ML-LOO: Detecting Adversarial Examples with Feature Attribution

Puyudi Yang,[1] **Jianbo Chen,**[2] **Cho-Jui Hsieh,**[3] **Jane-Ling Wang,**[1] **Michael I. Jordan**[2]
[1]University of California, Davis
[2]University of California, Berkeley
[3]University of California, Los Angeles

Abstract

Deep neural networks obtain state-of-the-art performance on a series of tasks. However, they are easily fooled by adding a small adversarial perturbation to the input. The perturbation is often imperceptible to humans on image data. We observe a significant difference in feature attributions between adversarially crafted examples and original examples. Based on this observation, we introduce a new framework to detect adversarial examples through thresholding a scale estimate of feature attribution scores. Furthermore, we extend our method to include multi-layer feature attributions in order to tackle attacks that have mixed confidence levels. As demonstrated in extensive experiments, our method achieves superior performances in distinguishing adversarial examples from popular attack methods on a variety of real data sets compared to state-of-the-art detection methods. In particular, our method is able to detect adversarial examples of mixed confidence levels, and transfer between different attacking methods. We also show that our method achieves competitive performance even when the attacker has complete access to the detector.

Introduction

Deep neural networks have achieved state-of-the-art performance on a variety of tasks, including image classification, object detection, speech recognition and machine translation. However, they have been shown to be vulnerable to adversarial examples. This incurs a security risk when DNNs are applied to sensitive areas such as finance, medicine, criminal justice and transportation. Adversarial examples are inputs to machine learning models that an attacker constructs intentionally to fool the model (Goodfellow et al. 2017). Szegedy et al. observed that a visually indistinguishable perturbation in pixel space to the original image can alter the prediction of a neural network. Later, a series of papers (Goodfellow, Shlens, and Szegedy 2015; Kurakin, Goodfellow, and Bengio 2017; Moosavi-Dezfooli, Fawzi, and Frossard 2016; Carlini and Wagner 2017b; Madry et al. 2018; Chen et al. 2017; Ilyas et al. 2018; Brendel, Rauber, and Bethge 2018) designed more sophisticated methods for the worst-case perturbation within a restricted set, often a small L_p ball with $p = 0, 2, \infty$.

While a line of work tries to explain why adversarial examples exist (Goodfellow, Shlens, and Szegedy 2015; Tanay and Griffin 2016; Fawzi, Fawzi, and Frossard 2018), a comprehensive analysis of underlying reasons has not yet been attained, largely because deep neural networks have complex functional forms such that mathematical characterizations are difficult to obtain. On the other hand, there has been a growing interest in developing tools for tackling the black-box nature of neural networks, among which feature attribution is a widely studied approach (Shrikumar, Greenside, and Kundaje 2017; Bach et al. 2015; Simonyan, Vedaldi, and Zisserman 2013; Ribeiro, Singh, and Guestrin 2016; Li, Monroe, and Jurafsky 2016; Lundberg and Lee 2017; Datta, Sen, and Zick 2016; Chen et al. 2019). Given a predictive model, such a method outputs, for each instance to which the model is applied, a vector of importance scores associated with the underlying features. Feature attribution has been used to improve transparency and fairness of machine learning models (Ribeiro, Singh, and Guestrin 2016; Datta, Sen, and Zick 2016).

In this paper, we investigate the application of feature attribution to detecting adversarial examples. In particular, we observe that the feature attribution map of an adversarial example near the boundary always differs from that of the corresponding original example. A motivating example is shown in Figure 1, which demonstrates images in CIFAR-10 to be fed into a residual neural network and the corresponding feature attribution from Leave-One-Out (LOO) (Li, Monroe, and Jurafsky 2016). The latter interprets decisions from a neural model by observing the effects on the model of erasing each pixel of input before and after the worst-case perturbation by a C&W attack. While the perturbation on the original image is visually imperceptible, the feature attribution is altered drastically. We further observe that the difference can be summarized by simple statistics that characterize *feature disagreement*, which are capable of distinguishing adversarial examples from natural images. We conjecture that this is because adversarial attacks tend to perturb samples into an unstable region on the decision surface.

The above observation led to an effective method for detecting adversarial examples near the decision boundary. On the other hand, there also exist adversarial examples in which the model has high confidence (Carlini and Wagner 2017b). Previous work has observed that several state-of-the-art de-

tection methods are vulnerable to such attacks (Lu, Chen, and Yu 2018; Athalye, Carlini, and Wagner 2018). However, we observe an interesting phenomenon: middle layers of neural networks still contain information on uncertainty even for high-confidence adversarial examples. Based on this observation, we generalize our method to incorporate multi-layer feature attribution, where attribution scores for intermediate layers are computed without incurring extra model queries.

In numerical experiments, our method achieves superior performance in detecting adversarial examples generated from popular attack methods on MNIST, CIFAR-10 and CIFAR-100 among state-of-the-art detection methods. The proposed method is also capable of detecting mixed-confidence adversarial examples, transferring between adversarial examples of different confidence levels, and adversarial examples generated by various types of attacks. We further show that the proposed method performs competitively under the setting where the attacker has complete access to the detector.

