

Multitask Learning Strengthens Adversarial Robustness

Chengzhi Mao, Amogh Gupta*, Vikram Nitin*,
Baishakhi Ray, Shuran Song, Junfeng Yang, and Carl Vondrick

Columbia University, New York, NY, USA
mcz,rayb,shurans,junfeng,vondrick@cs.columbia.edu,
ag4202,vikram.nitin@columbia.edu

Abstract. Although deep networks achieve strong accuracy on a range of computer vision benchmarks, they remain vulnerable to adversarial attacks, where imperceptible input perturbations fool the network. We present both theoretical and empirical analyses that connect the adversarial robustness of a model to the number of tasks that it is trained on. Experiments on two datasets show that attack difficulty increases as the number of target tasks increase. Moreover, our results suggest that when models are trained on multiple tasks at once, they become more robust to adversarial attacks on individual tasks. While adversarial defense remains an open challenge, our results suggest that deep networks are vulnerable partly because they are trained on too few tasks.

Keywords: Multi-task Learning, Adversarial Robustness

1 Introduction

Deep networks obtain high performance in many computer vision tasks [20, 59, 33, 19], yet they remain brittle to adversarial examples. A large body of work has demonstrated that images with human-imperceptible noise [36, 4, 12, 41] can be crafted to cause the model to mispredict. This pervasiveness of adversarial examples exposes key limitations of deep networks, and hampers their deployment in safety-critical applications, such as autonomous driving.

A growing body of research has been dedicated to answering what causes deep networks to be fragile to adversarial examples and how to improve robustness [48, 14, 37, 36, 22, 56, 51, 43, 8, 50]. The investigations center around two factors: the training data and the optimization procedure. For instance, more training data – both labeled and unlabeled – improves robustness [44, 53]. It has been theoretically shown that decreasing the input dimensionality of data improves robustness [48]. Adversarial training [36] improves robustness by dynamically augmenting the training data using generated adversarial examples. Similarly,

* Equal Contribution. Order is alphabetical.



Fig. 1: We find that multitask models are more robust against adversarial attacks. Training a model to solve multiple tasks improves the robustness when one task is attacked. The middle and right column show predictions for single-task and multitask models when one task is adversarially attacked.

optimization procedures that regularize the learning specifically with robustness losses have been proposed [57, 8]. This body of work suggests that the fragility of deep networks may stem from the training data and optimization procedure.

In this paper, we pursue a new line of investigation: how learning on multiple tasks affects adversarial robustness. While previous work shows that multitask learning can improve the performance of specific tasks [5, 49], we show that it increases robustness too. See Figure 1. Unlike prior work that trades off performance between natural and adversarial examples [52], our work improves adversarial robustness while also maintaining performance on natural examples.

Using the first order vulnerability of neural networks [48], we theoretically show that increasing output dimensionality – treating each output dimension as an individual task – improves the robustness of the entire model. Perturbations needed to attack multiple output dimensions cancel each other out. We formally quantify and upper bound how much robustness a multitask model gains against a multitask attack with increasing output dimensionality.

We further empirically show that multitask learning improves the model robustness for two classes of attack: both when a single task is attacked or several tasks are simultaneously attacked. We experiment with up to 11 vision tasks on two natural image datasets, Cityscapes [9] and Taskonomy [61]. When all tasks are under attack, multitask learning increases segmentation robustness by up to 7 points and reduces the error of other tasks up to 60% over baselines. We compare the robustness of a model trained for a main task with and without an auxiliary task. Results show that, when the main task is under attack, multitask learning improves segmentation overlap by up to 6 points and reduces the error of the other tasks by up to 23%. Moreover, multitask training is a complementary defense to adversarial training, and it improves both the clean and adversarial performance of the state-of-the-art, adversarially trained, single-task models. Code is available at <https://github.com/columbia/MTRobust>.

Overall, our experiments show that multitask learning improves adversarial robustness while maintaining most of the the state-of-the-art single-task model performance. While defending against adversarial attacks remains an open prob-

lem, our results suggest that current deep networks are vulnerable partly because they are trained for too few tasks.

2 Related Work

We briefly review related work in multitask learning and adversarial attacks.

Multitask Learning: Multitask learning [5, 16, 11, 26, 49] aims to solve several tasks at once, and has been used to learn better models for semantic segmentation [29], depth estimation [54], key-point prediction [24], and object detection [30]. It is hypothesized that multitask learning improves the performance of select tasks by introducing a knowledge-based inductive bias [5]. However, multi-objective functions are hard to optimize, where researchers design architectures [21, 25, 35, 39, 31] and optimization procedures [6, 13, 46, 60] for learning better multitask models. Our work complements this body of work by linking multitask learning to adversarial robustness.

Adversarial Attacks: Current adversarial attacks manipulate the input [50, 12, 10, 3, 7, 47, 55, 38] to fool target models. While attacking single output models [18, 27] is straightforward, Arnab et. al. [3] empirically shows the inherent hardness of attacking segmentation models with dense output. Theoretical insight of this robustness gain, however, is missing in the literature. While past theoretical work showed the hardness of multi-objective optimization [17, 45], we leverage this motivation and prove that multitask models are robust when tasks are simultaneously attacked. Our work contributes both theoretical and empirical insights on adversarial attacks through the lens of multitask learning.

Adversarial Robustness: Adversarial training improves models’ robustness against attacks, where the training data is augmented using adversarial samples [18, 36]. In combination with adversarial training, later works [22, 37, 62, 56] achieve improved robustness by regularizing the feature representations with additional loss, which can be viewed as adding additional tasks. Despite the improvement of robustness, adversarially trained models lose significant accuracy on clean (unperturbed) examples [36, 62, 52]. Moreover, generating adversarial samples slows down training several-fold, which makes it hard to scale adversarial training to large datasets.

Past work revealed that model robustness is strongly connected to the gradient of the input, where models’ robustness is improved by regularizing the gradient norm [42, 57, 8]. Parseval [8] regularizes the Lipschitz constant—the maximum norm of gradient—of the neural network to produce a robust classifier, but it fails in the presence of batch-normalization layers. [42] decreases the input gradients norm. These methods can improve the model’s robustness without compromising clean accuracy. Simon-Gabriel et al. [48] conducted a theoretical analysis of the vulnerability of neural network classifiers, and connected gradient norm and adversarial robustness. Our method enhances robustness by training a multitask model, which complements both adversarial training [36, 62] and existing regularization methods [40, 42, 57, 8].

3 Adversarial Setting

The goal of an adversary is to “fool” the target model by adding human-imperceptible perturbations to its input. We focus on untargeted attacks, which are harder to defend against than targeted attacks [15]. We classify adversarial attacks for a multitask prediction model into two categories: adversarial attacks that fool more than one task at once (multitask attacks), and adversarial attacks that fool a specific task (single-task attacks).

3.1 Multitask Learning Objective

Notations. Let \mathbf{x} denote an input example, and \mathbf{y}_c denote the corresponding ground-truth label for task c . In this work, we focus on multitask learning with shared parameters [25, 49, 34, 32, 28], where all the tasks share the same “backbone network” $F(\cdot)$ as a feature extractor with task-specific decoder networks $D_c(\cdot)$. The task-specific loss is formulated as:

$$\mathcal{L}_c(\mathbf{x}, \mathbf{y}_c) = \ell(D_c(F(\mathbf{x})), \mathbf{y}_c), \quad (1)$$

where ℓ is any appropriate loss function. For simplicity, we denote $(\mathbf{y}_1, \dots, \mathbf{y}_M)$ as $\bar{\mathbf{y}}$, where M is the number of tasks. The total loss for multitask learning is a weighted sum of all the individual losses:

$$\mathcal{L}_{all}(\mathbf{x}, \bar{\mathbf{y}}) = \sum_{c=1}^M \lambda_c \mathcal{L}_c(\mathbf{x}, \mathbf{y}_c) \quad (2)$$

For the simplicity of theoretical analysis, we set $\lambda_c = \frac{1}{M}$ for all $c = 1, \dots, M$, such that $\sum_{c=1}^M \lambda_c = 1$. In our experiments on real-world datasets, we will adjust the λ_c accordingly, following standard practice [49, 28].

3.2 Adversarial Multitask Attack Objective

The goal of a multitask attack is to change multiple output predictions together. For example, to fool an autonomous driving model, the attacker may need to deceive both the object classification and depth estimation tasks. Moreover, if we regard each output pixel of a semantic segmentation task as an individual task, adversarial attacks on segmentation models need to flip multiple output pixels, so we consider them as multitask attacks. We also consider other dense output tasks as a variant of multitask, such as depth estimation, keypoints estimation, and texture prediction.

