
Robustness Evaluation and Adversarial Training of an Instance Segmentation Model

Jacob Bond

General Motors Company
jacob.bond@gm.com

Andrew Lingg

General Motors Company
andrew.lingg@gm.com

Abstract

To evaluate the robustness of non-classifier models, we propose probabilistic local equivalence, based on the notion of randomized smoothing, as a way to quantitatively evaluate the robustness of an arbitrary function. In addition, to understand the effect of adversarial training on non-classifiers and to investigate the level of robustness that can be obtained without degrading performance on the training distribution, we apply Fast is Better than Free adversarial training together with the TRADES robust loss to the training of an instance segmentation network. In this direction, we were able to achieve a symmetric best dice score of 0.85 on the TuSimple lane detection challenge, outperforming the standardly-trained network's score of 0.82. Additionally, we were able to obtain an F-measure of 0.49 on manipulated inputs, in contrast to the standardly-trained network's score of 0. We show that probabilistic local equivalence is able to successfully distinguish between standardly-trained and adversarially-trained models, providing another view of the improved robustness of the adversarially-trained models.

1 Introduction

While deep learning methods are able to achieve excellent performance on inputs originating from the same data distribution as the training data, generalization to additional distributions, including data which has been intentionally manipulated, remains a challenge for deep learning models. The poor generalizability of these models is a concern both in safety-critical situations, for example as deployment as part of a warehouse robot or an advanced driver-assistance system, but also in situations of great social or economic consequence, such as in financial credit assessment [3] or recidivism algorithms [1].

On the concern of input manipulation [23], adversarial training techniques have made great strides since their principled formulation by Mądry et al. [18], however, a large proportion of the work has focused on object classification and, to a lesser extent, object detection. There has been significantly less work investigating the adversarial training and robustness of generic machine learning tasks, despite the fact that many of the security-critical applications which demand adversarial robustness lie outside of the object classification domain.

Additionally, evaluation of a model's robustness typically focuses on evaluating the adversarial robustness of the model relative to the training data distribution, despite the frequent tendency for models to see a different data distribution in production. There is a need for additional methods for evaluating a model's robustness beyond just its performance on manipulated and unmanipulated inputs from the training distribution.

To this end, we reformulated randomized smoothing [7] as a robustness metric. In order to analyze the efficacy of the robustness metric, we compared the performance of standardly-trained instance segmentation models to models trained using two different approaches to adversarial training. Specifi-

cally, we adversarially trained a lane line instance segmentation model on the TuSimple lane detection challenge [25]. By adding a TRADES loss [33] regularization for the segmentation prediction to the adversarial training, we were able to improve the average symmetric best dice score of the model from 0.82 to 0.85, while also improving the average F-measure on manipulated imputs from 0 to 0.49. We found that the proposed robustness metric was able to differentiate between each of the three models, indicating its effectiveness.

2 Related Work

Since the establishment of adversarial training through a principled formulation by Mądry [18], a number of improvements have been proposed. Although early attempts at adversarial training using the fast gradient sign method (FGSM) [12] to generate perturbations were at best mildly successful [18, 24], Wong et al. [30] was able to establish that FGSM-based adversarial training with randomized initializations achieved comparable results to adversarial training based on projected gradient descent (PGD). The Fast is Better than Free (FBF) approach proposed by Wong et al. crucially reduced the training time of adversarial training to approximately double that of standard training, resulting in an overhead that is typically manageable even for production systems.

Another potentially significant drawback of early approaches to adversarial training was a reduction in accuracy on the original training distribution. This reduction has since been mitigated to a significant extent. Zhang et al. [33] augment the standard loss function of a classifier by adding a robust loss term meant to encourage the model to return similar results for both natural and manipulated inputs. Carmon et al. [5] utilized a two-stage approach using a standardly-trained model to label a collection of unlabeled data which was then added to the data available for adversarial training. This approach helped to encourage a model’s outputs to be stable around natural inputs. Gowal et al. [13] investigated a number of these improvements in combination, albeit sometimes requiring significant changes from the original model architecture or training procedure, finding that adding TRADES loss, model weight averaging, and unlabeled data all provide meaningful improvements to both the standard and adversarial performance of a robust model.

While adversarial training undeniably makes a model more robust, it remains a challenge to truly identify the impact on a model’s performance beyond natural and adversarial performance on the training distribution. One way to gauge a model’s robustness to L^p -norm-bounded perturbations is by plotting a security curve across a range of L^p -norm perturbation bounds, as in Adversarial Robustness Toolbox [11]. Alternatively, AutoAttack [8] provides a parameter-free method for evaluating a model’s robustness to input manipulations, though it is largely limited to object classification models.

Following the treatment by Cohen et al. [7], the randomized smoothing framework established a scalable and effective method for creating smoothed classifiers and certifying their robustness. Salman et al. [21] then developed an input manipulation attack on smoothed classifiers, using it for adversarial training to improve the provable robustness of these classifiers. While the original randomized smoothing method was restricted to classifiers, the method has been extended in several directions, the extension to image segmentation by Fischer et al. [10] being most relevant to this work.

In [31], Tsipras et al. show that there exists a data distribution on which it is impossible to learn an accurate and robust classifier. However, this is only a single and contrived example of such a phenomenon. Additionally, Tsipras et al. note that in the limited data regime, as in this paper, adversarial training can be beneficial to a model’s standard accuracy.

3 Lane Line Instance Segmentation

The model we use for our experiments is a lane line instance segmentation model with an architecture adapted from Neven et al. [19], who themselves adopt the instance segmentation approach of DeBrabandere et al. [4]. In [4], an embedded feature space is learned from the data in a branch of the network having the same backbone architecture, but separate parameters from, a binary segmentation branch. In contrast to [4], who base their instance segmentation network on a DeepLabv1 architecture with a ResNet-38 backbone [32], we chose to adapt the DeepLabv3+ architecture [6], replacing DeepLabv3+’s Xception backbone with ResNet-101 [14]. We made these choices based on the relative performance of different permutations of these models on other tasks of interest with similar datasets.

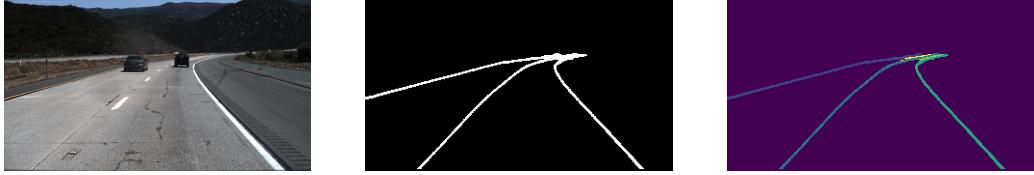


Figure 1: A model input (left), the resulting binary segmentation (middle), and the final instance segmentation (right)

As optimal model selection is not the subject of this work, we did not compare the performance of these permutations in adversarial conditions.