Related Work

In this section, we review related work in feature attribution, adversarial attack, adversarial defense and detection.

Feature attribution A variety of methods have been proposed to assign feature attribution scores. For each specific instance where the model is applied, an attribution method assigns an importance score for each feature, by approximating the target model via a linear model locally around the instance. One popular class of methods assumes the differentiability of the model, and propagates the prediction to features through gradients. Examples include direct use of gradient (Saliency Map) (Simonyan, Vedaldi, and Zisserman 2013), Layer-wise Relevance Propagation (LWRP) (Bach et al. 2015) and its improved version DeepLIFT (Shrikumar, Greenside, and Kundaje 2017), and Integrated Gradients (Sundararajan, Taly, and Yan 2017).

Another class is perturbation-based and thus model-agnostic. Given an instance, multiple perturbed samples are generated by masking different groups of features with a pre-specified reference value. The feature attribution of the instance is computed according to the prediction scores of a model on these samples. Popular perturbation based methods include Leave-One-Out (Zeiler and Fergus 2014; Li, Monroe, and Jurafsky 2016), LIME (Ribeiro, Singh, and Guestrin 2016) and KernelSHAP (Lundberg and Lee 2017).

It has been observed in Ghorbani, Abid, and Zou that gradient-based feature attribution maps are sensitive to small perturbations. Adversarial attack to feature attribution is designed to characterize the fragility. On the contrary, robustness of an attribution method has been observed on a robust model. In fact, Yeh et al. observed that gradient based explanations of an adversarially trained network are less sensitive, and Chalasani et al. established theoretical results for the robustness of attribution map on an adversarially trained logistic regression. These observations indicate that the sensitivity of a feature attribution might be rooted in the sensitivity of the model, instead of the attribution method. This motivates the detection of adversarial examples via attribution methods.

Adversarial attack Adversarial attacks try to alter, with minimal perturbation, the prediction of an original instance from a given model, which leads to adversarial examples. Adversarial examples can be categorized as targeted or untargeted, depending on whether the goal is to classify the perturbed instance into a given target class or an arbitrary class different from the correct one. Attacks also differ by the type of distance they use to characterize minimal perturbation. ℓ_∞, ℓ_0, and ℓ_2 distances are the most commonly used distances. Fast Gradient Sign Method (FGSM) by Goodfellow, Shlens, and Szegedy is an efficient method to minimize the ℓ_∞ distance. Kurakin, Goodfellow, and Bengio and Madry et al. proposed ℓ_∞-PGD (BIM), an iterative version of FGSM, which achieves a higher success rate with a smaller size of perturbation. DeepFool presented by Moosavi-Dezfooli, Fawzi, and Frossard minimizes ℓ_2 distance through an iterative linearization procedure. Carlini and Wagner proposed effective algorithms to generate adversarial examples for each of the three distances. In particular, Carlini and Wagner proposed a loss function that is capable of controlling the confidence level of adversarial examples. The Jacobian-based Saliency Map Attack (JSMA) by (Papernot et al. 2016a) is a greedy method for perturbation with ℓ_0 metric. Recently, several black-box adversarial attacks that solely depend on probability scores or decisions have been introduced. Chen et al. and Ilyas et al.; Ilyas, Engstrom, and Madry introduced score-based methods using zeroth-order gradient estimation to craft adversarial examples. Brendel, Rauber, and Bethge introduced Boundary Attack, as a black-box method to minimize the ℓ_2 distance, that does not need access to gradient information and relies solely on the model decision. We demonstrate in our experiments that our method is capable of detecting adversarial examples generated by these attacks, regardless of the distance, confidence level, or whether the gradient information is used.

Adversarial defense and detection To improve the robustness of neural networks, various approaches have been proposed to defend against adversarial attacks, including adversarial training (Goodfellow, Shlens, and Szegedy 2015; Kurakin, Goodfellow, and Bengio 2017; Madry et al. 2018; Tramèr et al. 2018; Liu and Hsieh 2019), distributional smoothing (Miyato et al. 2016), defensive distillation (Papernot et al. 2016b), generative models (Song et al. 2018), feature squeezing (Xu, Evans, and Qi 2018), randomized models (Liu et al. 2018; Lecuyer et al. 2019; Liu et al. 2019), and verifiable defense (Wong and Kolter 2018; Dvijotham et al. 2018). These defenses often involve modifications in the training process of a model, which often require higher computational or sample complexity (Schmidt et al. 2018), and lead to loss of accuracy (Tsipras et al. 2018).

