min_{x' \in \Omega_{\mathcal{A}}(x), d_{\mathcal{A}}(x', x) \leq \delta} \left(\min_{j \neq c} f_c(x') - f_j(x') \right) > 0, \tag{10.1}$$

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

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 $\Omega^k_{\mathcal{A}}$, there exists a function $g : X \times (I_1 \times I_2 \times \cdots \times I_k) \to X$ such that

$$\begin{aligned} &\Omega^k_{\mathcal{A}}(x) = \{g(x, \epsilon_1, \ldots, \epsilon_k) \mid \epsilon_i \in I_i\}, \\ &d_{\mathcal{A}}(g(x, \epsilon_1, \ldots, \epsilon_k), x) = ||(\epsilon_1, \ldots, \epsilon_k)||_p, \end{aligned} \tag{10.2}$$

where X is the pixel space (the raw image), I_i is the set of feasible semantic operations, and $\| \cdot \|_p$ denotes the 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 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 $\boldsymbol{\epsilon}^k$ rather than the raw pixel space. Notice that the conventional ℓ_p norm perturbations on the raw RGB pixels are a particular case under this definition: by letting I_i equal to a bounded real set (i.e., $x'_i - x_i$, all possible difference between ith pixel) and k 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 parameterized semantic perturbation

Translation: Translation is a two-dimensional semantic attack with the parameters being the relative position of left-uppermost pixel of perturbed image to the original image. Therefore $I_1 = \{0, 1, \ldots, r\}$ and $I_2 =$

$\{0, 1, \ldots, h\}$, where r and h 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. Similarly to translation, occlusion attack can be expressed by three-dimensional attack parameters: the coordinates of the left-uppermost pixel of the occlusion patch and the occlusion patch size.[1] Note that for discretely parameterized semantic perturbations, provided with sufficient computation resources, we 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., $I^k \subset \mathbb{R}^k$. Mohapatra et al. (2020) 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 Fig. 10.1. By letting $g_x(\boldsymbol{\epsilon}^k) = g(x, \boldsymbol{\epsilon}^k)$ the verification problem for neural network f formulated in (10.1) becomes

$$\min_{\boldsymbol{\epsilon}^k \in I^k, d_{\mathcal{A}}(g_x(\boldsymbol{\epsilon}^k), x) \leq \delta} \left(\min_{j \neq c} f_c(g_x(\boldsymbol{\epsilon}^k)) - f_j(g_x(\boldsymbol{\epsilon}^k)) \right) > 0. \tag{10.3}$$

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

$$\min_{\boldsymbol{\epsilon}^k \in I^k, ||\boldsymbol{\epsilon}^k||_p \leq \delta} \left(\min_{j \neq c} f_c^{sem}(\boldsymbol{\epsilon}^k) - f_j^{sem}(\boldsymbol{\epsilon}^k) \right) > 0, \tag{10.4}$$

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

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

the manner and the effect of the semantic perturbation on different pixels of the image. In other words, we can view our proposed SP-layers as a parameterized input transformation function from the semantic space to RGB space, and $g(x, \boldsymbol{\epsilon}^k)$ 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 (Zhang et al., 2018; Wang et al., 2018a; Singh et al., 2018a; Boopathy et al., 2019; Weng et al., 2018a), as Semantify-NN only requires 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.

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 it is 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 the scale 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

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

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

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

$$\begin{aligned}\phi_R^h(h') &= 1 + \sigma_2(h') + \sigma_4(h') - (\sigma_5(h') + \sigma_1(h')),\\ \phi_G^h(h') &= \sigma_0(h') + \sigma_4(h') - (\sigma_1(h') + \sigma_3(h')),\\ \phi_B^h(h') &= \sigma_2(h') + \sigma_6(h') - (\sigma_5(h') + \sigma_3(h')).\end{aligned} \tag{10.5}$$

- *Saturation* This corresponds to the colorfulness of the picture. At saturation 0, we get grey-scale images, whereas 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

$$\phi^s(s') = \min(\max(s', 0), 1) = \sigma_0(s') - \sigma_1(s'). \tag{10.6}$$

- *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

$$\begin{aligned}\phi_1^l(l') &= 1 - |2 \cdot \min(\max(l', 0), 1) - 1|\\ &= -\sigma_0(2 \cdot l') - \sigma_2(2 \cdot l') + 2 \cdot \sigma_1(2 \cdot l') + 1,\\ \phi_2^l(l') &= \min(\max(l', 0), 1) = \sigma_0(l') - \sigma_1(l').\end{aligned} \tag{10.7}$$

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

$$\begin{aligned}g(x, \epsilon_b, \epsilon_c) &= \min(\max((1 + \epsilon_c) \cdot x + \epsilon_b, 0), 1)\\ &= \sigma_0((1 + \epsilon_c) \cdot x + \epsilon_b) - \sigma_1((1 + \epsilon_c) \cdot x + \epsilon_b).\end{aligned} \tag{10.8}$$

Therefore g can be expressed as one additional ReLU layer before the original network model, which is the proposed SP Layers in Fig. 10.1.