The binary segmentation branch of the network segments the pixels as belonging to either the background or a lane line, while the instance embedding branch embeds the pixels in a latent feature space. Each pixel is embedded in this feature space in a manner so that the pixels for each instance are near each other, while pixels from distinct instances are far away from each other. The loss function for the binary segmentation branch is the cross-entropy loss, while the instance embedding branch uses the discriminative loss function found in [4]. The model’s output is evaluated using the F-measure of the binary segmentation, as well as the symmetric best dice score [22] of the final instance segmentation.

4 Probabilistic Local Equivalence Certification

Comparing the innate robustness of two models is challenging. Evaluating the adversarial robustness of a model is often too aggressive, as any model which hasn’t been adversarially trained will receive a score of 0. Formally verifying a model typically requires an analysis limited to a specific architecture. We propose a method based on randomized smoothing certification [7] which is applicable to arbitrary functions while also being more sensitive than evaluation of adversarial robustness.

4.1 Probabilistic Local Equivalence Certification

A function f is robust if small changes to its input result in small changes to its output. To this end, given inputs x and x' , we will define a notion of how similar the outputs $f(x)$ and $f(x')$ are. Let $f : X \rightarrow Y$ be a function and let $\mathcal{M} : Y \times Y \rightarrow [0, 1]$ be a scoring function evaluating how similar two elements $y, y' \in Y$ are to each other. Then \mathcal{M} induces a semipseudometric d on X by

$$d(x, x') = 1 - \mathcal{M}(f(x), f(x')) \quad (1)$$

and the set $B_d(x, t) := \{x' \in X \mid d(x, x') < t\}$ defines the points $x' \in X$ so that $f(x')$ is close to $f(x)$ relative to \mathcal{M} . The robustness threshold t is ultimately an application-specific choice depending on a number of factors, but the goal is to capture the amount of variation in the model’s output which will avoid significant changes to any downstream results. An extreme example, as mentioned below, occurs when \mathcal{M} is the accuracy metric, in which case all values of $t \in (0, 1)$ are equivalent.

For d defined as in (1), the points in $B_d(x, t)$ can be considered as being equivalent to x relative to f , \mathcal{M} , and t since for suitable t , f gives equivalent outputs at all points of $B_d(x, t)$. Returning to the concept of robustness, f is robust if at all nearby inputs, f gives similar outputs. Translating this into the language introduced above, f is robust at x_0 for radius δ if $B_{L^2}(x_0, \delta) \subseteq B_d(x_0, t)$. While evaluating whether $B_{L^2}(x_0, \delta) \subseteq B_d(x_0, t)$ is difficult, the randomized smoothing framework provides a method for approximating this. Specifically, the framework allows us to determine whether

$$\text{for all } x \in B_{L^2}(x_0, \delta), \quad P_{\varepsilon \sim \mathcal{N}}(x + \varepsilon \in B_d(x_0, t)) > 0.5,$$

where \mathcal{N} is a normal distribution. In this direction, we have the following definitions.

Definition. *Relative to a choice of d , t , and distribution \mathcal{D} ,*

1. *If $B_{L^2}(x_0, \delta) \subseteq B_d(x_0, t)$, then f is locally equivalent to $f(x_0)$ at x for radius δ .*
2. *If $P_{\varepsilon \sim \mathcal{D}}(x + \varepsilon \in B_d(x_0, t)) \geq 0.5$, then f is probabilistically locally equivalent to $f(x_0)$ around x .*

Having defined d as in (1), fix a value of t and for any $x \in X$, let $B_x := B_d(x, t)$. The framework of [7, 21] will be applied to the indicator function $\mathbb{1}_{x_0} : x \mapsto \mathbb{1}(x \in B_{x_0})$. Specifically, let $\mathcal{N} := \mathcal{N}(0, \sigma^2 I)$, the normal distribution centered at 0 with standard deviation σ with cumulative distribution function Φ , and consider the smoothed function

$$\widehat{\mathbb{1}}_{x_0}(x) := \mathbb{E}_{\varepsilon \sim \mathcal{N}} \mathbb{1}_{x_0}(x + \varepsilon) = P_{\varepsilon \sim \mathcal{N}}(x + \varepsilon \in B_{x_0}). \quad (2)$$

Then the *Certify* algorithm presented in [7], when applied to $\widehat{\mathbb{1}}_{x_0}$, first determines a lower bound \underline{p} for $\widehat{\mathbb{1}}_{x_0}(x_0)$, which establishes that f is locally equivalent to $f(x_0)$ at x_0 with probability at least \underline{p} . However, [21] shows that $\Phi^{-1} \circ \widehat{\mathbb{1}}_{x_0}$ is $1/\sigma$ -Lipschitz, so that the value $\widehat{\mathbb{1}}_{x_0}(x_0)$ provides additional information about the surrounding neighborhood. In this direction, the *Certify* algorithm applied to $\widehat{\mathbb{1}}_{x_0}$ is then able to provide a radius R guaranteeing, with probability $1 - \alpha$, that f is probabilistically locally equivalent to $f(x_0)$ around all $x \in B_{L^2}(x_0, R)$. This algorithm is reframed for the current context as *CertifyProbabilisticEquivalence*; see [7] for additional details surrounding the original algorithm.

Algorithm 1 Certification of probabilistic local equivalence of a function f

```

1: function CERTIFYPROBABILISTICEQUIVALENCE( $f, x_0, y, \sigma, n, \alpha, \mathcal{M}, t$ )
2:  $y_0 \leftarrow f(x_0)$ 
3:  $x' \leftarrow n$  samples from  $\mathcal{N}(x, \sigma^2 I)$ 
4:  $y' \leftarrow f(x')$ 
5: if  $y$  is None
6:   num_equivalent  $\leftarrow \text{COUNTTRUE}(1 - \mathcal{M}(y_0, y') < t)$ 
7: else
8:   num_equivalent  $\leftarrow \text{COUNTTRUE}\left(1 - \frac{\mathcal{M}(y_0, y)}{\mathcal{M}(y', y)} < t\right)$ 
9:  $\underline{p} \leftarrow \text{LOWERCONFBOUND}(\text{num\_equivalent}, n, 1 - \alpha)$ 
10: if  $\underline{p} > \frac{1}{2}$  return radius  $\sigma\Phi^{-1}(\underline{p})$ 
11: else return ABSTAIN