Complimentary to the previous defending techniques, an alternative line of work focuses on screening out adversarial examples in the test stage without touching the training of the original model. Data transformations such as PCA have

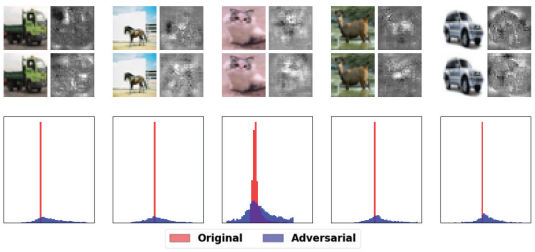

Figure 1: The first row shows the original CIFAR-10 examples and their corresponding feature attributions. The second row shows the adversarial examples and their corresponding feature attributions. The third row plots the histograms of the original and adversarial feature attributions.

been used to extract features from the input and layers of neural networks for adversarial detection (Li and Li 2017; Bhagoji, Sitawarin, and Mittal 2018; Hendrycks and Gimpel 2017). Alternative neural networks are used to classify adversarial and original images (Grosse et al. 2017; Gong, Wang, and Ku 2017; Metzen et al. 2017). Feinman et al. proposed to use kernel density estimate (KD) and Bayesian uncertainty (BU) in hidden layers of the neural network for detection. Ma et al. observed Local Intrinsic Dimension (LID) of hidden-layer outputs differ between the original and adversarial examples. Lee et al. obtained the class conditional Gaussian distributions with respect to lower-level and upper-level features of the deep neural network under Gaussian discriminant analysis, which result in a confidence score based on the Mahalanobis distance (MAHA), followed by a logistic regression model on the confidence scores to detect adversarial examples. Through vast experiments, we show that our method achieves comparable or superior performance than these detection methods across various attacks. Furthermore, we show that our method achieves competitive performance for attacks with a varied confidence level, a setting where the other detection methods fail to work (Lu, Chen, and Yu 2018; Athalye, Carlini, and Wagner 2018).

Most related to our work, Tao et al. proposed to identify neurons critical for individual attributes to detect adversarial examples, but their method is restricted to models in face recognition. Instead, our method is applicable across different types of image data. Zhang et al. proposed to identify adversarial perturbations by training a neural network on the saliency map of inputs. However, their method depends on additional neural networks, which are vulnerable to white-box attacks when attackers perturb the image to fool the original model and the new neural network simultaneously. As we will show in experiments, our method achieves competitive performance under white-box attacks.

Adversarial detection with feature attribution

We motivate our method by an observation on feature attribution with and without adversarial perturbation. Then we discuss metrics to quantify the dispersion in attribution. Finally, we extend our method to the multi-layer version for detecting adversarial examples with mixed confidence levels.

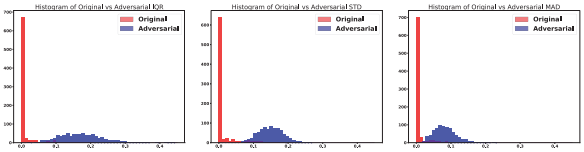

Figure 2: Histogram of dispersion measures

Feature attribution before and after perturbation

Assume that the model is a function $f : \mathbb{R}^d \to [0, 1]^C$ which maps an image x of dimension $d = h \times w \times c$ to a probability vector $f(x)$ of dimension C, where C is the number of classes. A feature attribution method ϕ maps an input image $x \in \mathbb{R}^d$ to an attribution vector of the same shape as the image: $\phi(x) \in \mathbb{R}^d$, such that the i-th dimension of $\phi(x)$ is the contribution of feature i in the prediction of the model on the specific image x. We suppress the dependence of ϕ on the model f for notational convenience. We focus on the leave-One-Out (LOO) method (Zeiler and Fergus 2014; Li, Monroe, and Jurafsky 2016) throughout the paper, which assigns to each feature the reduction in the probability of the selected class when the feature in consideration is masked by some reference value, e.g. 0. Denoting the example with the i-th feature masked by 0 as $x_{(i)}$, LOO defines ϕ as

$$\phi(x)_i := f(x)_c - f(x_{(i)})_c, \text{ where } c = \arg\max_{j \in C} f(x)_j. \quad (1)$$

Adversarial attacks aim to change the prediction of a model with minimal perturbation of a sample, so that human is not able to detect the difference between an original image x and its perturbed version x'. Yet we observed that ϕ is sensitive to the small difference between x and x'. Figure 1 shows the attribution maps $\phi(x), \phi(x')$ with the original image x and its adversarially perturbed counterpart x' by C&W attack. Even with human eyes, we can observe an explicit difference in the attribution maps of the original and adversarial images. In particular, adversarial images have a larger dispersion in its importance scores, as demonstrated in Figure 1. We comment here that our proposed framework of adversarial detection via feature attribution is generic to popular feature attribution methods. As an example, we show the performance of Integrated Gradients (Sundararajan, Taly, and Yan 2017) for adversarial detection in the supplementary material at https://github.com/Jianbo-Lab/ML-LOO. LOO achieves the best performance among all attribution methods across different data sets.

Quantify the dispersion in feature attribution maps

Motivated by the apparent differences in the distributions of importance scores between the original and adversarial images, as demonstrated in Figure 1, we propose to use measures of statistical dispersion in feature attribution to detect adversarial examples. In particular, we tried standard deviation (STD), median absolute deviation (MAD), which is the median of absolute differences between entries and their median, and interquartile range (IQR), which is the difference