Rotation. We have a one-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 $i' = i\cos\theta - j\sin\theta$ and $j' = j\cos\theta + i\sin\theta$. Then

$$x'_{i,j} = \frac{\sum_{k,l} x_{k,l} \cdot \max(0, 1 - \sqrt{(k-i')^2 + (l-j')^2})}{\sum_{k,l} \max(0, 1 - \sqrt{(k-i')^2 + (l-j')^2})}, \tag{10.9}$$

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

$$m_{(k,l),(i,j)}(\theta) = \frac{\max(0, 1 - \sqrt{(k-i')^2 + (l-j')^2})}{\sum_{k',l'} \max(0, 1 - \sqrt{(k'-i')^2 + (l'-j')^2})}, \tag{10.10}$$

which is highly nonlinear. It is 0 for most θ, and for a very small range of θ, it takes nonzero values, which can go up to 1. Thus it makes naive verification infeasible. One idea is to use *Explicit Input Splitting* 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, Mohapatra et al. (2020) 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.

10.3 Input space refinement for semantify-NN

To better handle highly nonlinear functions that might arise from the general activation functions in the SP-layers, Mohapatra et al. (2020) propose two types of input-level refinement strategies. For linear-relaxation-based verification methods, they prove that 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. We remark that for some nonconvex

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. Thus we cannot certify large intervals of perturbations using a single certification cycle of linear relaxation-based verifiers.

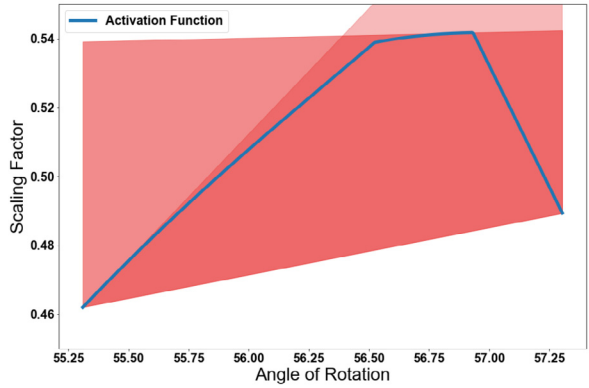
Explicit input splitting. As we cannot certify large ranges of perturbation simultaneously, input-splitting is essential for verifying semantic perturbations. It reduces the gap between the linear bounds on activation functions and yields tighter bounds. We observe that

$$\begin{aligned}&\min_{\boldsymbol{\epsilon}^k \in (I_1^k \cup I_2^k), h(\boldsymbol{\epsilon}^k) \leq \delta} \min_{j \neq c} f_c^{sem}(\boldsymbol{\epsilon}^k) - f_j^{sem}(\boldsymbol{\epsilon}^k) \\ = &\min_{l \in \{1,2\}} \min_{\boldsymbol{\epsilon}^k \in I_l^k, h(\boldsymbol{\epsilon}^k) \leq \delta} \min_{j \neq c} f_c^{sem}(\boldsymbol{\epsilon}^k) - f_j^{sem}(\boldsymbol{\epsilon}^k).\end{aligned}$$

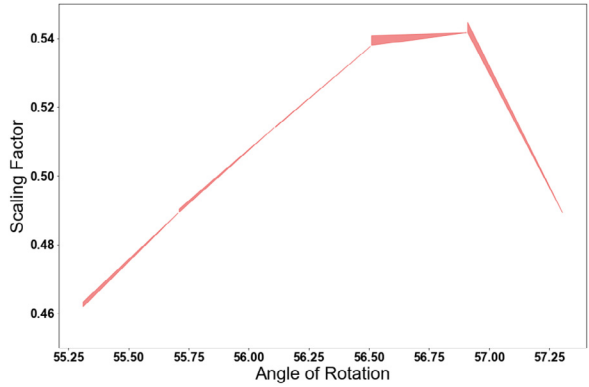
If $\min_{\boldsymbol{\epsilon}^k \in I_l^k, h(\boldsymbol{\epsilon}^k) \leq \delta}(\min_j f_c^{sem}(\boldsymbol{\epsilon}^k) - f_j^{sem}(\boldsymbol{\epsilon}^k)) > 0$ for both parameter dimensions $l = \{1, 2\}$, then we have $\min_{\boldsymbol{\epsilon}^k \in I_1^k \cup I_2^k, h(\boldsymbol{\epsilon}^k) \leq \delta}(\min_j f_c^{sem}(\boldsymbol{\epsilon}^k) - f_j^{sem}(\boldsymbol{\epsilon}^k)) > 0$. As a result, we can split the original interval into smaller parts and certify each of them separately 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, it is found that a few partitions are already sufficient for tight certificate.