```

Algorithm 2 Compute a robustness score of the function f

```

1: function ROBUSTNESSSCORE( $f, E, \sigma, n, \alpha, \mathcal{M}, t$ )
2: radii  $\leftarrow []$ 
3: for  $(x, y) \in E$  do
4:   radius  $\leftarrow \text{CERTIFYPROBABILISTICEQUIVALENCE}(f, x, y, \sigma, n, \alpha, \mathcal{M}, t)$ 
5:   if radius is ABSTAIN then Append 0 to radii
6:   else Append radius to radii
7: return MEAN(radii)

```

Definition. Let

$$\begin{aligned} \text{ProbLocEquiv}(S, f, r) &:= \{x_0 \in S \mid \\ &\quad f \text{ is probabilistically locally equivalent to } f(x_0) \text{ around all } x \in B_{L^2}(x_0, r)\}. \end{aligned}$$

Proposition 1. *With probability at least $1 - \alpha$ over the randomness in CertifyProbabilisticEquivalence, if CertifyProbabilisticEquivalence returns a radius R , then $x \in \text{ProbLocEquiv}(X, f, R)$.*

For a given radius r and evaluation set E , Proposition 1 can be used to determine the percentage of E which lie in $\text{ProbLocEquiv}(E, f, r)$, giving an indication of the robustness of the model to perturbations with an L^2 norm less than r . A robustness score, relative to a given evaluation set E and choice of hyperparameters, can then be computed for a function f by taking the mean of the certified radii over the set E . As in the case of randomized smoothing certification, labels for the evaluation set E are not required.

4.2 When Labels Are Available

In the case that ground-truth labels $\{y_i\}$ are available for points $\{x_i\} \subseteq X$, there is an alternative formulation for (1). In this case, rather than simply looking at the similarity between $f(x_1)$ and



Figure 2: The original input x , the model’s prediction y , and the ground truth label, resulting in a symmetric best dice score of 0.674.

$f(x_2)$, it may be desirable to determine how these values differ relative to the ground-truth y . When a pair $(x, y) \sim \mathcal{D} = X \times Y$, (1) can be replaced with

$$d_y(x, x') = 1 - \frac{\mathcal{M}(f(x'), y)}{\mathcal{M}(f(x), y)}. \quad (3)$$

The convention $d_y(x, x') = 1$ if $\mathcal{M}(f(x), y) = 0 = \mathcal{M}(f(x'), y)$ and $d_y(x, x') = -\infty$ if $\mathcal{M}(f(x), y) = 0 \neq \mathcal{M}(f(x'), y)$ will be adopted. While d_y is no longer even a semipseudometric, the set $B_{d_y}(x, t)$ contains the points x' so that replacing x by x' does not significantly degrade the model’s performance. In particular, note that if $\mathcal{M}(f(x'), y) > \mathcal{M}(f(x), y)$, then $d_y(x, x') < 0$ and $x' \in B_{d_y}(x, t)$.

4.3 The Case of Classification

Importantly, in the case of classifiers, the above measure of robustness reduces to the standard measure of robustness. In this case, \mathcal{M} is the accuracy metric:

$$\mathcal{M}(y_1, y_2) = \begin{cases} 1 & \text{if } y_1 = y_2, \\ 0 & \text{otherwise.} \end{cases}$$

Defining d as in (1),

$$x' \in B_d(x, t) \iff 1 - \mathcal{M}(f(x), f(x')) < t \iff 1 - t < \mathcal{M}(f(x), f(x')),$$

so that for $0 < t < 1$, $x' \in B_d(x, t) \iff f(x) = f(x')$.

4.4 The Case of Lane Line Segmentation

For lane line instance segmentation evaluation, we use symmetric best dice (SBD) as the score function \mathcal{M} and define d as in (3). Figures 2 and 3 illustrate the situation using a model trained following the approach in Section 5, showing an equivalent and an inequivalent example. Figure 2 shows the original input, the model’s output, and the ground truth label, leading to an SBD score of 0.674. Using a relative threshold of $t = 0.1$ and rearranging (3) results in

$$d_y(x, x') < 0.1 \iff 0.9\mathcal{M}(f(x), y) < \mathcal{M}(f(x'), y) \iff \mathcal{M}(f(x'), y) > 0.60066.$$

In Figure 3, the first example, using $\sigma = 0.2$, leads to an SBD score of 0.661, indicating that the model f has made an equivalent prediction on the given input. However, the second example, using $\sigma = 0.3$, results in an SBD score 0.573, outside of the threshold considered as equivalent.

5 Adversarial Training Image Segmentation

While the explicit premise of adversarial training [18] is to improve robustness of a network to small perturbations through data augmentation, the result is improved robustness beyond just the small perturbations included in the data augmentation. As noted in Engstrom et al. [9], as human perception is invariant to these perturbations, any model intended to mimic human perception should be invariant to them as well. In this way, training a model to be robust to these perturbations serves to improve the model’s alignment with human perception. Further, because the model is able to identify features which are invariant to the worst-case perturbations presented during adversarial training, the model should also be robust to other non-worst-case perturbations. As such, adversarial robustness is a desirable property for models to possess even beyond its importance for security considerations.

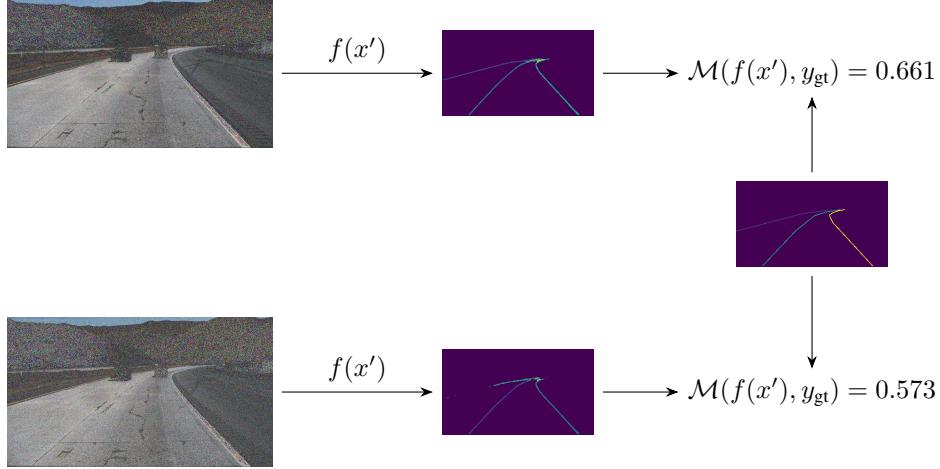


Figure 3: (Above) An input leading to a symmetric best dice score of 0.661, within the defined permissible threshold of 0.0674 from the original score 0.674. (Below) An input leading to a symmetric best dice score of 0.573, outside of the permissible threshold.