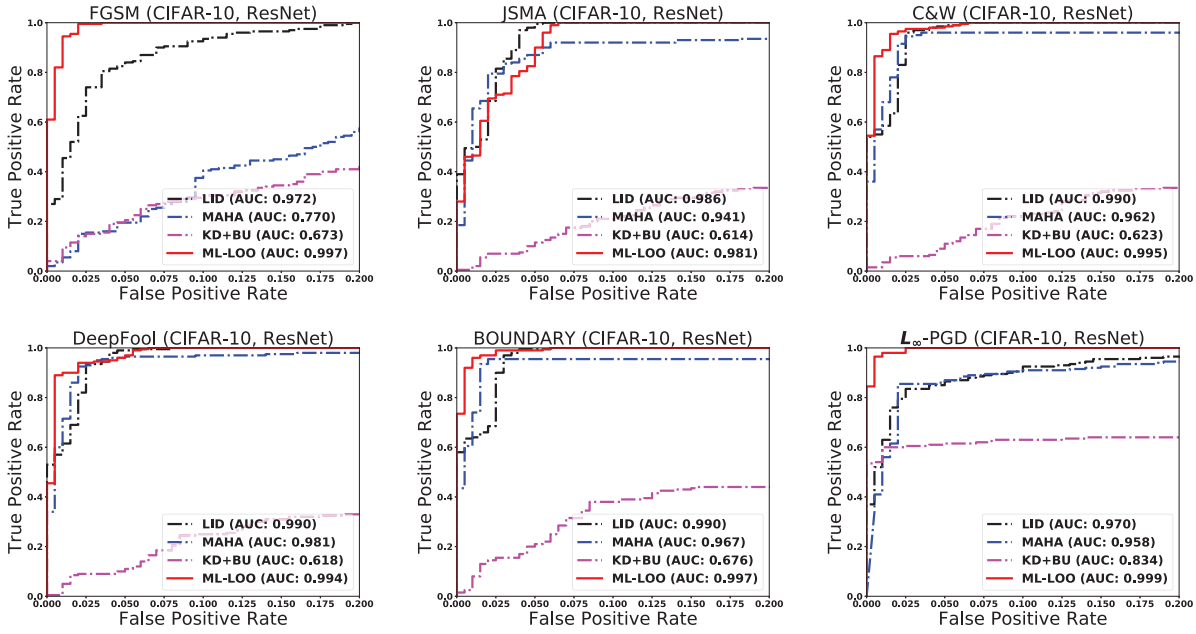

Figure 3: ROC curves of detection methods on CIFAR-10 with ResNet

between the 75th percentile and the 25th percentile among all entries of $\phi(x) \in \mathbb{R}^d$:

$$\text{IQR}(\phi(x)) = Q_{\phi(x)}(0.75) - Q_{\phi(x)}(0.25),$$
$$\text{where } Q_{\phi(x)}(p) := \min\{\beta : \frac{\#\{i : \phi(x)_i < \beta\}}{d} \geq p\}.$$

We observe there is a larger dispersion, which we call *feature disagreement*, between feature contribution to a model for an adversarially perturbed image. The difference is universal across different images. Figure 2 compares the histograms of these three dispersion measures of feature attributions for ResNet on natural test images from CIFAR-10 with those on adversarially perturbed images, where the adversarial perturbation is carried out by C&W Attack. We can see there is a significant difference in the distributions of STD, MAD and IQR between natural and adversarial images. A majority of adversarially perturbed images have a larger dispersion in feature attribution than an arbitrary natural image, besides the corresponding original images. We propose to distinguish adversarial images from natural images by thresholding the IQR of feature attribution maps. In the supplementary material, we show the ROC curves of adversarial detection using the three dispersion measures on CIFAR-10 with ResNet across three different attacks. All the three measures yield competitive performance. We stick to IQR for the rest of the paper, which is robust and has a slightly superior performance among the three.

Extension to multi-layer LOO: detection of attacks with mixed confidence levels

Carlini and Wagner proposed the following objective to generate adversarial images with small ℓ_2 perturbation.

$$\min_w \|x' - x\|_2 + \alpha \max\{F(x)_{y_{\text{true}}} - \max_{j \neq y_{\text{true}}} F(x')_j + c, 0\}, \quad (2)$$

where $x' = 0.5(\tanh(w) + 1)$, F maps an image to logits, $y_{\text{true}} = \arg\max F(x)$ is the original label, and c is a hyper-parameter for tuning confidence. Adversarial images with high confidence can be obtained by assigning a large value to c. The loss can be modified to generate ℓ_∞ constrained perturbation at different confidence levels as well (Madry et al. 2018). Recently, Lu, Chen, and Yu and Athalye, Carlini, and Wagner observed that LID has a poor performance when faced with adversarial examples at various confidence scales. In our experiments, a similar phenomenon is observed for several other state-of-the-art detection methods, including KD+BU and MAHA, as is shown in Figure 4. This suggests that characterization of adversarial examples in related work may only hold true for adversarial examples near the decision boundary. IQR of feature attribution map, unfortunately, suffers from the same problem.