In general, given an input example \mathbf{x} , the objective function for multitask attacks against models with multiple outputs is the following:

$$\operatorname{argmax}_{\mathbf{x}_{adv}} \mathcal{L}_{all}(\mathbf{x}_{adv}, \bar{\mathbf{y}}) \quad \text{s.t.} \quad \|\mathbf{x}_{adv} - \mathbf{x}\|_p \leq r \quad (3)$$

where the attacker aims to maximize the joint loss function by finding small perturbations within a p -norm bounded distance r of the input example. Intuitively, a multitask attack is not easy to perform because the attacker needs to optimize the perturbation to fool each individual task simultaneously. The robustness of the overall model can be a useful property - for instance, consider an autonomous-driving model trained for both classification and depth estimation. If either of the two tasks is attacked, the other can still be relied on to prevent accidents.

3.3 Adversarial Single-Task Attack Objective

In contrast to a multitask attack, a single-task attack focuses on a selected target task. Compared with attacking all tasks at once, this type of attack is more effective for the target task, since the perturbation can be designed solely for this task without being limited by other considerations. It is another realistic type of attack because some tasks are more important than the others for the attacker. For example, if the attacker successfully subverts the color prediction for a traffic light, the attacker may cause an accident even if the other tasks predict correctly. The objective function for single-task attack is formulated as:

$$\operatorname{argmax}_{\mathbf{x}_{adv}} \mathcal{L}_c(\mathbf{x}_{adv}, \mathbf{y}_c), \text{ s.t. } \|\mathbf{x}_{adv} - \mathbf{x}\|_p \leq r \quad (4)$$

For any given task, this single-task attack is more effective than jointly attacking the other tasks. We will empirically demonstrate that multitask learning also improves model robustness against this type of attack in Section 5.

4 Theoretical Analysis

We present theoretical insights into the robustness of multitask models. A prevalent formulation of multitask learning work uses shared backbone network with task-specific branches [34, 32, 28]. We denote the multitask predictor as F and each individual task predictor as F_c . Prior work [48] showed that the norm of gradients captures the vulnerability of the model. We thus measure the multitask models' vulnerability with the same metric. Since we are working with deep networks, we assume all the functions here are differentiable. Details of all proofs are in the supplementary material.

Definition 1. *Given classifier F , input \mathbf{x} , output target \mathbf{y} , and loss $\mathcal{L}(\mathbf{x}, \mathbf{y}) = \ell(F(\mathbf{x}), \mathbf{y})$, the feasible adversarial examples lie in a p -norm bounded ball with radius r , $B(\mathbf{x}, r) := \{\mathbf{x}_{adv}, \|\mathbf{x}_{adv} - \mathbf{x}\|_p < r\}$. Then adversarial vulnerability of a classifier over the whole dataset is*

$$\mathbb{E}_{\mathbf{x}}[\Delta\mathcal{L}(\mathbf{x}, \mathbf{y}, r)] = \mathbb{E}_{\mathbf{x}}\left[\max_{\|\delta\|_p < r} |\mathcal{L}(\mathbf{x}, \mathbf{y}) - \mathcal{L}(\mathbf{x} + \delta, \mathbf{y})|\right]$$

$\Delta\mathcal{L}$ captures the maximum change of the output loss from arbitrary input change δ inside the p -norm ball. Intuitively, a robust model should have smaller change of the loss given any perturbation of the input. Given the adversarial noise is imperceptible, i.e., $r \rightarrow 0$, we can approximate $\Delta\mathcal{L}$ with a first-order Taylor expansion [48].

Lemma 1. *For a given neural network F that predicts multiple tasks, the adversarial vulnerability is*

$$\mathbb{E}_{\mathbf{x}}[\Delta\mathcal{L}(\mathbf{x}, \mathbf{y}, r)] \approx \mathbb{E}_{\mathbf{x}} [\|\partial_{\mathbf{x}}\mathcal{L}_{all}(\mathbf{x}, \bar{\mathbf{y}})\|_q] \cdot \|\delta\|_p \propto \mathbb{E}_{\mathbf{x}} [\|\partial_{\mathbf{x}}\mathcal{L}_{all}(\mathbf{x}, \bar{\mathbf{y}})\|_q]$$

where q is the dual norm of p , which satisfies $\frac{1}{p} + \frac{1}{q} = 1$ and $1 \leq p \leq \infty$. Without loss of generality, let $p = 2$ and $q = 2$. Note that from equation 2, we get $\mathcal{L}_{all}(\mathbf{x}, \bar{\mathbf{y}}) = \sum_{c=1}^M \frac{1}{M} \mathcal{L}_c(\mathbf{x}, \mathbf{y}_c)$. Thus we get the following equation:

$$\partial_{\mathbf{x}}\mathcal{L}_{all}(\mathbf{x}, \bar{\mathbf{y}}) = \partial_{\mathbf{x}} \sum_{c=1}^M \frac{1}{M} \mathcal{L}_c(\mathbf{x}, \mathbf{y}_c) = \frac{1}{M} \sum_{c=1}^M \partial_{\mathbf{x}}\mathcal{L}_c(\mathbf{x}, \mathbf{y}_c) \quad (5)$$

We denote the gradient for task c as \mathbf{r}_c , i.e., $\mathbf{r}_c = \partial_{\mathbf{x}}\mathcal{L}_c(\mathbf{x}, \mathbf{y}_c)$. We propose the following theory for robustness of different numbers of randomly selected tasks.

Theorem 1. (Adversarial Vulnerability of Model for Multiple Correlated Tasks) *If the selected output tasks are correlated with each other such that the covariance between the gradient of task i and task j is $\text{Cov}(\mathbf{r}_i, \mathbf{r}_j)$, and the gradient for each task is i.i.d. with zero mean (because the model is converged), then adversarial vulnerability of the given model is proportional to*

$$\sqrt{\frac{1 + \frac{2}{M} \sum_{i=1}^M \sum_{j=1}^{i-1} \frac{\text{Cov}(\mathbf{r}_i, \mathbf{r}_j)}{\text{Cov}(\mathbf{r}_i, \mathbf{r}_i)}}{M}}$$

where M is the number of output tasks selected.

The idea is that when we select more tasks as attack targets, the gradients for each of the individual tasks on average cancels out with each other. We define the joint gradient vector \mathbf{R} as follows:

$$\mathbf{R} = \partial_{\mathbf{x}}\mathcal{L}_{all}(\mathbf{x}, \bar{\mathbf{y}}) = \frac{1}{M} \sum_{c=1}^M \partial_{\mathbf{x}}\mathcal{L}_c(\mathbf{x}, \mathbf{y}_c)$$

The joint gradient is the sum of gradients from each individual task. We then obtain the expectation of the L_2 norm of the joint gradient:

$$\begin{aligned} \mathbb{E}(\|\mathbf{R}\|_2^2) &= \mathbb{E} \left[\left\| \frac{1}{M} \sum_{i=1}^M \mathbf{r}_i \right\|_2^2 \right] = \frac{1}{M^2} \mathbb{E} \left[\sum_{i=1}^M \|\mathbf{r}_i\|^2 + 2 \sum_{i=1}^M \sum_{j=1}^i \mathbf{r}_i \mathbf{r}_j \right] \\ &= \frac{1}{M^2} \left(\sum_{i=1}^M \mathbb{E}[\text{Cov}(\mathbf{r}_i, \mathbf{r}_i)] + 2 \sum_{i=1}^M \sum_{j=1}^i \mathbb{E}[\text{Cov}(\mathbf{r}_i, \mathbf{r}_j)] \right) \end{aligned}$$

The last equation holds due to the $\mathbf{0}$ mean assumption of the gradient. For further details of the proof, please see the supplementary material.

Corollary 1. (Adversarial Vulnerability of Model for Multiple Independent Tasks) *If the output tasks selected are independent of each other, and the gradient for each task is i.i.d. with zero mean, then the adversarial vulnerability of given model is proportional to $\frac{1}{\sqrt{M}}$, where M is the number of independent output tasks selected.*

Based on the independence assumption, all covariances becomes zero. Thus Theorem 2 can be simplified as:

$$\mathbb{E}[\|\partial_{\mathbf{x}}\mathcal{L}_{all}(\mathbf{x}, \bar{\mathbf{y}})\|_2^2] = \mathbb{E}(\|\mathbf{R}\|_2^2) = \frac{1}{M}\mathbb{E}\|\mathbf{r}_i\|^2 = \frac{\sigma^2}{M} \propto \frac{1}{M} \quad (6)$$

Remark 1. By increasing the number of output tasks M , the first order vulnerability [48] of network decreases. In the ideal case, if the model has an infinite number of uncorrelated tasks, then it is impossible to find an adversarial examples that fools all the tasks.

Remark 2. Theorem 1 studies the robustness for multiple **correlated** tasks, which is true for most computer vision tasks. The independent tasks assumption in Corollary 1 is a simplified, idealistic instance of Theorem 1 that upper-bounds the robustness of models under multitask attacks. Together, Theorem 1 and Corollary 1.1 demonstrate that unless the tasks are 100% correlated (the same task), multiple tasks together are more robust than each individual one.

Our theoretical analysis shows that more outputs, especially if they are less correlated, improve the model’s robustness against multitask attacks. Past work shows that segmentation is inherently robust [3, 7] compared to classification. Our analysis provides a formal explanation to this inherent robustness because a segmentation model can be viewed as a multitask model (one task per pixel).