Implicit input splitting. As a motivating example, in Fig. 10.2(a), we give the form of the activation function for rotation. Even in a small range of rotation angle θ (2°), the function is quite nonlinear, 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 such as (Raghunathan et al., 2018; Zhang et al., 2018) to compute the suitable relaxation for the layerwise nonlinear activations. The idea is to use the shared linear bounds among all subproblems, and hence we only need to construct the matrices (Def. 3.3 and Cor. 3.7 in Weng et al., 2018a) $\mathbf{A}^{(k)}$, $\mathbf{T}^{(k)}$, $\mathbf{H}^{(k)}$ 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 to get good refinement).

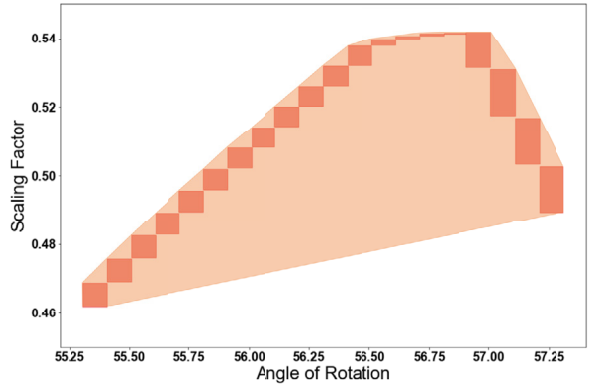
For implementation, we split the original problem into S-subproblems. To derive bounds on the output of a neuron at any given layer l, we cal-



(a) Without splitting the input range



(b) Explicit splitting the input (5 divisions)



(c) Implicit splitting the input (20 divisions)

Figure 10.2 Bounds for activation function of SP layer in rotation.

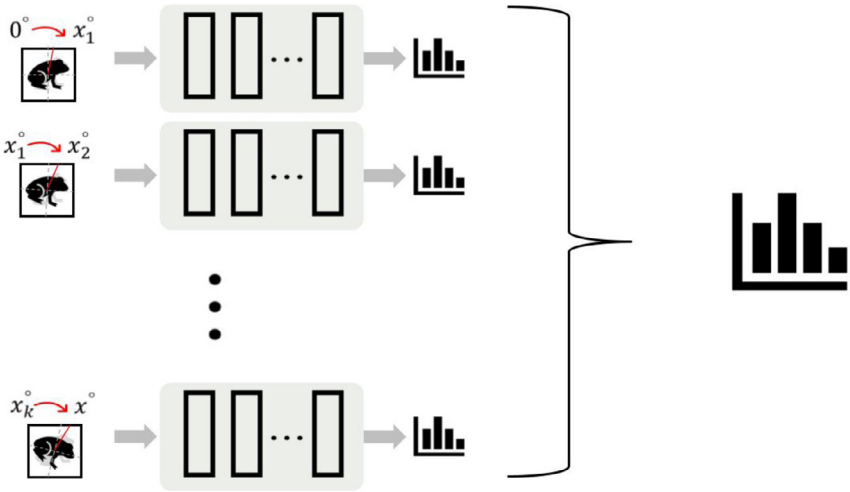
culate the preactivation range for every subproblem. Then we merge the intervals of each neuron among all the subproblems (e.g., set $\boldsymbol{u}_r = \max_i \boldsymbol{u}_{r,\text{sub}i}$ and $\boldsymbol{l}_r = \min_i \boldsymbol{l}_{r,\text{sub}i}$ in Weng et al. (2018a)) to construct the linear relaxation, $\mathbf{A}^{(k)}$, $\mathbf{T}^{(k)}$, $\mathbf{H}^{(k)}$ for the postactivation output of layer l. Continuing this procedure till the last layer gives the bounds on the output of the whole neural network.

Fig. 10.3 illustrates the difference between explicit and implicit input space splittings. Recall that in Fig. 10.2(a), we give the form of the activation function for rotation. Even in a small range of rotation angle θ (2°), we see that the function is quite nonlinear resulting in very loose linear bounds. Splitting the images explicitly into five parts and running them separately (i.e., explicit splitting as shown in Fig. 10.2(b)) give us a much tighter approximation. However, explicit splitting results in a high computation time as the time scales linearly with the number of splits. To efficiently approximate this function, we can instead make the splits to get explicit bounds on each subinterval and then run them through certification simultaneously (i.e., implicit splitting as shown in Fig. 10.2(c)). As we observe in Fig. 10.2(c), splitting into 20 implicit parts gives a very good approximation with very little overhead (the number of certification cycles used is still the same).