To detect adversarial images with mixed confidence levels, we generalize our method to capture dispersion of feature attributions beyond the output layer of the model. For an adversarial example within a small neighborhood of its original example in the pixel space but achieving a high confidence at the output layer in a different class from the original one, the feature representation deviates away from that of its original example gradually along the layers. Thus, we expect neurons of middle layers contain uncertainty that can be captured by a feature attribution map. We denote the map from input to an arbitrary neuron n of an intermediate layer of the model by $f_n : \mathbb{R}^d \to \mathbb{R}$. The feature attribution of neuron n is defined as $\phi_{f_n}(x) : \mathbb{R}^d \to \mathbb{R}^d$, such that the i-th entry quantifies

Data	Model	Metric	Attacks											
			C&W				ℓ_∞-PGD				FGSM			
			KD+BU	LID	MAHA	ML-LOO	KD+BU	LID	MAHA	ML-LOO	KD+BU	LID	MAHA	ML-LOO
MNIST	CNN	AUC	0.893	1.000	0.957	**1.000**	0.766	0.902	0.736	**1.000**	0.744	0.780	0.967	**1.000**
		TPR (FPR@0.01)	0.23	**0.99**	0.94	0.98	0.09	0.32	0.01	**0.99**	0.01	0.09	0.54	**0.99**
		TPR (FPR@0.05)	0.46	**0.99**	0.94	0.98	0.28	0.58	0.12	**0.99**	0.15	0.23	0.92	**0.99**
		TPR (FPR@0.10)	0.55	**0.99**	0.94	0.98	0.34	0.72	0.29	**0.99**	0.24	0.40	0.94	**0.99**
CIFAR10	ResNet	AUC	0.623	0.990	0.962	**0.995**	0.834	0.970	0.958	**0.999**	0.673	0.972	0.770	**0.997**
		TPR (FPR@0.01)	0.01	0.55	0.57	**0.86**	0.54	0.52	0.41	**0.96**	0.04	0.29	0.04	**0.82**
		TPR (FPR@0.05)	0.09	**0.98**	0.95	**0.98**	0.61	0.85	0.86	**0.98**	0.20	0.82	0.16	**0.99**
		TPR (FPR@0.10)	0.22	**0.99**	0.95	**0.99**	0.62	0.91	0.91	**0.98**	0.29	0.93	0.38	**0.99**
	DenseNet	AUC	0.679	0.958	0.966	**0.977**	0.955	0.952	0.768	**0.997**	0.790	0.706	0.829	**1.000**
		TPR (FPR@0.01)	0.06	0.30	**0.48**	0.33	0.69	0.51	0.03	**0.99**	0.17	0.04	0.00	**0.99**
		TPR (FPR@0.05)	0.13	0.79	**0.91**	0.84	0.74	0.84	0.23	**0.99**	0.28	0.12	0.29	**0.99**
		TPR (FPR@0.10)	0.22	0.91	0.94	**0.98**	0.80	0.88	0.31	**0.99**	0.41	0.23	0.51	**0.99**
CIFAR100	ResNet	AUC	0.637	0.717	0.945	**0.967**	0.855	0.984	0.966	**0.999**	0.773	0.985	0.875	**1.000**
		TPR (FPR@0.01)	0.07	0.00	0.00	**0.33**	0.59	0.69	0.48	**0.94**	0.39	0.48	0.12	**0.99**
		TPR (FPR@0.05)	0.16	0.01	0.52	**0.70**	0.61	0.94	0.82	**0.99**	0.49	0.89	0.43	**0.99**
		TPR (FPR@0.10)	0.29	0.01	0.80	**0.92**	0.64	0.96	0.92	**0.99**	0.56	0.99	0.57	**0.99**
	DenseNet	AUC	0.567	0.727	0.916	**0.958**	0.549	0.732	0.947	**0.971**	0.577	0.751	0.951	**0.974**
		TPR (FPR@0.01)	0.02	**0.07**	0.00	**0.07**	0.01	0.00	0.00	**0.21**	0.01	0.01	0.00	**0.31**
		TPR (FPR@0.05)	0.17	0.15	0.61	**0.66**	0.14	0.01	0.70	**0.75**	0.17	0.06	0.77	**0.81**
		TPR (FPR@0.10)	0.22	0.26	0.84	**0.88**	0.20	0.04	0.91	**0.96**	0.23	0.18	0.93	**0.94**