5 Experiments

We validate our analysis with empirical results on the Cityscapes and the Taskonomy datasets. We evaluate the robustness of multitask models against two types of attack: multitask attack (Section 5.3) and single-task attacks (Section 5.4). We also conduct multitask learning experiments on adversarial training and show that they are complementary (Section 5.5).

5.1 Datasets

Cityscapes. The Cityscapes dataset [9] consists of images of urban driving scenes. We study three tasks: semantic segmentation, depth estimation, and image reconstruction. We use the full resolution (2048×1024) for analyzing pre-trained state-of-the-art models. We resize the image to 680×340 to train our single task (baseline) and multitask models.¹

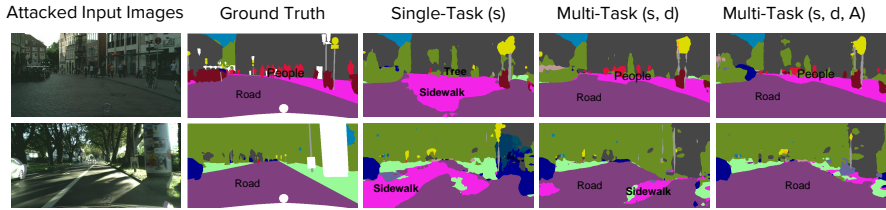


Fig. 2: We show model predictions on Cityscapes under multitask attack. The single-task segmentation model misclassifies the ‘road’ as ‘sidewalk’ under attack, while the multitask model can still segment it correctly. The multitask models are more robust than the single-task trained model.

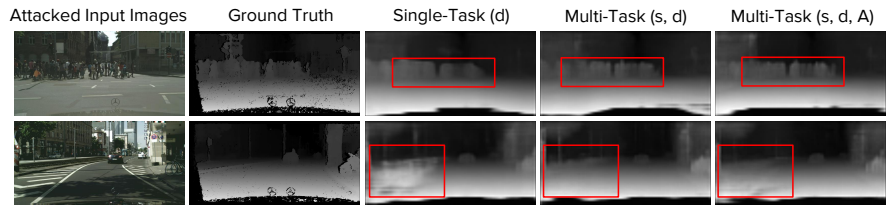


Fig. 3: We show depth predictions of multitask models under multitask attacks. To emphasize the differences, we annotated the figure with red boxes where the errors are particularly noticeable. The multitask trained model outperforms the single-task trained model under attack.

Taskonomy. The Taskonomy dataset [61] consists of images of indoor scenes. We train on up to 11 tasks: semantic segmentation (s), depth euclidean estimation (D), depth zbuffer estimation (d), normal (n), edge texture (e), edge occlusion (E), keypoints 2D (k), keypoints 3D (K), principal curvature (p), re-shading (r), and image reconstruction (A). We use the “tiny” version of their dataset splits [1]. We resize the images to 256×256 .

5.2 Attack Methods

We evaluate the model robustness with L_∞ bounded adversarial attacks, which is a standard evaluation metric for adversarial robustness [36]. We evaluate with four different attacks:

FGSM: We evaluate on the Fast Gradient Sign Method (FGSM) [18], which generates adversarial examples \mathbf{x}_{adv} by $\mathbf{x}_{adv} = \mathbf{x} + \epsilon \cdot \text{sign}(\nabla_{\mathbf{x}} \ell(F(\mathbf{x}), y))$. It is a single step, non-iterative attack.

PGD: Following the attack setup for segmentation in [3], we use the widely used attack PGD (Iteratively FGSM with random start [36]), set the number of iterations of attacks to $\min(\epsilon + 4, \lceil 1.25\epsilon \rceil)$ and step-size $\alpha = 1$. We choose

¹ We use the same dimension for baselines and ours during comparison because input dimension impacts robustness [48].

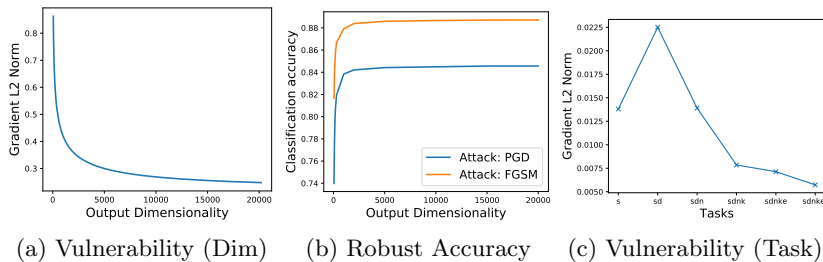


Fig. 4: The effect of output dimensionality and number of tasks on adversarial robustness. We analyzed the pre-trained DRN model on Cityscapes (a,b), and a multitask model trained on Taskonomy (c). The x-axis of (a,b) represents the output dimensionality, the x-axis of (c) shows the combination of multiple tasks. The y-axis of (a,c) is the L2 norm of the joint gradient and is proportional to the model’s adversarial vulnerability. The y-axis of (b) is classification accuracy. The robust performances for (c) are shown in Fig 5. Increasing the output dimensionality or number of tasks improves the model’s robustness.

the L_∞ bound ϵ from $\{1, 2, 4, 8, 16\}$ where noise is almost imperceptible. Under $\epsilon = 4$, we also evaluate the robustness using PGD attacks with $\{10, 20, 50, 100\}$ steps, which is a stronger attack compared to 5 steps attack used in [3].

MIM: We also evaluate on MIM attack [12], which adds momentum to iterative attacks to escape local minima and won the NeurIPS 2017 Adversarial Attack Competition.

Houdini: We evaluate the semantic segmentation task with the state-of-the-art Houdini attack [7], which directly attacks the evaluation metric, such as the non-differentiable mIoU criterion (mean Intersection over Union).

We do not use the DAG [55] attack for segmentation because it is an unrestricted attack without controlling L_∞ bound. For all the iterative attacks, the step size is 1.

5.3 Multitask Models Against Multitask Attack

High Output Dimensionality as Multitask. Our experiment first studies the effect of a higher number of output dimensions on adversarial robustness. As an example, we use semantic segmentation. The experiment uses a pre-trained Dilated Residual Network (DRN-105) [58, 59] model on the Cityscapes dataset. To obtain the given output dimensionality, we randomly select a subset of pixels from the model output. We mitigate the randomness of the sampling by averaging the results over 20 random samples. Random sampling is a general dimension reduction method, which preserves the correlation and structure for high dimensional, structured data [23]. Figure 4a shows that the model’s vulnerability (as measured by the norm of the gradients) decreases as the number of output dimension increases, which validates Theorem 2.

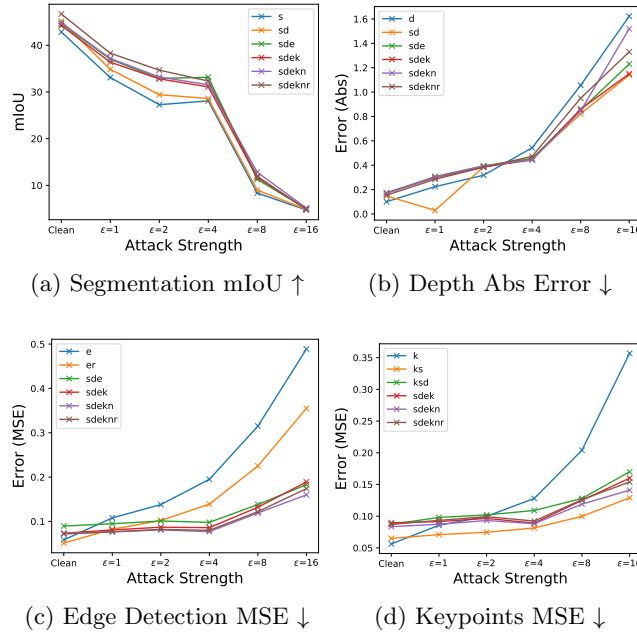


Fig. 5: Adversarial robustness against multitask attack on Taskonomy dataset. The x-axis is the attack strength, ranging from no attack (clean) to the strongest attack ($\epsilon = 16$ PGD). For each subfigure, the y-axis shows the performance of one task under multitask attack. \uparrow means the higher, the better. \downarrow means the lower, the better. The multitask model names are in the legend, we refer to the task by their initials, e.g., ‘sde’ means the model is trained on segmentation, depth, and edge simultaneously. The blue line is the single-task baseline performance, the other lines are multitask performance. The figures show that it is hard to attack all the tasks in a multitask model simultaneously. Thus multitask models are more robust against multitask attacks.

Besides the norm of gradient, we measure the performance under FGSM [18] and PGD [36] adversarial attacks, and show that it improves as output dimensionality increases (Figure 4b). Notice when few pixels are selected, the robustness gains are faster. This is because with fewer pixels: (1) the marginal gain of the inverse function is larger; and (2) the select pixels are sparse and tend to be far away and uncorrelated to each other. The correlation between the output pixels compounds as more nearby pixels are selected, which slows down the improvements to robustness. The results demonstrate that models with higher output dimension/diversity are inherently more robust against adversarial attacks, consistent with the observation in [3, 7] and our Theorem 2.