A significant improvement toward the use of adversarial training for production systems was made by Wong et al. [30], reducing the computation required for adversarial training, accomplishing this by leveraging perturbations generated using FGSM after initializing the perturbation at a random point within $B_\epsilon(0)$. Computing an update then requires only two backpropagation computations: one to compute the manipulated input and one to compute the weight update. The result is that adversarial training becomes tractable, even for relatively large models and models with production deadlines.

However, there remains a question of the performance of adversarially trained models on the original training data distribution. Given the entire training pipeline is optimized towards performing well on training distributions, it is unsurprising that adversarially-trained models within the same pipeline often struggle to match this performance when being asked to *also* perform well on a manipulated training distribution. On the other hand, as the inference-time distribution often varies from the training distribution, there is evidence that the adversarially-trained model may outperform standardly-trained models in the presence of this distribution shift [20].

Nevertheless, improvements to the base adversarial training procedure have helped to improve both the standard and adversarial performance of adversarially-trained models. One of the more common approaches to improving performance is the use of the TRADES loss function [33]. Zhang et al. adds a regularization term to improve the robustness of the model by seeking to maximize the distance of the decision boundary from each input. This is accomplished by minimizing the difference between the model’s output on an original input and a resulting manipulated input.

Because TRADES is more readily adapted to the binary segmentation output of our network, we use FBF as our base training method to provide robustness to the instance embedding branch. We then add the TRADES robust loss for the binary segmentation branch. Note that the original input x' received by the TRADES regularization has already been manipulated from the input x in the training set by FBF using FGSM. That input is then further manipulated by the TRADES regularization to obtain x'' , in this case to maximize the KL divergence between $f(x'')$ and $f(x')$. The total loss is then a linear combination of the model’s standard loss and the TRADES regularization term. Additional details are given in Algorithms 3 and 4. For information related to network-specific outputs or loss functions, see Section 3 and the corresponding references.

6 Experiments

6.1 Compute Resources Used for Experiments

Experiments were performed on an internal Kubernetes cluster containing NVIDIA DGX-1 machines. Each node on the cluster contains $8 \times$ NVIDIA Tesla V100 GPUs, $2 \times$ 20-Core Intel Xeon

Algorithm 3 FBF + TRADES training of an instance segmentation model f with parameters ϑ for T epochs, given perturbation bound ε , step size α , and a dataset of size M

```

1: function TRAIN( $f, \vartheta, X, Y$ )
2: for  $t = 1, \dots, T$  do
3:   for  $i = 1, \dots, M$  do
4:      $\delta \leftarrow \text{Uniform}(-\varepsilon, \varepsilon)$ 
5:      $\delta \leftarrow \Pi_{B_{L^\infty}(x_i, \varepsilon)} \delta + \alpha \cdot \text{sgn}(\nabla_\delta \text{LOSS}(f(x_i + \delta), y_i))$ 
6:      $\vartheta \leftarrow \vartheta - \nabla_\vartheta \text{LOSS}(f(x_i + \delta), y_i)$ 
7: return  $\vartheta$ 

```

Algorithm 4 Instance segmentation model loss with TRADES regularization for N steps, given step size η , perturbation bound ε , and regularization weight β

```

1: function LOSS( $f, x, y_{\text{seg}}, y_{\text{inst}}$ )
2: ( $\text{segmentation}, \text{instance\_embedding}$ )  $\leftarrow f(x)$ 
3:  $\text{seg\_loss} \leftarrow \text{CROSSENTROPYLOSS}(\text{segmentation}, y_{\text{seg}})$ 
4:  $\text{instance\_loss} \leftarrow \text{DISCRIMINATIVELOSS}(\text{instance\_embedding}, y_{\text{inst}})$ 
5: //Compute input for TRADES robust loss:
6:  $x' \leftarrow x + 0.001 \cdot \mathcal{N}(0, I)$ 
7: for  $j = 1, \dots, N$  do
8:   ( $\text{segmentation}', \text{instance\_embedding}'$ )  $\leftarrow f(x')$ 
9:    $x' \leftarrow \Pi_{B_{L^\infty}(x, \varepsilon)} x' + \eta \text{sgn} \nabla_{x'} \text{KLDIVLOSS}(\text{segmentation}, \text{segmentation}')$ 
10:  $\text{trades\_loss} \leftarrow \text{KLDIVLOSS}(\text{segmentation}, \text{segmentation}')$ 
11: return  $\text{seg\_loss} + \text{instance\_loss} + \beta \cdot \text{trades\_loss}$ 

```

E5-2698 v4 CPUs at a clock speed of 2.2 GHz for a total of 80 logical cores, and 512 GB of DDR4 RAM at a bus speed of 2.133 GHz. Each individual experiment conducted in this paper requested one of the Tesla V100 GPUs, 9 CPU cores, and 60 GB of RAM.

6.2 Carbon Emissions Related to Experiments

Altogether, training and validation of our final experiments, those displayed in Figure 4, totaled 7037 hours of GPU utilization (Appendix D). Based on the utilized GPU, experiment duration, and Michigan’s carbon efficiency of 0.4976 kg CO₂/kWh [27], the ML CO₂ Impact Calculator [16], as presented in [17], estimates the carbon emissions from these experiments to be 1050.49 kg CO₂. Additionally, testing the best model from each training method and computing the security and robustness curves in Figure 5 required 101.84 hours of computation on an NVIDIA Tesla P100 GPU, resulting in additional emissions of 12.67 kg CO₂. According to [29], our total emissions equates to 4298 km driven by an average internal combustion engine vehicle (0.247 kg CO₂/km) or 13 819 km driven by an average electric vehicle (0.077 kg CO₂/km; Appendix E).

6.3 Adversarial Training Setup

Training was performed using the implementation of FBF adversarial training in [11], modified for use with non-classifiers, using inputs in the space $[-1, 1]^{3 \times 720 \times 1280}$. The dataset used for development was the TuSimple lane detection challenge [25], with a train/validation/test split of 3082/181/363 images. We used the SGD optimizer with a momentum factor of 0.9, a perturbation bound of $\varepsilon = 8/255$ for both FBF and TRADES, $N = 10$ steps of size $\eta = \varepsilon/10$ for TRADES, and a regularization weighting of $\beta = 2.0 \times 10^{-5}$ that was tuned through experimentation. The TRADES repository [34] recommends $1 \leq \beta \leq 10$ for training a classifier for which the KL divergence is calculated over a one-dimensional array of class probabilities. As our KL divergence was calculated over a 320×180 -dimensional array of probabilities, this translates to a recommendation of $1.74 \times 10^{-5} \leq \beta \leq 17.4 \times 10^{-5}$, in line with our choice of β . We used L^∞ -norm perturbations for TRADES, corresponding to the use of FGSM in FBF adversarial training.