Data	Model	Metric	Attacks											
			JSMA				DeepFool				Boundary			
			KD+BU	LID	MAHA	ML-LOO	KD+BU	LID	MAHA	ML-LOO	KD+BU	LID	MAHA	ML-LOO
MNIST	CNN	AUC	0.886	**1.000**	0.976	**1.000**	0.901	**1.000**	0.869	**1.000**	0.905	**1.000**	0.991	**1.000**
		TPR (FPR@0.01)	0.30	**1.00**	0.87	0.99	0.32	**1.00**	0.04	**1.00**	0.32	**1.00**	0.79	**1.00**
		TPR (FPR@0.05)	0.46	**1.00**	0.94	**1.00**	0.43	**1.00**	0.36	**1.00**	0.45	**1.00**	0.98	**1.00**
		TPR (FPR@0.10)	0.51	**1.00**	0.95	**1.00**	0.57	**1.00**	0.59	**1.00**	0.55	**1.00**	0.98	**1.00**
CIFAR10	ResNet	AUC	0.614	**0.986**	0.941	0.981	0.618	0.990	0.981	**0.994**	0.676	0.990	0.967	**0.997**
		TPR (FPR@0.01)	0.01	**0.49**	0.45	0.46	0.01	0.57	0.60	**0.89**	0.03	0.64	0.60	**0.92**
		TPR (FPR@0.05)	0.10	**0.98**	0.87	0.82	0.10	**0.99**	0.96	0.96	0.20	**0.99**	0.94	**0.99**
		TPR (FPR@0.10)	0.21	**0.99**	0.90	**0.99**	0.24	**0.99**	0.96	**0.99**	0.38	**0.99**	0.94	**0.99**
	DenseNet	AUC	0.645	0.937	0.947	**0.964**	0.646	0.976	**0.977**	0.976	0.700	**0.983**	0.981	0.980
		TPR (FPR@0.01)	0.04	0.14	**0.41**	0.12	0.03	0.34	**0.51**	0.24	0.05	0.58	**0.62**	0.31
		TPR (FPR@0.05)	0.10	0.67	0.68	**0.72**	0.09	0.90	**0.95**	0.82	0.12	**0.93**	0.91	0.89
		TPR (FPR@0.10)	0.18	0.86	0.88	**0.96**	0.17	**0.98**	0.97	**0.98**	0.23	**0.98**	0.96	**0.98**
CIFAR100	ResNet	AUC	0.600	0.740	0.907	**0.964**	0.610	0.714	0.953	**0.970**	0.635	0.732	0.956	**0.972**
		TPR (FPR@0.01)	0.00	0.01	0.00	**0.42**	0.06	0.00	0.00	**0.41**	0.07	0.01	0.00	**0.49**
		TPR (FPR@0.05)	0.12	0.14	0.49	**0.70**	0.14	0.01	0.56	**0.74**	0.16	0.07	0.61	**0.78**
		TPR (FPR@0.10)	0.27	0.24	0.77	**0.91**	0.29	0.01	0.87	**0.94**	0.30	0.15	**0.94**	0.93
	DenseNet	AUC	0.567	0.727	0.916	**0.958**	0.549	0.732	0.947	**0.971**	0.577	0.751	0.951	**0.974**
		TPR (FPR@0.01)	0.02	**0.07**	0.00	**0.07**	0.01	0.00	0.00	**0.21**	0.01	0.01	0.00	**0.31**
		TPR (FPR@0.05)	0.17	0.15	0.61	**0.66**	0.14	0.01	0.70	**0.75**	0.17	0.06	0.77	**0.81**
		TPR (FPR@0.10)	0.22	0.26	0.84	**0.88**	0.20	0.04	0.91	**0.96**	0.23	0.18	0.93	**0.94**

Table 1: Performance of detection methods on different data sets, models and attack methods.

the contribution of feature i to neuron n. For Leave-One-Out (LOO), we have

$$\phi_{f_n}(x)_i = f_n(x) - f_n(x_{(i)}).$$

To coordinate the scale difference between different neurons, we fit a logistic regression for the dispersion of feature attribution from different neurons on a hold-out training set to distinguish adversarial images from original images. The multi-layer extension of our method is called 'ML-LOO'.

Experiments

We present an experimental evaluation of ML-LOO, and compare our method with several state-of-the-art detection methods. Then we consider the setting where attacks have different confidence levels. We further evaluate the transferability of various detection methods on an unknown attack. Finally, we evaluate the performance of our method under the white-box attacker who knows the existence of our detector. The code for ML-LOO is available at our Github page.

Known attacks

We compare our method with state-of-the-art detection algorithms including LID (Ma et al. 2018), Mahalanobis (MAHA) (Lee et al. 2018), and KD+BU (Feinman et al. 2017), on three data sets: MNIST, CIFAR-10 and CIFAR-100, with the standard train/test split (Chollet and others 2015). We used a convolutional network composed of 32-filter convolutional layers followed by a hidden dense layer with 1024 units for MNIST. Each convolutional layer was followed by a max-pooling layer. For both CIFAR-10 and CIFAR-100, we trained a 20-layer ResNet (He et al. 2016) and 121-layer DenseNet (Huang et al. 2017) respectively. For each data set, we generated 2,000 adversarial examples from correctly classified test images by each attacking method. Among them, 1,000 adversarial images with the corresponding 1,000 natural images were used for the training process of LID, Mahalanobis and our method. Results are reported for the other 1,000 adversarial images with the corresponding natural images. We consider the following attacking methods, grouped by the norms they are optimized for:

- L_∞: FGSM (Goodfellow, Shlens, and Szegedy 2015), L_∞-PGD (Kurakin, Goodfellow, and Bengio 2017; Madry et al. 2018).
- ℓ_2: C&W (Carlini and Wagner 2017b), DeepFool (Moosavi-Dezfooli, Fawzi, and Frossard 2016), Boundary Attack (Brendel, Rauber, and Bethge 2018).
- ℓ_0: JSMA (Papernot et al. 2016a).