Number of Tasks. We now consider the case where the number of tasks increases, which is a second factor that increases output dimensionality. We evaluate the robustness of multitask models on the Cityscapes and Taskonomy datasets. We equally train all the tasks with the shared backbone architecture mentioned in Section 3. On Cityscapes, we use DRN-105 model as the architecture for encoder and decoder; on Taskonomy, we use Resnet-18 [61]. Each task

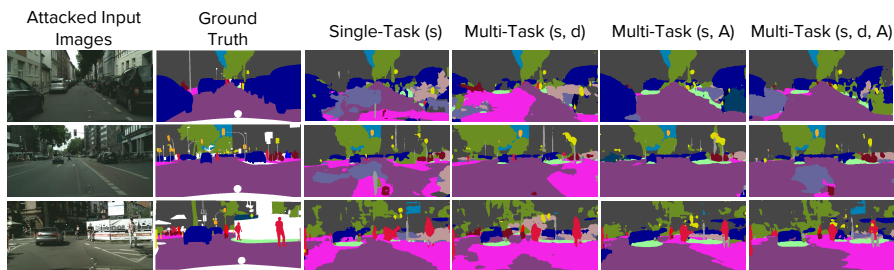


Fig. 6: Performance of single-task attack for multitask models trained on Cityscapes. We show segmentation under attack for single-task and three multitask models. The multitask trained model out-performs the single-task trained model.

has its own decoder. For the Cityscapes dataset, we start with training only the semantic segmentation task, then add the depth estimation and input reconstruction task. For the Taskonomy dataset, following the setup in [49], we start with only semantic segmentation, and add depth estimation, normal, keypoints 2D, edge texture, and reshading tasks to the model one by one. In our figures and tables, we refer to these tasks by the task’s first letter.

Figure 4c shows the L2 norm of the joint gradient for many tasks, which measures the adversarial vulnerability. Overall, as we add more tasks, the norm of the joint gradient decreases, indicating improvement to robustness [48]. The only exception is the depth estimation task, which we believe is due to the large range of values (0 to $+\infty$) that its outputs take. Empirically, a larger output range leads to a larger loss, which implies a larger gradient value.

We additionally measure the robust performance on different multitask models under multitask attacks. Following the setup in [3], we enumerate the ϵ of the L_∞ attack from 1 to 16. Figure 5 shows the robustness of multitask models using Taskonomy, where the adversarial robustness of multitask models are better than single-task models, even if the clean performance of multitask models may be lower. We also observe some tasks gain more robustness compared to other tasks when they are attacked together, which suggests some tasks are inherently harder to attack. Overall, the attacker cannot simultaneously attack all the tasks successfully, which results in improved overall robustness of multitask models.

Training Tasks	Baseline	Multitask		Training Tasks	Baseline	Multitask	
	s	sd	sdA		d	sd	sdA
Clean SemSeg \uparrow	44.77	46.53	45.82	Clean Depth \downarrow	1.82	1.780	<u>1.96</u>
PGD SemSeg \uparrow	15.75	16.01	16.36	PGD Depth \downarrow	6.81	6.08	5.81

Table 1: The models’ performances under multitask PGD attack and clean images on Cityscapes using DRN-D-105 [59]. The **bold** demonstrate the better performance for each row, underline shows inferior results of multitask learning. The results show that multitask models are overall more robust under multitask attack.

In Table 1, we observe the same improvement on Cityscapes. Qualitative results are shown in Figure 2 and Figure 3. Please see the supplemental material for additional results.

		SemSeg mIoU Score \uparrow			Depth Abs Error \downarrow				
Training Tasks \rightarrow		Baseline	Multitask Learning			Baseline	Multitask Learning		
λ_a		s	sd	sA	sdA	d	ds	dA	dAs
	Clean	48.58	48.61	49.61	<u>48.19</u>	1.799	1.792	<u>1.823</u>	1.798
Attacks	FGSM	26.35	<u>26.28</u>	26.79	26.71	3.16	3.01	3.00	3.24
	PGD10	13.04	13.64	14.76	14.48	6.96	6.15	6.03	6.59
	PGD20	11.41	11.98	12.79	12.73	8.81	7.70	7.64	8.38
	PGD50	10.49	10.95	11.68	11.86	10.23	9.07	9.12	9.81
	PGD100	10.15	10.51	11.22	11.52	10.8	9.69	9.74	10.41
	MIM100	9.90	10.17	10.93	11.24	12.04	10.72	10.97	11.69
	Houdini100	5.04	5.14	6.24	6.21	-	-	-	-

Table 2: Model’s robust performance under $L_\infty = 4$ bounded single-task attacks on Cityscapes. Each column is a DRN-22 model trained on a different combination of tasks, where “s,” “d,” and “A” denote segmentation, depth, and auto-encoder, respectively. \uparrow means the higher, the better. \downarrow means the lower, the better. **Bold** in each row, shows the best performance under the same attack. Multitask learning models out-perform single-task models except for the underlined ones. While nearly maintaining the performance on clean examples, multitask models are consistently more robust under strong adversarial attacks.

5.4 Multitask Models Against Single-Task Attacks

Following the setup for multitask learning in [34, 32, 28], we train the multitask models using a main task and auxiliary tasks, where we use $\lambda = 1$ for the main task and λ_a for the auxiliary tasks. We then evaluate the robustness of the main task under single-task attacks. On Cityscapes, the main and the auxiliary tasks share 16 layers of an encoding backbone network. The decoding network for each individual task has 6 layers. For all the models, we train for 200 epochs. For adversarial robustness evaluation, we use strong attacks including PGD100 and MIM100 for attacking the segmentation accuracy², and use 100 steps Houdini [7] to attack the non-differentiable mIoU of the Segmentation model directly. We do not use Houdini to attack the depth because the L1 loss for depth is differentiable and does not need any surrogate loss. The results in Table 2 show that multitask learning improves the segmentation mIoU by 1.2 points and the performance of depth estimation by 11% under attack, while maintaining the performance on most of the clean examples. Qualitative results are in Figure 6.

On the Taskonomy dataset, we conduct experiments on 11 tasks. Following the setup in [49], we use ResNet-18 [20] as the shared encoding network, where

² Suffixed number indicates number of steps for attack

Training Task 2

		None (Baseline)	SemSeg	DepthZ	Edge2D	Normal	Reshading	Key2D	Key3D	DepthE	AutoE	Edge3D	PCurve	Mean
Training Task 1 + Testing Task	SemSeg *	13.36	0.00	44.61	4.42	24.48	9.13	17.51	3.14	11.60	5.61	10.18	11.53	14.22
	DepthZ (10^{-2})	11.49	59.00	0.00	40.99	-1.10	-8.02	3.22	27.27	8.64	56.65	-6.43	56.18	23.64
	Edge2D (10^{-2})	10.67	7.79	12.27	0.00	10.55	6.83	8.81	9.54	8.98	6.85	6.50	5.41	8.35
	Normal (10^{-2})	40.93	14.06	-4.75	3.89	0.00	1.04	3.58	2.43	-2.80	12.71	9.42	-0.70	3.89
	Reshading (10^{-2})	57.89	15.66	0.18	5.05	2.42	0.00	3.36	7.93	-3.68	-5.37	14.80	0.46	4.08
	Key2D (10^{-2})	11.72	7.40	7.08	8.69	10.19	7.13	0.00	6.56	9.41	6.27	8.23	9.88	8.08
	Key3D (10^{-2})	49.70	37.72	0.18	-2.19	7.73	15.14	11.81	0.00	-3.04	34.47	-7.48	-6.39	8.80
	DepthE (10^{-3})	4.85	27.15	29.95	32.96	11.87	-17.11	24.24	23.07	0.00	23.57	31.19	39.44	22.63
	AutoE (10^{-2})	59.31	2.60	-1.62	1.75	-5.08	-0.17	-0.03	-2.37	1.80	0.00	-1.94	-3.68	-0.87
	Edge3D (10^{-2})	15.90	8.36	3.58	-2.71	3.38	4.45	1.96	-6.32	3.12	20.69	0.00	6.98	4.35
	PCurve (10^{-2})	11.47	22.22	22.38	9.74	19.51	16.14	22.39	9.04	2.81	19.91	9.31	0.00	15.34