Because many aspects of AutoAttack [8] are targeted at classification, for adversarial evaluation we generated manipulated inputs using the untargeted Projected Gradient Descent attack with a

perturbation bound of $\varepsilon = 8/255$, a step size of $2/255$, 20 steps of gradient descent, and an L^∞ distance metric. For the evaluation phase of the FBF+TRADES model, the TRADES loss was not included in the computation of the manipulated inputs.

6.4 Results

To balance the competing objectives of maximizing both the F-measure of the binary segmentation result as well as the symmetric best dice (SBD) score of the instance segmentation result, we adopt a weighted sum approach to this multi-objective optimization problem, assigning a weight of 1 to each objective function. The resulting objective, evaluated on the validation set, is plotted as a function of the number of training epochs in Figure 4. Plots for the individual metrics are provided in Appendix B. In the case of a manipulated input, the typical behavior of PGD was to cause the model to classify the entire image as a single lane line, resulting in an F-measure of ≈ 0 . Due to the symmetric nature of SBD, this lane line class was associated with the background class of the ground-truth label, rather than any of the 3 lane line instances, resulting in an SBD score of $\approx 1/4$. This leads to a minimum adversarial $F_1 + \text{SBD}$ score of ≈ 0.25 .

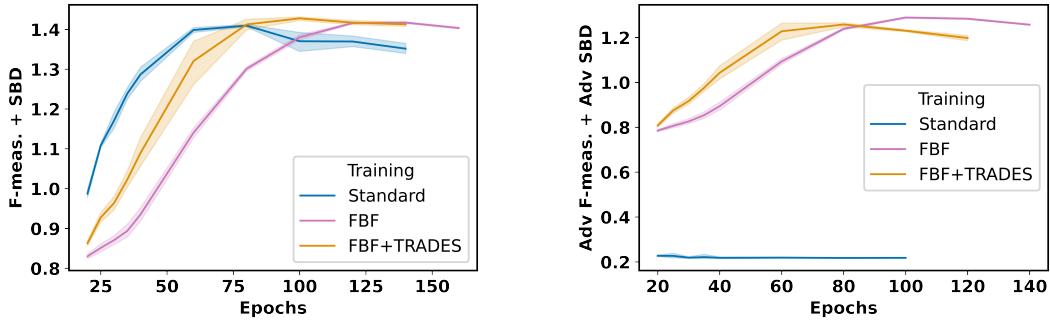


Figure 4: Performance of the models on natural (left) and adversarial (right) inputs from the validation set as a function of the number of training epochs

Ultimately, we found that while standard training was able to achieve a peak in performance more rapidly than FBF+TRADES or FBF, both FBF+TRADES and FBF were able to match this performance with sufficient training epochs. Notably, the performance on natural inputs of the best adversarially-trained models did not suffer as a result of adversarial training, in contrast to what is frequently observed when adversarially training on large object classification datasets. Both FBF+TRADES and FBF achieved similar levels of robustness, compared to no robustness from standard training, through FBF+TRADES was able to reach a peak in both natural and adversarial performance more quickly than FBF. Additionally, we found that after peaking, the F-measure of the models plateaued with additional training, while the SBD score began to drop as the models began overfitting to the training set (Appendix B).

Table 1: Performance of best model from each training method on the test set

Training Method	Natural		Adversarial		
	F-measure	SBD	F-measure	SBD	Time (hrs)
Standard (80 Epochs)	0.59	0.82	0.00	0.21	7.28
FBF ([30]; 140 Epochs)	0.58	0.84	0.49	0.76	27.01
FBF+TRADES (Ours; 100 Epochs)	0.59	0.85	0.48	0.76	22.78

For each training method, we selected the model which performed best on the validation set and listed its performance on the test set in Table 1. To better understand the robustness of the different models, we performed a more in-depth analysis of the robustness using the test set. We first performed an analysis of the models’ local equivalence robustness following the method in Section 4.4 and plotted the results in Figure 5a, finding that adversarial training improves the robustness of the model to the random perturbations generated in the evaluation. The local equivalence robustness analysis was

performed using $\sigma = 0.1$, $\alpha = 0.05$, and $n = 160$ samples per input. Figure 5a shows that local equivalence robustness is able to distinguish between the levels of robustness to normally distributed perturbations present in the standardly-trained and adversarially-trained models, and even between the different adversarially-trained models. We leave to a future work an analysis of how other robust networks, such as those robust to common corruptions [15], perform under this evaluation.

We also evaluated the models’ adversarial robustness. Although, we weren’t able to apply AutoAttack [8], to mitigate the effect of the perturbation budget ε as a hyperparameter, we plotted the security curve for each model, evaluating the model’s performance under a 10-step L^∞ PGD manipulation across a range of perturbation budgets in Figure 5b. As noted in [8], after a few iterations of PGD, the loss function plateaus (see Appendix F), so that 10-step PGD results in a good approximation of the optimum, while balancing the computational load required for a thorough sweep of perturbation budgets. For a given budget ε , a step size of $3\varepsilon/(2 \cdot 10)$ was used.

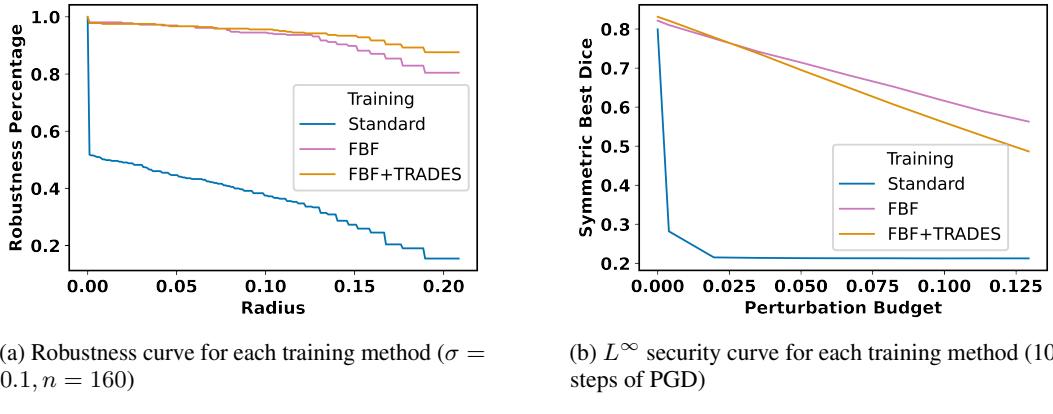


Figure 5: Robustness evaluation of the best performing model from each training method

From Figure 5, we see that while FBF training slightly outperforms FBF+TRADES training in robustness to L^∞ PGD attacks with budget $\varepsilon > 8/255$, FBF+TRADES provides improved robustness to the random perturbations used in the probabilistic local equivalence analysis compared to FBF. The security curve for L^2 PGD and the local equivalence curve for $\sigma = 0.25$ are given in Appendix C.