Let true positive rate (TPR) be the proportion of adversarial

Data	Model	Metric	Attacks											
			C&W MIX				C&W LC				C&W HC			
			KD+BU	LID	MAHA	ML-LOO	KD+BU	LID	MAHA	ML-LOO	KD+BU	LID	MAHA	ML-LOO
CIFAR10	ResNet	AUC	0.620	0.649	0.640	**0.840**	0.623	0.445	0.641	**0.711**	0.829	0.816	0.966	**0.988**
		TPR (FPR@0.01)	0.04	0.01	0.03	**0.25**	0.01	0.00	0.01	**0.12**	0.52	0.23	0.51	**0.87**
		TPR (FPR@0.05)	0.17	0.06	0.14	**0.42**	0.09	0.06	0.10	**0.21**	0.59	0.43	0.90	**0.94**
		TPR (FPR@0.10)	0.28	0.19	0.21	**0.59**	0.22	0.11	0.16	**0.34**	0.60	0.62	0.93	**0.97**

Data	Model	Metric	Attacks											
			ℓ_∞-PGD-MIX				ℓ_∞-PGD-LC				ℓ_∞-PGD-HC			
			KD+BU	LID	MAHA	ML-LOO	KD+BU	LID	MAHA	ML-LOO	KD+BU	LID	MAHA	ML-LOO
CIFAR10	ResNet	AUC	0.753	0.812	0.813	**0.953**	0.606	0.578	0.578	**0.767**	0.834	0.935	0.962	**0.996**
		TPR (FPR@0.01)	0.20	0.10	0.11	**0.60**	0.01	0.01	0.01	**0.09**	0.54	0.26	0.46	**0.89**
		TPR (FPR@0.05)	0.37	0.36	0.45	**0.77**	0.12	0.07	0.04	**0.23**	0.61	0.67	0.89	**0.98**
		TPR (FPR@0.10)	0.46	0.41	0.56	**0.84**	0.25	0.17	0.12	**0.33**	0.62	0.85	0.91	**0.99**

Table 2: Top: Performance of detection methods trained with C&W-MIX and tested on C&W-LC, C&W-HC and C&W-MIX. Bottom: Performance of detection methods trained with ℓ_∞-PGD-MIX and tested on ℓ_∞-PGD-LC, ℓ_∞-PGD-HC and ℓ_∞-PGD-MIX.

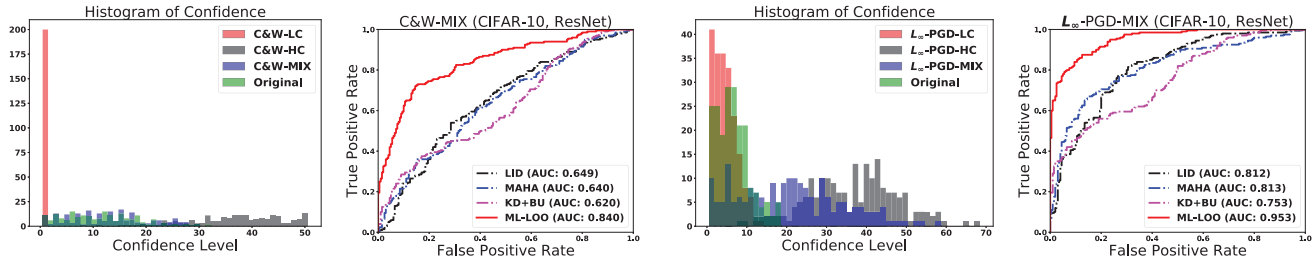

Figure 4: The left two figures plot the histogram of confidence levels of C&W-LC, C&W-HC, and C&W-MIX, and the ROC curves of detection methods under C&W-MIX attack. The right two figures plot the histogram of confidence levels of ℓ_∞-PGD-LC, ℓ_∞-PGD-HC, and ℓ_∞-PGD-MIX, and the ROC curves of detection methods under ℓ_∞-PGD-MIX attack.

images classified as adversarial, and false positive rate (FPR) be the proportion of natural images classified as adversarial. We report area under the curve (AUC) of the ROC curve as the performance evaluation as well as the true positive rates by thresholding FPR at 0.01,0.05 and 0.1, as it is practical to keep misclassified natural images at a low proportion.

The results are reported in Table 1, and the ROC curves on CIFAR-10 with ResNet are shown in Figure 3. The rest of the plots can be found in the supplementary material. ML-LOO shows superior performance over the other three detection methods across different data sets, models for all attacks optimized for ℓ_2 and ℓ_∞ distances. By controlling FPR at 0.1, our method is able to find over 95% adversarial examples generated by most existing attacks.

Attacks with varied confidence levels

Lu, Chen, and Yu and Athalye, Carlini, and Wagner observed that LID fails when the confidence level of adversarial examples generated from C&W attack varies. We consider adversarial images with varied confidence levels for both ℓ_2 and ℓ_∞ attacks. We use C&W attack for optimizing ℓ_2 distance, and adjust the confidence hyperparameter c in Equation (2) to achieve mixed confidence levels. To achieve adversarial examples optimized for ℓ_∞ distance, we use ℓ_∞-PGD for optimizing ℓ_∞ distance, and vary the constraint ε for different confidence levels.