(a) Performance Under Attack

Training Task 2

		None (Baseline)	SemSeg	DepthZ	Edge2D	Normal	Reshading	Key2D	Key3D	DepthE	AutoE	Edge3D	PCurve	Mean
Training Task 1 + Testing Task	SemSeg *	43.19	0.00	7.20	6.92	7.32	7.06	5.21	5.63	3.47	3.03	4.93	3.01	5.38
	DepthZ (10^{-2})	2.85	4.15	0.00	-36.05	0.19	12.16	-0.78	-24.91	-17.07	-11.20	-8.26	-64.44	-14.62
	Edge2D (10^{-2})	3.38	-15.91	0.05	0.00	-3.64	-1.51	1.60	-4.08	-1.46	-5.62	-5.49	-2.07	-3.81
	Normal (10^{-2})	7.00	-2.64	-1.38	-0.13	0.00	0.11	0.10	-2.64	0.82	1.91	0.93	-2.06	-0.50
	Reshading (10^{-2})	8.03	0.52	-0.94	1.07	1.57	0.00	-0.18	0.88	1.09	1.71	-0.48	-1.53	0.37
	Key2D (10^{-2})	4.16	0.96	6.22	8.67	6.99	0.21	0.00	5.08	7.20	8.01	7.35	6.68	5.74
	Key3D (10^{-2})	8.77	3.72	0.96	2.93	1.84	5.17	0.77	0.00	3.18	4.62	4.66	2.19	3.00
	DepthE (10^{-3})	6.37	-3.17	6.70	0.35	2.15	8.96	-0.71	-1.53	0.00	6.67	10.32	1.91	3.17
	AutoE (10^{-2})	3.47	-4.21	-6.90	-2.27	-3.38	-2.02	-8.96	-8.40	-2.09	0.00	-1.75	-2.40	-4.24
	Edge3D (10^{-2})	4.65	-1.00	0.87	-1.69	1.86	-1.64	6.12	0.31	0.82	9.45	0.00	20.35	3.54
	PCurve (10^{-2})	10.48	20.19	21.87	20.26	18.47	30.96	26.18	26.26	22.17	25.03	26.19	0.00	23.76

(b) Performance on Clean Examples

Fig. 7: We consider models trained on two tasks. In each matrix, the rows show the first training task and the testing task. The columns show the auxiliary training task. The first column without color shows the absolute value for the baseline model (single-task). The middle colored columns show the relative improvement of multitask models over the single-task model in percentage. The last colored column shows the average relative improvement. We show results for both (a) adversarial and (b) clean performance. Multitask learning improves the performance on clean examples for 70/110 cases, and the performance on adversarial examples for 90/110 cases. While multitask training does not always improve clean performance, we show multitask learning provides more gains for adversarial performance.

each individual task has its own prediction network using the encoded representation. We train single-task models for each of the 11 tasks as baselines. We train a total of 110 multitask models — each main task combined with 10 different auxiliary tasks — for 11 main tasks. We evaluate both the clean performance and adversarial performance. λ_a is either 0.1 or 0.01 based on the tasks. We use PGD attacks bounded with $L_\infty = 4$ with 50 steps, where the step size is 1. The attack performance plateaus for more steps. Figure 7 shows the performance

of the main task on both clean and adversarial examples. While maintaining the performance on clean examples (average improvement of 4.7%), multitask learning improves 90/110 the models’ performance under attacks, by an average of 10.23% relative improvement. Our results show that one major advantage of multitask learning, which to our knowledge is previously unknown, is that it improves the model’s robustness under adversarial attacks.

5.5 Multitask Learning Complements Adversarial Training

		SemSeg mIoU Score \uparrow				Depth Abs Error \downarrow			
Training Tasks \rightarrow		Baseline	Multitask Learning			Baseline	Multitask Learning		
		s	sd	sA	sdA	d	ds	dA	dAs
Clean		41.95	43.27	43.65	43.26	2.24	2.07	2.15	2.15
Attacks	PGD50	19.73	22.08	20.45	21.93	2.85	2.61	2.75	2.67
	PGD100	19.63	21.96	20.31	21.83	2.85	2.61	2.75	2.67
	MIM100	19.54	21.89	20.20	21.74	2.85	2.61	2.75	2.67
	Houdini100	17.05	19.45	17.36	19.16	-	-	-	-

Table 3: Adversarial robustness of adversarial training models under $L_\infty = 4$ bounded attacks on Cityscapes. Each column is a model trained on a different combination of tasks. “s,” “d,” and “A” denote segmentation, depth, and auto-encoder respectively. \uparrow indicates the higher, the better. The \downarrow indicates the lower, the better. **Bold** shows the best performance of the same task for each row. Multitask learning improves both the clean performance and robustness upon single-task learning.

We study whether multitask learning helps adversarial robust training. We use DRN-22 on the Cityscapes dataset, and train both single-task and multitask models for 200 epoch under the same setup. The single-task model follows the standard adversarial training algorithm, where we train the model on the generated single-task (segmentation) adversarial attacks. For the multitask adversarial training, we train it on the generated multitask attack images for both semantic segmentation and the auxiliary task. Details are in the supplementary material. Table 3 shows that multitask learning improves the robust performance of both clean examples and adversarial examples, where segmentation mIoU improves by 2.40 points and depth improves by 8.4%.

6 Conclusion

The widening deployment of machine learning in real-world applications calls for versatile models that solve multiple tasks or produce high-dimensional outputs. Our theoretical analysis explains that versatile models are inherently more robust than models with fewer output dimensions. Our experiments on real-world datasets and common computer vision tasks measure improvements in adversarial robustness under attacks. Our work is the first to connect this vulnerability

with multitask learning and hint towards a new direction of research to understand and mitigate this fragility.

Acknowledgements: This work was in part supported by Amazon Research Award; NSF grant CNS-15-64055; NSF-CCF 1845893; NSF-IIS 1850069; ONR grants N00014-16-1-2263 and N00014-17-1-2788; a JP Morgan Faculty Research Award; a DiDi Faculty Research Award; a Google Cloud grant; an Amazon Web Services grant. The authors thank Vaggelis Atlidakis, Augustine Cha, Dídac Surís, Lovish Chum, Justin Wong, and Shunhua Jiang for valuable comments.

References

1. <https://github.com/StanfordVL/taskonomy/tree/master/data>
2. <https://slideslive.com/38917690/multitask-learning-in-the-wilderness>
3. Arnab, A., Miksik, O., Torr, P.H.: On the robustness of semantic segmentation models to adversarial attacks. CVPR (Jun 2018)
4. Carlini, N., Wagner, D.A.: Towards evaluating the robustness of neural networks. In: 2017 IEEE Symposium on Security and Privacy. pp. 39–57 (2017)
5. Caruana, R.: Multitask learning. Mach. Learn. **28**(1), 41–75 (Jul 1997)
6. Chen, Z., Badrinarayanan, V., Lee, C.Y., Rabinovich, A.: Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks (2017)
7. Cisse, M., Adi, Y., Neverova, N., Keshet, J.: Houdini: Fooling deep structured prediction models (2017)
8. Cissé, M., Bojanowski, P., Grave, E., Dauphin, Y., Usunier, N.: Parseval networks: Improving robustness to adversarial examples. In: ICML. pp. 854–863 (2017)
9. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
10. Costales, R., Mao, C., Norwitz, R., Kim, B., Yang, J.: Live trojan attacks on deep neural networks (2020)
11. Doersch, C., Zisserman, A.: Multi-task self-supervised visual learning. arXiv:1708.07860 (2017)
12. Dong, Y., Liao, F., Pang, T., Su, H., Zhu, J., Hu, X., Li, J.: Boosting adversarial attacks with momentum. In: CVPR. pp. 9185–9193 (2018)
13. Désidéri, J.A.: Multiple-gradient descent algorithm (mgda) for multiobjective optimization. Comptes Rendus Mathématique **350**, 313–318 (03 2012)
14. Engstrom, L., Gilmer, J., Goh, G., Hendrycks, D., Ilyas, A., Madry, A., Nakano, R., Nakkiran, P., Santurkar, S., Tran, B., et al.: A discussion of “adversarial examples are not bugs, they are features”. Distill **4**(8) (Aug 2019)
15. Engstrom, L., Ilyas, A., Athalye, A.: Evaluating and understanding the robustness of adversarial logit pairing (2018)
16. Evgeniou, T., Pontil, M.: Regularized multi-task learning. pp. 109–117 (08 2004)
17. Glaßer, C., Reitwießner, C., Schmitz, H., Witek, M.: Approximability and hardness in multi-objective optimization. In: Programs, Proofs, Processes. pp. 180–189 (2010)
18. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. arXiv:1412.6572 (2014)
19. Gur, S., Wolf, L.: Single image depth estimation trained via depth from defocus cues. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
20. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. arXiv 1512.03385 (2015)
21. Kaiser, L., Gomez, A.N., Shazeer, N., Vaswani, A., Parmar, N., Jones, L., Uszkoreit, J.: One model to learn them all (2017)
22. Kannan, H., Kurakin, A., Goodfellow, I.J.: Adversarial logit pairing (2018)
23. Keshavan, R.H., Montanari, A., Oh, S.: Matrix completion from noisy entries. NIPS (2009)
24. Kocabas, M., Karagoz, S., Akbas, E.: Multiposenet: Fast multi-person pose estimation using pose residual network. CoRR (2018)