7 Conclusion

We introduced a method based on the theory of randomized smoothing [7] for assessing the robustness of an arbitrary function to randomized input perturbations. We showed probabilistic local equivalence is able to discriminate between a standardly-trained model and an adversarially-trained model. While the formulation of adversarial training implies robustness to such random perturbations, probabilistic local equivalence demonstrates that the standardly-trained model does not exhibit robustness to such random perturbations and that adversarial training is able to significantly improve the robustness of a model to this type of perturbation as well.

Further, we demonstrated that, in certain instances, adversarial training, whether FBF or FBF+TRADES, is able to match the performance of standard training on natural inputs. It is likely that the size of the network (2× ResNet-101 backbones) relative to the size of the training dataset (3082 images) played a role in these results. Nevertheless, our results show that the foregone assumption that adversarial training will necessarily reduce the performance of the network on the training data distribution does not hold in general. Rather, it stands to reason that the reduction in performance on the training distribution is most prevalent when the network is overfitting to the that distribution. In our case, where the size of the dataset relative to the model makes it likely the model will underfit, adversarial training served as a form of data augmentation improving the network’s performance on the training distribution and beyond. Alternatively, in the regime where overfitting is likely, adversarial training can help to prevent such overfitting to the training distribution.

References

- [1] J. Angwin, J. Larson, S. Mattu, and L. Kirchner. Machine bias. 2016.
- [2] E. Balkanski, H. Chase, K. Oshiba, A. Rilee, Y. Singer, and R. Wang. Adversarial attacks on binary image recognition systems, 2020.
- [3] L. Blattner and S. Nelson. How costly is noise? data and disparities in consumer credit, 2021.
- [4] B. D. Brabandere, D. Neven, and L. V. Gool. Semantic instance segmentation with a discriminative loss function, 2017.
- [5] Y. Carmon, A. Raghunathan, L. Schmidt, J. C. Duchi, and P. Liang. Unlabeled data improves adversarial robustness. In *Advances in Neural Information Processing Systems 32 (NeurIPS)*, pages 11190–11201, 2019.
- [6] L. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Computer Vision - ECCV 2018, 15th European Conference, Proceedings Part VII*, volume 11211 of *Lecture Notes in Computer Science*, pages 833–851, 2018.
- [7] J. M. Cohen, E. Rosenfeld, and J. Z. Kolter. Certified adversarial robustness via randomized smoothing. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pages 1310–1320, 2019.
- [8] F. Croce and M. Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pages 2206–2216, 2020.
- [9] L. Engstrom, A. Ilyas, S. Santurkar, D. Tsipras, B. Tran, and A. Mądry. Adversarial robustness as a prior for learned representations, 2019.
- [10] M. Fischer, M. Baader, and M. T. Vechev. Scalable certified segmentation via randomized smoothing. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, pages 3340–3351, 2021.
- [11] L. . D. Foundation. Adversarial robustness toolbox, 2018.
- [12] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations (ICLR)*, 2015.
- [13] S. Gowal, C. Qin, J. Uesato, T. Mann, and P. Kohli. Uncovering the limits of adversarial training against norm-bounded adversarial examples, 2021.
- [14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *29th IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.
- [15] D. Hendrycks, N. Mu, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. In *8th International Conference on Learning Representations (ICLR)*, 2020.
- [16] A. Lacoste, A. Lucioni, V. Schmidt, and T. Dandres. Machine Learning CO2 Impact Calculator, 2019.
- [17] A. Lacoste, A. Lucioni, V. Schmidt, and T. Dandres. Quantifying the carbon emissions of machine learning, 2019.
- [18] A. Mądry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations (ICLR)*, 2018.
- [19] D. Neven, B. D. Brabandere, S. Georgoulis, M. Proesmans, and L. V. Gool. Towards end-to-end lane detection: an instance segmentation approach. In *2018 IEEE Intelligent Vehicles Symposium*, pages 286–291. IEEE, 2018.

- [20] H. Salman, A. Ilyas, L. Engstrom, A. Kapoor, and A. Madry. Do adversarially robust imagenet models transfer better? In *Advances in Neural Information Processing Systems 33 (NeurIPS)*, 2020.
- [21] H. Salman, J. Li, I. P. Razenshteyn, P. Zhang, H. Zhang, S. Bubeck, and G. Yang. Provably robust deep learning via adversarially trained smoothed classifiers. In *Advances in Neural Information Processing Systems 32 (NeurIPS)*, pages 11289–11300, 2019.
- [22] H. Scharr, M. Minervini, A. P. French, C. Klukas, D. M. Kramer, X. Liu, I. Luengo, J. Pape, G. Polder, D. Vukadinovic, X. Yin, and S. A. Tsaftaris. Leaf segmentation in plant phenotyping: a collation study. *Mach. Vis. Appl.*, 27(4):585–606, 2016.
- [23] V. Shepardson, G. McGraw, H. Figueroa, and R. Bonett. A Taxonomy of ML Attacks, 2019.
- [24] F. Tramèr, A. Kurakin, N. Papernot, I. J. Goodfellow, D. Boneh, and P. D. McDaniel. Ensemble adversarial training: Attacks and defenses. In *6th International Conference on Learning Representations (ICLR)*, 2018.
- [25] TuSimple. Tusimple lane detection challenge. Joint Workshop on Computer Vision in Vehicle Technology and Autonomous Driving Challenge, 2017.
- [26] U.S. Department of Energy. Data Sources and Assumptions for the Electricity Sources and Emissions Tool, 2019.
- [27] U.S. Energy Information Administration. Michigan Electricity Profile 2020, 2021.
- [28] U.S. Energy Information Administration. United States Electricity Profile 2020, 2021.
- [29] U.S. Environmental Protection Agency. Greenhouse Gases Equivalencies Calculator – Calculations and References, 2021.
- [30] E. Wong, L. Rice, and J. Z. Kolter. Fast is better than free: Revisiting adversarial training. In *8th International Conference on Learning Representations (ICLR)*, 2020.
- [31] D. Tsipras, S. Santurkar, L. Engstrom, A. Turner, and A. Madry. Robustness May Be at Odds with Accuracy. In *7th International Conference on Learning Representations (ICLR)*, 2019.
- [32] Z. Wu, C. Shen, and A. van den Hengel. Wider or deeper: Revisiting the resnet model for visual recognition. *Pattern Recognit.*, 90:119–133, 2019.
- [33] H. Zhang, Y. Yu, J. Jiao, E. P. Xing, L. E. Ghaoui, and M. I. Jordan. Theoretically principled trade-off between robustness and accuracy. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pages 7472–7482, 2019.
- [34] H. Zhang, Y. Yu, J. Jiao, E. P. Xing, L. E. Ghaoui, and M. I. Jordan. TRADES (TRadeoff-inspired Adversarial DEfense via Surrogate-loss minimization), 2019.