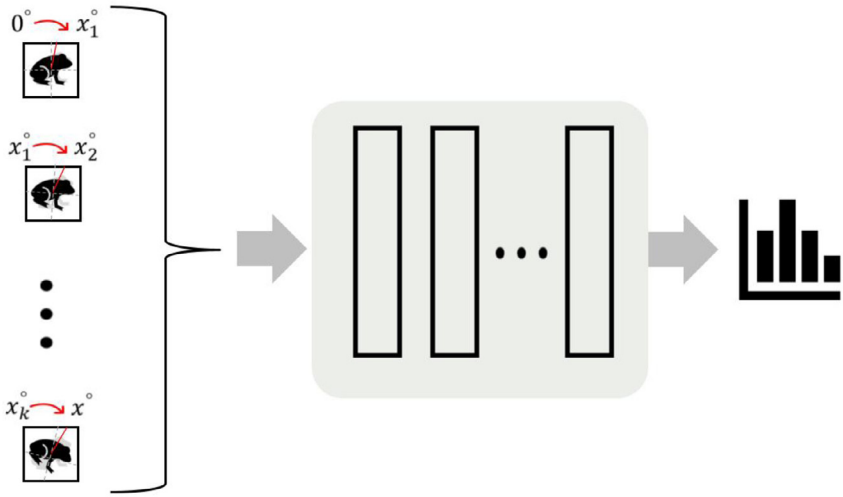
10.4 Empirical comparison

Following Mohapatra et al. (2020), we conduct extensive experiments for all the continuously parametrized semantic attack threat models presented in this chapter. The verification of discretely parameterized semantic perturbations can be straightforward using enumeration. We apply Semantify-NN to ℓ_p-norm verification algorithms proposed by Zhang et al. (2018) and Boopathy et al. (2019) to certify multilayer perceptron (MLP) models and convolutional neural network (CNN) models.

- *Baselines*. We calculate the upper and lower bounds for possible value ranges of each pixel x_i of the original image given perturbation magnitude in the semantic space. Then we use an ℓ_∞-norm-based verifier to perform bisection on the perturbation magnitude and report its value. We show that directly converting the perturbation range from semantic space to original RGB space and then applying ℓ_p-norm-based verifiers give very poor results in all tables. We also give a weighted-ϵ version, where we allow for different levels of perturbation for different pixels.



(a) Explicit splitting



(b) Implicit splitting

Figure 10.3 Illustration of refinement techniques.

- *Attack*. We use a grid-search attack with the granularity of the order of the size of the subintervals after input splitting. Although this is not the optimal attack value, it is indeed an upper bound for the mininum

adversarial perturbation. Increasing the granularity would only result in a tighter upper bound and does not affect the lower bound (the certificate we deliver).

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

The SP-layers are added as fully connected layers for MLPs 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 released MNIST and CIFAR models, 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 (LeCun et al., 1998), 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 (Madry et al., 2018). 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 10.1 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× 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× 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.

Table 10.1 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

Table 10.2 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

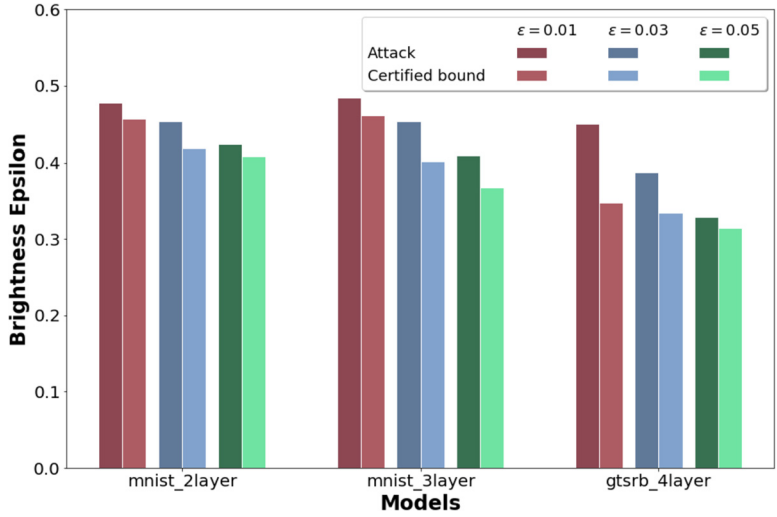


Figure 10.4 Semantic certification for brightness and contrast.

Experiment (II): Rotation. Table 10.2 shows the results of rotation space verification. Rotation induces a highly nonlinear transformation on the pixel space, so we use this to illustrate the use of refinement for certifying such functions. As the transforms are very nonlinear, 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 10.2 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.

Experiment (III): Brightness and contrast. For multidimensional 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 Fig. 10.4, 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.