25. Kokkinos, I.: Ubernet: Training a 'universal' convolutional neural network for low-,mid-, and high-level vision using diverse datasets and limited memory. arXiv:1609.02132 (2016)
26. Kumar, A., III, H.D.: Learning task grouping and overlap in multi-task learning (2012)
27. Kurakin, A., Goodfellow, I.J., Bengio, S.: Adversarial examples in the physical world. arXiv:1607.02533 (2017)
28. Lee, T., Ndirango, A.: Generalization in multitask deep neural classifiers: a statistical physics approach (2019)
29. Liu, S., Johns, E., Davison, A.J.: End-to-end multi-task learning with attention. arXiv:1803.10704 (2018)
30. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector p. 21–37 (2016)
31. Liu, X., He, P., Chen, W., Gao, J.: Multi-task deep neural networks for natural language understanding (2019)
32. Liu, X., He, P., Chen, W., Gao, J.: Multi-task deep neural networks for natural language understanding. Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (2019)
33. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. arXiv:1411.4038 (2014)
34. Luong, M.T., Le, Q.V., Sutskever, I., Vinyals, O., Kaiser, L.: Multi-task sequence to sequence learning (2015)
35. Ma, J., Zhao, Z., Yi, X., Chen, J., Hong, L., Chi, E.: Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. pp. 1930–1939 (07 2018)
36. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards deep learning models resistant to adversarial attacks. In: ICLR (2018)
37. Mao, C., Zhong, Z., Yang, J., Vondrick, C., Ray, B.: Metric learning for adversarial robustness (2019)
38. Metzen, J.H., Kumar, M.C., Brox, T., Fischer, V.: Universal adversarial perturbations against semantic image segmentation. ICCV (Oct 2017)
39. Misra, I., Shrivastava, A., Gupta, A., Hebert, M.: Cross-stitch networks for multi-task learning (2016)
40. Pang, T., Xu, K., Du, C., Chen, N., Zhu, J.: Improving adversarial robustness via promoting ensemble diversity. arXiv:1901.08846 (2019)
41. Papernot, N., McDaniel, P.D., Jha, S., Fredrikson, M., Celik, Z.B., Swami, A.: The limitations of deep learning in adversarial settings. arXiv:1511.07528 (2015)
42. Ross, A.S., Doshi-Velez, F.: Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. arXiv:1711.09404 (2017)
43. Samangouei, P., Kabkab, M., Chellappa, R.: Defense-gan: Protecting classifiers against adversarial attacks using generative models. arXiv:1805.06605 (2018)
44. Schmidt, L., Santurkar, S., Tsipras, D., Talwar, K., Madry, A.: Adversarially robust generalization requires more data. In: NeurIPS. pp. 5019–5031 (2018)
45. Schutze, O., Lara, A., Coello, C.A.C.: On the influence of the number of objectives on the hardness of a multiobjective optimization problem. IEEE Transactions on Evolutionary Computation **15**(4), 444–455 (2011)
46. Sener, O., Koltun, V.: Multi-task learning as multi-objective optimization (2018)
47. Shen, G., Mao, C., Yang, J., Ray, B.: Advspade: Realistic unrestricted attacks for semantic segmentation (2019)

48. Simon-Gabriel, C.J., Ollivier, Y., Bottou, L., Schölkopf, B., Lopez-Paz, D.: First-order adversarial vulnerability of neural networks and input dimension. In: Proceedings of the 36th International Conference on Machine Learning. vol. 97, pp. 5809–5817 (2019)
49. Standley, T., Zamir, A.R., Chen, D., Guibas, L.J., Malik, J., Savarese, S.: Which tasks should be learned together in multi-task learning? arXiv:1905.07553 (2019)
50. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I.J., Fergus, R.: Intriguing properties of neural networks. arXiv:1312.6199 (2013)
51. Tramèr, F., Kurakin, A., Papernot, N., Boneh, D., McDaniel, P.D.: Ensemble adversarial training: Attacks and defenses. arXiv:1705.07204 (2017)
52. Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., Madry, A.: Robustness may be at odds with accuracy. In: International Conference on Learning Representations (2019)
53. Uesato, J., Alayrac, J., Huang, P., Stanforth, R., Fawzi, A., Kohli, P.: Are labels required for improving adversarial robustness? CoRR (2019)
54. Xiao, T., Liu, Y., Zhou, B., Jiang, Y., Sun, J.: Unified perceptual parsing for scene understanding. CoRR (2018)
55. Xie, C., Wang, J., Zhang, Z., Zhou, Y., Xie, L., Yuille, A.: Adversarial examples for semantic segmentation and object detection. In: ICCV (Oct 2017)
56. Xie, C., Wu, Y., van der Maaten, L., Yuille, A.L., He, K.: Feature denoising for improving adversarial robustness. CoRR (2018)
57. Yan, Z., Guo, Y., Zhang, C.: Deep defense: Training dnns with improved adversarial robustness. In: Proceedings of the 32Nd International Conference on Neural Information Processing Systems. pp. 417–426. NIPS’18 (2018)
58. Yu, F., Koltun, V.: Multi-scale context aggregation by dilated convolutions. In: International Conference on Learning Representations (ICLR) (2016)
59. Yu, F., Koltun, V., Funkhouser, T.: Dilated residual networks. In: Computer Vision and Pattern Recognition (CVPR) (2017)
60. Yu, T., Kumar, S., Gupta, A., Levine, S., Hausman, K., Finn, C.: Gradient surgery for multi-task learning (2020)
61. Zamir, A.R., Sax, A., Shen, W.B., Guibas, L., Malik, J., Savarese, S.: Taskonomy: Disentangling task transfer learning. In: 2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE (2018)
62. Zhang, H., Yu, Y., Jiao, J., Xing, E.P., Ghaoui, L.E., Jordan, M.I.: Theoretically principled trade-off between robustness and accuracy. arXiv abs/1901.08573 (2019)

1.A Proof for Theoretical Analysis

We present theoretical analysis to quantify how much multi-task learning improves model’s overall adversarial robustness.

Definition 2. *Given classifier F , input \mathbf{x} , output target \mathbf{y} , and loss $\mathcal{L}(\mathbf{x}, \mathbf{y}) = \ell(F(\mathbf{x}), \mathbf{y})$, the feasible adversarial examples lie in a p -norm bounded ball with radius r , $B(\mathbf{x}, r) := \{\mathbf{x}_{adv}, \|\mathbf{x}_{adv} - \mathbf{x}\|_p < r\}$. Then adversarial vulnerability of a classifier over the whole dataset is*

$$\mathbb{E}_{\mathbf{x}}[\Delta\mathcal{L}(\mathbf{x}, \mathbf{y}, r)] = \mathbb{E}_{\mathbf{x}}\left[\max_{\|\delta\|_p < r} |\mathcal{L}(\mathbf{x}, \mathbf{y}) - \mathcal{L}(\mathbf{x} + \delta, \mathbf{y})|\right].$$

$\Delta\mathcal{L}$ captures the change of output loss given a change in input. Intuitively, a robust model should have a smaller change in loss given a perturbation of the input. Given the adversarial noise is imperceptible, i.e., $r \rightarrow 0$, we can approximate $\Delta\mathcal{L}$ with a first-order Taylor expansion, where

$$|\mathcal{L}(\mathbf{x}, \mathbf{y}) - \mathcal{L}(\mathbf{x} + \delta, \mathbf{y})| = |\partial_{\mathbf{x}}\mathcal{L}(\mathbf{x}, \mathbf{y})\delta + O(\delta)|$$

Lemma 2. *For a given neural network F that predicts multiple tasks, the adversarial vulnerability is*

$$\mathbb{E}_{\mathbf{x}}[\Delta\mathcal{L}(\mathbf{x}, \mathbf{y}, r)] \approx \mathbb{E}_{\mathbf{x}}\left[\|\partial_{\mathbf{x}}\mathcal{L}_{all}(\mathbf{x}, \bar{\mathbf{y}})\|_q\right] \cdot \|\delta\|_p \propto \mathbb{E}_{\mathbf{x}}\left[\|\partial_{\mathbf{x}}\mathcal{L}_{all}(\mathbf{x}, \bar{\mathbf{y}})\|_q\right]$$

Proof. According to the definition of dual norm:

$$\Delta\mathcal{L} \approx \max_{\|\delta\|_p < r} |\partial_{\mathbf{x}}\mathcal{L}(\mathbf{x}, \mathbf{y})\delta| = \|\partial_{\mathbf{x}}\mathcal{L}_{all}(\mathbf{x}, \bar{\mathbf{y}})\|_q \cdot \|\delta\|_p$$

$$\mathbb{E}_{\mathbf{x}}[\Delta\mathcal{L}] \approx \mathbb{E}_{\mathbf{x}}\left[\|\partial_{\mathbf{x}}\mathcal{L}_{all}(\mathbf{x}, \bar{\mathbf{y}})\|_q\right] \cdot \|\delta\|_p$$

where q is the dual norm of p , which satisfies $\frac{1}{p} + \frac{1}{q} = 1$ and $1 \leq p \leq \infty$.