A Proof of Proposition 1

Lemma 2. [21, Appendix A] Let $f : \mathbb{R}^n \rightarrow [0, 1]^2$ be defined by $f(x) = (f_0(x), f_1(x))$ with $f_i : \mathbb{R}^n \rightarrow [0, 1]$ and $f_0(x) + f_1(x) = 1$ for all $x \in \mathbb{R}^n$. Let $\varepsilon \sim \mathcal{N}(0, \sigma^2 I)$. If $\mathbb{E}_\varepsilon f_1(x + \varepsilon) \geq \frac{1}{2}$, then

$$\mathbb{E}_\varepsilon f_1(x + \delta + \varepsilon) \geq \frac{1}{2} \text{ for all } |\delta|_2 < \sigma \Phi^{-1} \circ f_1(x).$$

Proof. Appendix A in [21] shows that the lemma holds for all δ with

$$|\delta|_2 \geq \frac{1}{2} (\Phi^{-1} \circ f_1(x) - \Phi^{-1} \circ f_0(x)).$$

Noting that $1 - \Phi \circ \Phi^{-1} \circ f_1(x) = \Phi \circ \Phi^{-1}(-f_1(x))$,

$$\begin{aligned}\Phi^{-1} \circ f_0(x) &= \Phi^{-1}(1 - f_1(x)) = \Phi^{-1}(1 - \Phi \circ \Phi^{-1} \circ f_1(x)) \\ &= \Phi^{-1} \circ \Phi(-\Phi^{-1} \circ f_1(x)) = -\Phi^{-1} \circ f_1(x),\end{aligned}$$

and the result follows. \square

Proposition 1. For any function $f : \mathbb{R}^n \rightarrow [0, 1]$ and $x \in X \subseteq \mathbb{R}^n$, with probability at least $1 - \alpha$ over the randomness in CertifyProbabilisticEquivalence, if CertifyProbabilisticEquivalence returns a radius R , then $x \in \text{ProbLocEquiv}(X, f, R)$.

Proof. If CertifyProbabilisticEquivalence returns a radius R , then LowerConfBound returned a value $p = \Phi(R/\sigma) > 1/2$ and as a lower bound on the probability of success in the Bernoulli trial of being locally equivalent,

$$\begin{aligned}\underline{p} &\leq P_{y'}(1 - \mathcal{M}(y_0, y') < t) \\ &= P_{x'}(1 - \mathcal{M}(f(x_0), f(x')) < t) \\ &= P_{\varepsilon \sim \mathcal{N}}(1 - \mathcal{M}(f(x_0), f(x_0 + \varepsilon)) < t) \\ &= P_{\varepsilon \sim \mathcal{N}}(d(x_0, x_0 + \varepsilon) < t) \quad (\text{from (1)}) \\ &= P_{\varepsilon \sim \mathcal{N}}(x_0 + \varepsilon \in B_d(x_0, t)) \\ &= \widehat{\mathbb{1}}_{x_0}(x_0). \quad (\text{by (2)})\end{aligned}$$

Lemma 2 applied to the function $f = (1 - \mathbb{1}_{x_0}, \mathbb{1}_{x_0})$ then implies that for $x \in B_{L^2}(x_0, R)$, $\widehat{\mathbb{1}}_{x_0}(x) \geq 1/2$ and $x_0 \in \text{ProbLocEquiv}(X, f, R)$. \square

B Plots of F-measure and Symmetric Best Dice Score as a Function of Training Epochs

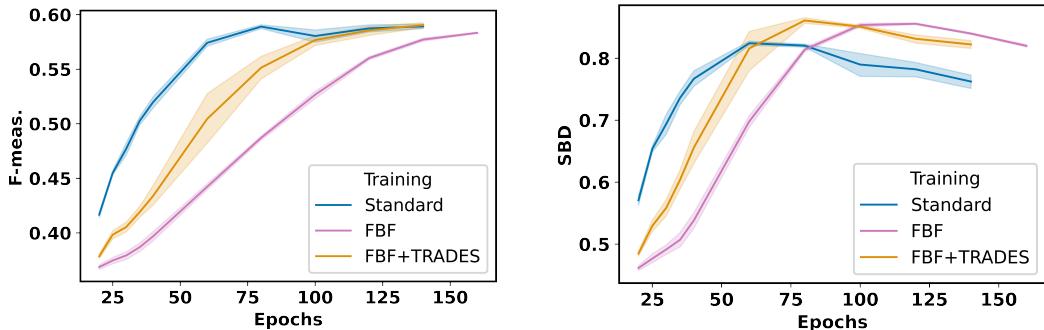


Figure 6: F-measure (left) and SBD (right) performance of the models on natural inputs from the validation set as a function of the number of training epochs

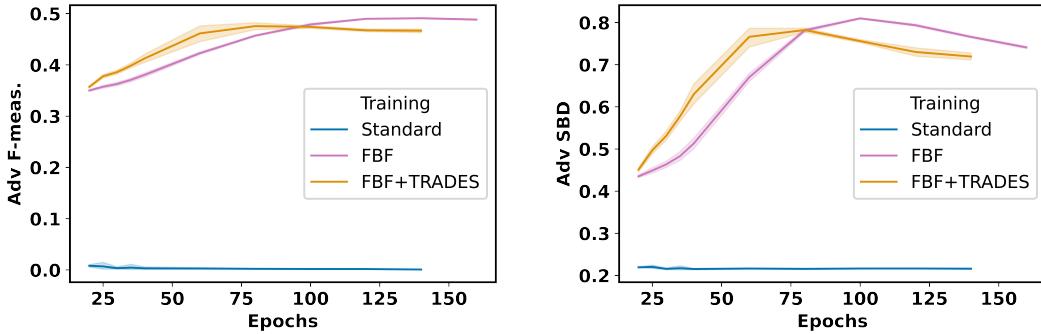


Figure 7: F-measure (left) and SBD (right) performance of the models on adversarial inputs from the validation set as a function of the number of training epochs

C Additional Adversarial Robustness Local Equivalence Plots

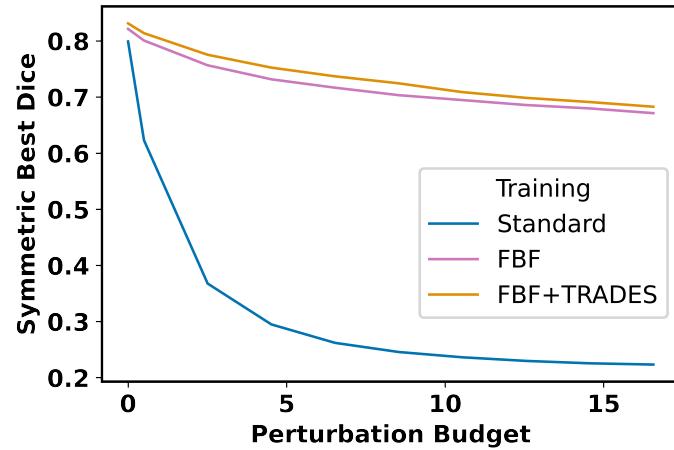


Figure 8: L^2 security curve for each training method (10 steps of PGD)

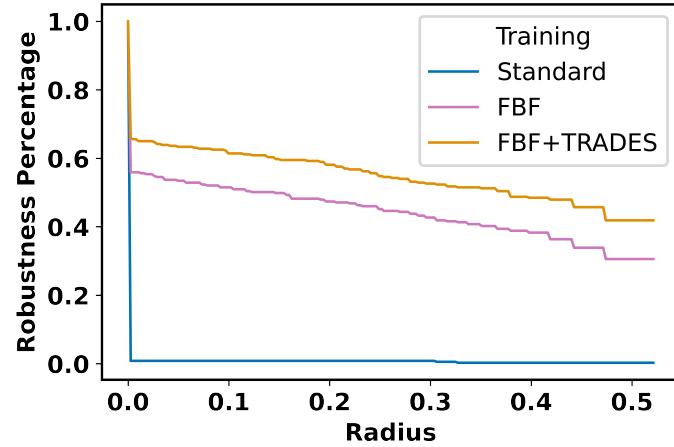


Figure 9: Robustness curve for each training method ($\sigma = 0.25, n = 160$)

D Calculation of Total Training & Evaluation Time

Table 2: Average training time for a given number of training epochs and training procedure

	Standard		FBF		FBF+TRADES	
	Avg. (min.)	#	Avg. (min.)	#	Avg. (min.)	#
20 Epochs	114.5	32	239.5	32	284.5	32
25 Epochs	141.5	16	300.0	32	353.0	32
30 Epochs	168.0	16	356.0	32	420.5	32
35 Epochs	194.0	16	411.5	16	486.0	16
40 Epochs	221.0	16	465.5	16	545.0	16
60 Epochs	327.0	16	696.5	16	831.5	16
80 Epochs	437.0	16	925.5	16	1087.5	16
100 Epochs	541.5	16	1184.5	16	1367.0	16
120 Epochs	758.0	16	1417.0	16	1636.0	16
140 Epochs	885.5	16	1620.5	16	1886.5	16
160 Epochs	0		1878.5	16	0	

Table 3: Total time spent training and evaluating networks during final experiments

Phase	Total Time (min.)
Training	413 936
Validation (14 min./network)	8288
Testing (incl. Robustness/ Security curves)	6111
Total	428 335

E Calculation of carbon emissions in terms of electric vehicle kilometers

Based on figures on US carbon efficiency [28] and electric vehicle energy efficiency [26], we find the following relationship between kg CO₂ and km driven by an electric vehicle:

$$1 \text{ kg CO}_2 \cdot \frac{2.205 \text{ lb}}{1 \text{ kg}} \cdot \frac{1 \text{ MWh}}{853 \text{ lb CO}_2} \cdot \frac{1000 \text{ kWh}}{1 \text{ MWh}} \cdot \frac{1 \text{ mi}}{0.32 \text{ kWh}} \cdot \frac{1.609 \text{ km}}{1 \text{ mi}} = 12.998 \text{ km}$$

F Effect of Number of PGD Iterations on Performance

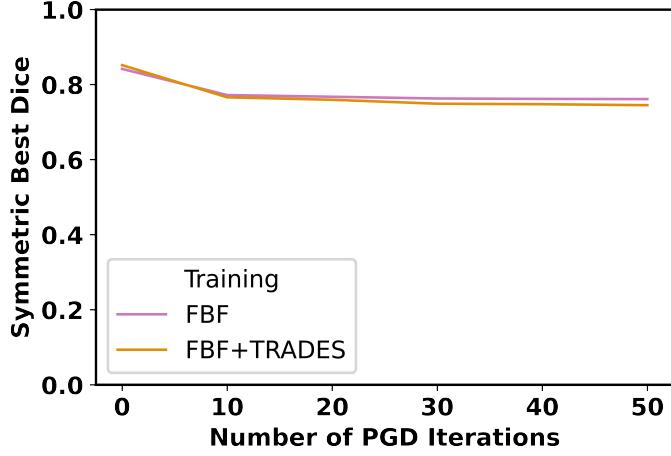


Figure 10: Model performance as a function of the number of PGD iterations ($\varepsilon = 8/255$)

G Societal Impacts and Limitations

The methods presented in this paper aim to improve the robustness of models, either by identifying non-robust models so their robustness can be improved or by training the models to be robust. A positive impact of our work is in identifying non-robust models before their deployment can lead to harm [2]. The effect of the methods presented will hopefully be to improve the overall performance of models to which they are applied. For systems with a positive impact on society, a more robust system should have a similarly positive impact. However, the inverse is also true in that for a system with a detrimental impact on society, a more robust system will likely exacerbate this negative impact.

As a framework, probabilistic local equivalence certification is highly customizable, requiring choices for similarity metric \mathcal{M} , robustness threshold t , and standard deviation σ , among other hyperparameters. The flexibility of the framework can also become a limitation as it can be difficult to decide on appropriate choices for each of these hyperparameters in a principled manner.