Once given the p -norm bounded ball, i.e., $\|\delta\|_p$ is constant, we get

$$\mathbb{E}_{\mathbf{x}}[\Delta\mathcal{L}] \propto \mathbb{E}_{\mathbf{x}}\left[\|\partial_{\mathbf{x}}\mathcal{L}_{all}(\mathbf{x}, \bar{\mathbf{y}})\|_q\right]$$

□

Theorem 2. (Adversarial Vulnerability of Model for Multiple Correlated Tasks) *If the selected output tasks are correlated with each other such that the covariance between the gradient of task i and task j is $\text{Cov}(\mathbf{r}_i, \mathbf{r}_j)$, and the gradient for each task is i.i.d. with zero mean (because the model is converged), then adversarial vulnerability of the given model is proportional to*

$$\frac{\sqrt{\left(1 + \frac{2}{M} \sum_{i=1}^M \sum_{j=1}^{i-1} \frac{\text{Cov}(\mathbf{r}_i, \mathbf{r}_j)}{\text{Cov}(\mathbf{r}_i, \mathbf{r}_i)}\right)}}{\sqrt{M}}$$

where M is the number of output tasks selected.

Proof. Denote the gradient for task c as \mathbf{r}_c , i.e.,

$$\mathbf{r}_c = \partial_{\mathbf{x}} \mathcal{L}_c(\mathbf{x}, \mathbf{y}_c)$$

We define the joint gradient vector \mathbf{R} as follows:

$$\mathbf{R} = \partial_{\mathbf{x}} \mathcal{L}_{all}(\mathbf{x}, \bar{\mathbf{y}}) = \partial_{\mathbf{x}} \left(\frac{1}{M} \sum_{c=1}^M \mathcal{L}_c(\mathbf{x}, \mathbf{y}_c) \right) = \frac{1}{M} \sum_{c=1}^M \partial_{\mathbf{x}} \mathcal{L}_c(\mathbf{x}, \mathbf{y}_c) = \sum_{c=1}^M \mathbf{r}_c$$

As we can see, the joint gradient is the sum of gradients from each individual task. Then we consider the expectation of the square of the L_2 norm of the joint gradient:

$$\begin{aligned} \mathbb{E}(\|\mathbf{R}\|_2^2) &= \mathbb{E} \left(\left\| \frac{1}{M} \sum_{c=1}^M \mathbf{r}_c \right\|_2^2 \right) = \frac{1}{M^2} \mathbb{E} \left(\sum_{c=1}^M \|\mathbf{r}_c\|^2 + 2 \sum_{i=1}^M \sum_{j=1}^{i-1} \mathbf{r}_i \mathbf{r}_j \right) \\ \mathbb{E}(\|\mathbf{R}\|_2^2) &= \frac{1}{M^2} \left(\sum_{i=1}^M \mathbb{E} \|\mathbf{r}_i\|^2 + 2 \sum_{i=1}^M \sum_{j=1}^i \mathbb{E}(\mathbf{r}_i \mathbf{r}_j) \right) \end{aligned}$$

Since

$$\text{Cov}(\mathbf{r}_i, \mathbf{r}_j) = \mathbb{E}(\mathbf{r}_i \mathbf{r}_j) - \mathbb{E}(\mathbf{r}_i) \mathbb{E}(\mathbf{r}_j)$$

According to the assumption

$$\mathbb{E}(\mathbf{r}_j) = \mathbf{0}$$

We know

$$\text{Cov}(\mathbf{r}_i, \mathbf{r}_j) = \mathbb{E}(\mathbf{r}_i \mathbf{r}_j)$$

Then we get

$$\mathbb{E}(\|\mathbf{R}\|_2^2) = \frac{1}{M^2} \left(\sum_{i=1}^M \mathbb{E}(\text{Cov}(\mathbf{r}_i, \mathbf{r}_i)) \right) + 2 \sum_{i=1}^M \sum_{j=1}^i \mathbb{E}(\text{Cov}(\mathbf{r}_i, \mathbf{r}_j)) = \frac{1}{M^2} \left(\sum_{i=1}^M \sigma^2 + 2 \sum_{i=1}^M \sum_{j=1}^i \mathbb{E}[\text{Cov}(\mathbf{r}_i, \mathbf{r}_j)] \right)$$

where $\sigma^2 = \text{Cov}(\mathbf{r}_i, \mathbf{r}_i)$

Thus, the adversarial vulnerability is:

$$\mathbb{E}_{\mathbf{x}}[\Delta \mathcal{L}] \propto \mathbb{E}_{\mathbf{x}} [\|\partial_{\mathbf{x}} \mathcal{L}_{all}(\mathbf{x}, \bar{\mathbf{y}})\|_2] = \frac{\sqrt{(1 + \frac{2}{M} \sum_{i=1}^M \sum_{j=1}^{i-1} \frac{\text{Cov}(\mathbf{r}_i, \mathbf{r}_j)}{\text{Cov}(\mathbf{r}_i, \mathbf{r}_i)})}}{\sqrt{M}}$$

□

For a special case where all the tasks are independent of each other, independent gradients with respect to the input are produced, we have the following corollary:

Corollary 2. (Adversarial Vulnerability of Model for Multiple Independent Tasks) *If the output tasks selected are independent of each other, and the gradient for each task is i.i.d. with zero mean, then the adversarial vulnerability of given model is proportional to $\frac{1}{\sqrt{M}}$, where M is the number of independent output tasks selected.*

Proof. According to the independent assumption, we have

$$\text{Cov}(\mathbf{r}_i, \mathbf{r}_j) = \mathbf{0}$$

Let $\sigma^2 = \text{Cov}(\mathbf{r}_i, \mathbf{r}_i)$. Thus we get the adversarial vulnerability to be:

$$\mathbb{E}_{\mathbf{x}}[\Delta\mathcal{L}] \propto \mathbb{E}_{\mathbf{x}} [\|\partial_{\mathbf{x}}\mathcal{L}_{all}(\mathbf{x}, \bar{\mathbf{y}})\|_2] = \sqrt{\frac{\sigma^2}{M}} \propto \frac{1}{\sqrt{M}}$$

□

1.B Experimental Setup

1.B.1 Cityscapes

We train DRN-105 model and evaluate against multi-task attack. We follow the original architecture setup of the original DRN paper [59]. We used 93 layers in the shared backbone encoder network, and 13 layers in the decoder branch for individual task prediction. We use a batch size of 24. We start with a learning rate of 0.01 and decrease the learning rate by a factor of 10 after every 100 epochs. We trained the model for 250 epochs.

We train multi-task model against single task attack using DRN-22 model. We use 18 layers in the shared backbone encoder network, and 9 layers in the decoder branch for individual task prediction. We use batch size of 32. We optimize with SGD, with learning rate of 0.01, then decrease it to 0.001 at 180 epoch. We train model for 200 epoch in total. We applied a weight decay of 0.0001 for all the models.

1.B.2 Taskonomy

Taskonomy dataset [61] consists of millions of indoor scenes with labels for multiple tasks, we use 11 tasks including semantic segmentation, depth estimation, 2D and 3D edge detection, normal vector estimation, reshading, 2D and 3D keypoint detection, Euclidean depth, auto-encoding, and principal curvature estimation. We use the publicly available Tiny version of dataset, which consists of 9464 images from 1500 rooms. We use examples from 80% of the rooms as training data and examples from 20% of the rooms as test data. Images from the same room are only contained in either the training set or the test set, and not in both. The quality of the model is measured by its ability to generalize to new rooms.

For learning a multi-task model for joint robustness, we follow the set up described in [49]. We train a ResNet-18 as the shared backbone encoder network for all the tasks. Each multi-task model consists of 1 to 6 different tasks. We use an input size of 512×512 . We use an 8 layer decoder for each individual task prediction. Following the data preprocessing of [49], we apply equal weights to all the tasks. Start from task "semantic segmentation" (s), we add tasks "depth" (d), "edge texture" (e), "keypoints 2d" (k), "normal" (n), and "reshading" (r). Thus we train 6 models 's,' 'sd,' 'sde,' 'sdek,' 'sdekn,' 'sdeknr.' We also train 'd,' 'e,' 'er,' 'k,' 'ks,' 'ksd' tasks, so that we can analysis the trend of 4 tasks' performance after multitask learning. We use the same learning rate schedule for all the models — SGD with learning rate 0.01 and momentum 0.99. We decrease the learning rate at 100 epoch by 10 times. We train all the models for 150 epoch. Results are shown in Figure 5 in the main paper.

For training robust models on select tasks, we use ResNet-18 as the shared encoder network. We select 11 tasks trained in pairs with each other, which results in 110 models. We study their robustness under a single-task attack. We follow the data processing in [61]. For each task we considered, we try weights of 0.1 and 0.01 for the auxiliary task, and choose the weight that produces higher robust accuracy. The chosen λ_a for the auxiliary tasks are shown in Table 5. The selection of weights is important due to the complex interactions of different tasks [2]. We follow the setup in [61], and subsample the image from 512 to 256 using linear interpolation. For segmentation, reshading, keypoint 3D, depth Euclidean, Auto Encoder, principle curvature, we use SGD, with learning rate 0.01 and decrease by 10 times at 140 epoch. For the other tasks we use adam, with learning rate 0.001 and decrease by 10 times at 120 and 140 epoch. Due to the inherent difference between different tasks, we use different optimizer for different tasks for better convergence. All the models are trained for 150 epoch. All the results are shown in Table 4. As we can see, learning versatile, multi-task models improves adversarial robustness on 90/110 tasks.

1.B.3 Adversarial Training

We present the details for multi-task adversarial training in Algorithm 1. For single task model, we choose $S = \{\{T_m\}\}$. The algorithm is the same as the adversarial training procedure of Madry et. al. [36]. For multi-task model, we set $S = \{\{T_m\}, \{T_m, T_a^{(1)}, \dots\}\}$, thus the generated adversarial images under multi-task are more diversified compared with single-task models. In addition, all the adversarial examples are trained on multi-task loss function, where the auxiliary task can introduce useful knowledge for learning the robust main task. We use $\lambda = 0.01$ for the auxiliary task. For all the task, we train using SGD optimizer with batch size of 32, for 200 epoch. We start with learning rate of 0.01, and decrease the learning rate by 10 times at 180 epoch. The experiment are conducted on Cityscapes dataset.

	Baseline	PGD Adversarial										
		SemSeg	DepthZ	Edge2D	Normal	Reshad	Key2D	Key3D	DepthE	AutoE	Edge3D	PCurve
Semseg *	13.360	—	19.320	13.950	16.630	14.580	15.700	13.780	14.910	14.110	14.720	14.900
DepthZ (10^{-2})	11.491	4.712	—	6.780	11.617	12.412	11.120	8.358	10.498	4.981	12.230	5.035
Edge2D (10^{-2})	10.672	9.841	9.363	—	9.546	9.943	9.732	9.654	9.714	9.941	9.978	10.095
Normal (10^{-2})	40.926	35.171	42.871	39.335	—	40.501	39.462	39.930	42.071	35.726	37.070	41.212
Reshad (10^{-2})	57.900	48.800	57.800	55.000	56.500	—	55.900	53.300	60.000	61.000	49.300	57.600
Key2D (10^{-2})	11.700	10.900	10.900	10.700	10.500	10.900	—	11.000	10.600	11.000	10.800	10.600
Key3D (10^{-2})	49.700	31.000	49.600	50.800	45.900	42.200	43.800	—	51.200	32.600	53.400	52.900
DepthE (10^{-3})	4.850	3.530	3.390	3.250	4.270	5.670	3.670	3.730	—	3.700	3.330	2.930
AutoE (10^{-2})	59.300	57.800	60.300	58.300	62.300	59.400	59.300	60.700	58.200	—	60.500	61.500
Edge3D (10^{-2})	15.900	14.600	15.300	16.300	15.400	15.200	15.600	16.900	15.400	12.600	—	14.800
PCurve (10^{-4})	11.500	8.920	8.900	10.400	9.230	9.620	8.900	10.400	11.100	9.190	10.400	—
Clean												
SemSeg *	43.190	—	46.300	46.180	46.350	46.240	45.440	45.620	44.690	44.500	45.320	44.490
DepthZ (10^{-2})	2.852	2.734	—	3.880	2.846	2.505	2.874	3.562	3.339	3.171	3.088	4.690
Edge2D (10^{-2})	3.384	3.922	3.382	—	3.507	3.435	3.330	3.522	3.433	3.574	3.569	3.454
Normal (10^{-2})	6.997	7.181	7.093	7.006	—	6.989	6.990	7.182	6.940	6.864	6.931	7.141
Reshad (10^{-2})	8.027	7.985	8.103	7.941	7.901	—	8.041	7.957	7.940	7.890	8.065	8.150
Key2D (10^{-2})	4.156	4.116	3.897	3.795	3.865	4.147	—	3.944	3.857	3.823	3.850	3.878
Key3D (10^{-2})	8.771	8.445	8.686	8.514	8.610	8.318	8.703	—	8.492	8.366	8.362	8.578
DepthE (10^{-3})	6.373	6.575	5.946	6.350	6.236	5.802	6.418	6.470	—	5.948	5.715	6.251
AutoE (10^{-2})	3.470	3.616	3.709	3.548	3.587	3.540	3.780	3.761	3.542	—	3.530	3.553
Edge3D (10^{-2})	4.649	4.695	4.608	4.727	4.562	4.725	4.364	4.635	4.611	4.210	—	3.703
PCurve (10^{-4})	8.017	8.360	8.184	8.353	8.541	7.232	7.733	7.725	8.153	7.854	7.732	—

Table 4: The absolute performance of all models trained on two tasks (Relative are shown in Figure 7 in the main paper). Each row in the first column lists the name of the main task. The second column (baseline) shows the performance of a model trained on a single task. The * in the row indicates the mIoU score for semantic segmentation, for which higher is better. The values in the other rows of the table show the l1 loss, for which lower is better. The (10^{-n}) in the first column indicates the unit for the error. ‘SemSeg’ denotes ‘semantic segmentation,’ ‘DepthZ’ denotes ‘depth estimation,’ ‘Edge2D’ denotes ‘2D edge detection,’ ‘Normal’ denotes ‘Normal Vector estimation,’ ‘Reshad’ denotes ‘Reshading,’ ‘Key2D’ denotes ‘2D Keypoint detection,’ ‘Key3D’ denotes ‘3D Keypoint detection,’ ‘DepthE’ denotes ‘Euclidean depth,’ ‘AutoE’ denotes ‘Auto Encoder,’ ‘Edge3D’ denotes ‘3D Edge detection,’ ‘PCurve’ denotes ‘Curvature estimation.’ Values superior to the baseline are **bold**, and the best performance for each row is in a **box**. The table lists the IoU (large is better) for the segmentation model, and error (small is better) for all the other tasks. We pair each selected model with 11 other models. All the models converge after training for 150 epochs. Overall, training on two tasks can help the individual task’s adversarial robustness on **90/110** cases, while surpassing the baseline’s performance on the clean examples on **70/110**. For instance, the adversarial robustness for the semantic segmentation and keypoints3D estimation is always improved by multi-task learning while the clean accuracy also improves. The results on 11 tasks support our claim that training on multiple tasks improves adversarial robustness.

	λ_a										
	SemSeg	DepthZ	Edge2D	Normal	Reshad	Key2D	Key3D	DepthE	AutoE	Edge3D	PCurve
Semseg *	0	0.01	9.01	0.1	0.1	0.01	0.01	0.1	0.01	0.1	0.01
DepthZ	0.1	0	0.1	0.01	0.1	0.1	0.01	0.1	0.1	0.1	0.01
Edge2D	0.1	0.1	0	0.1	0.1	0.01	0.1	0.1	0.01	0.01	0.1
Normal	0.1	0.01	0.1	0	0.01	0.1	0.01	0.1	0.1	0.01	0.01
Reshad	0.01	0.1	0.01	0.01	0	0.1	0.01	0.01	0.1	0.01	0.1
Key2D	0.1	0.1	0.01	0.01	0.01	0	0.01	0.01	0.01	0.01	0.1
Key3D	0.1	0.01	0.1	0.1	0.1	0.1	0	0.1	0.1	0.01	0.01
DepthE	0.1	0.01	0.01	0.01	0.01	0.1	0.01	0	0.01	0.1	0.1
AutoE	0.1	0.01	0.01	0.1	0.01	0.1	0.01	0.1	0	0.01	0.1
Edge3D	0.1	0.01	0.01	0.01	0.01	0.1	0.01	0.01	0.1	0	0.1
PCurve	0.1	0.01	0.1	0.01	0.01	0.1	0.01	0.1	0.01	0.01	0

Table 5: The λ_a value for the auxiliary task for Figure 7 in the main paper.**Algorithm 1** Adversarial Training with Multi-task Learning

Input: Initialized networks F_i , dataset D , main task T_m , auxiliary task $T_a^{(i)}$. Construct multi-task combination set $S = \{\{T_m\}, \{T_m, T_a^{(1)}, \dots, \}\}$

Output:

for number of training epochs **do**

for number of iterations in each epoch **do**

 Sample minibatch of n images \mathbf{x} from D .

for each task combination S_t in S **do**

 Let $\mathcal{L}_t(\mathbf{x}, \mathbf{y}) = \sum_i \lambda_i \ell(F_i(\mathbf{x}), \mathbf{y}_i)$, where $i = 1, \dots, \#(S_t)$, $T_i \in S_t$.

 Compute adversarial attack images \mathbf{x}_{adv}

$$\operatorname{argmax}_{\mathbf{x}_{adv}} \mathcal{L}_t(\mathbf{x}_{adv}, \mathbf{y}), \text{ s.t. } \|\mathbf{x}_{adv} - \mathbf{x}\|_p \leq r$$

 Training the multi-task model using the generated attack image \mathbf{x}_{adv} by optimizing the following loss function:

$$\min \mathcal{L}_t(\mathbf{x}, \mathbf{y})$$

end for

end for

end for

return Neural network model F_i