C&W Attack for optimizing ℓ_2 distance We consider three settings for C&W attack, low-confidence (C&W-LC), mixed-confidence (C&W-MIX) and high-confidence (C&W-HC). We set the confidence parameter $c = 0$ for C&W-LC and $c = 50$ for C&W-HC. For mixed-confidence C&W attack, we generate adversarial images from C&W attack with the confidence parameter in Equation (2) randomly selected from $\{1, 3, 5, \cdots, 29\}$ when generating an adversarial image, so that the distribution of confidence levels for adversarial images is comparable with that of original images. The confidence levels of images under the three settings, along with confidence levels of original images are shown in Figure 4. The confidence level in Figure 4 is defined as $-\log(1-p)$, where p is the probability score of the predicted class.

We carried out the experiments on ResNet trained on CIFAR-10 using $1,000$ adversarial images generated from the mixed-confidence C&W attack, together with the corresponding original images, as the training data for LID, Mahalanobis, KD+BU, and our method. We test the detection methods on a different set of original and adversarial images generated from three versions: low-confidence C&W attack ($c = 0$), high-confidence C&W attack ($c = 50$), and the mixed-confidence C&W attack. Table 2 (Top) and Figure 4 (Left) show TPRs at different FPR thresholds, AUC, and the ROC curve. Mahalanobis, LID and KD+BU fail to detect adversarial examples of mixed-confidence effectively, while our method performs consistently better for adversarial images across the three settings.

Data	Model	Metric	Attacks																			
			ℓ_∞-PGD				DeepFool				FGSM				JSAM				Boundary			
			KD+BU	LID	MAHA	ML-LOO	KD+BU	LID	MAHA	ML-LOO	KD+BU	LID	MAHA	ML-LOO	KD+BU	LID	MAHA	ML-LOO	KD+BU	LID	MAHA	ML-LOO
CIFAR10	ResNet	AUC	0.753	0.763	0.818	**0.879**	0.618	0.990	0.962	**0.992**	0.673	0.610	0.730	**0.796**	0.614	0.984	0.957	**0.984**	0.676	0.991	0.964	**0.994**
		TPR (FPR@0.01)	0.20	0.08	0.14	**0.21**	0.01	0.56	0.61	**0.72**	0.04	**0.07**	0.06	0.04	0.01	0.43	0.44	**0.45**	0.03	0.56	0.60	**0.82**
		TPR (FPR@0.05)	0.37	0.35	0.45	**0.48**	0.10	0.96	0.94	**0.96**	0.20	0.17	**0.22**	0.14	0.10	**0.93**	0.91	0.91	0.20	**0.99**	0.95	0.97
		TPR (FPR@0.10)	0.46	0.45	0.60	**0.65**	0.24	0.98	0.94	**0.99**	0.29	0.23	0.34	**0.37**	0.21	0.98	0.94	**0.99**	0.38	**0.99**	0.95	**0.99**

Table 3: Performance of detection methods trained with C&W and transferred to ℓ_∞-PGD, FGSM, JSMA, Boundary and DeepFool.

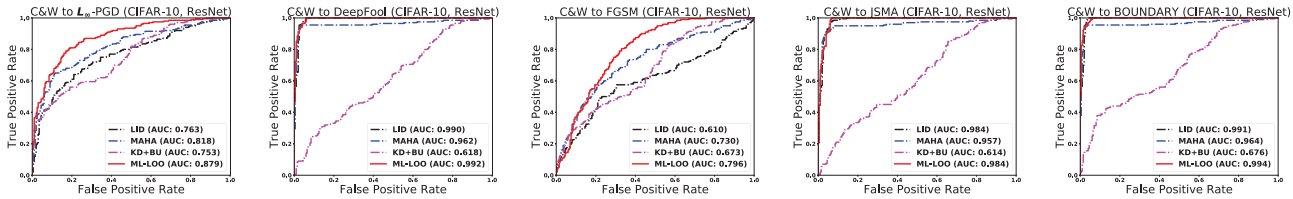

Figure 5: Transferability of detection methods trained with C&W attack and tested on ℓ_∞-PGD, FGSM, JSMA, Boundary and DeepFool.

Detector	None	SD	ML-LOO(SD)	ML-LOO(IQR)
Succ. rate on Model	100%	93%	52%	58%
Succ. rate on Detector	N/A	100%	100%	78%
Succ. rate on Both	100%	93%	52%	**36%**
Avg. ℓ_2 distance	0.31	0.43	**1.23**	1.07

Table 4: Performance under the white-box attacks.

L_∞-PGD for optimizing ℓ_∞ distance L_∞-PGD (Madry et al. 2018), also named as BIM (Kurakin, Goodfellow, and Bengio 2017), searches for adversarial examples by iteratively updating the original image with the following:

$$x_{N+1} = \text{Clip}_{x,\varepsilon}\{x_N + \alpha \text{sign}(\nabla_X J(x_N, y_{\text{true}}))\}, \quad (3)$$

where y_{true} is the original class, J is the cross-entropy loss, and Clip operator clips an image elementwise to an ε-neighborhood. For mixed-confidence ℓ_∞-PGD attack, we generated adversarial images from ℓ_∞-PGD with different confidence levels by randomly selecting the constraint ε in Equation (3) from $\{1, 2, 3, 4, 5, 6, 7, 8\}/255$. The confidence levels of images from mixed-confidence ℓ_∞-PGD attack are shown in Figure 4.

We used $1,000$ adversarial images generated from the mixed-confidence ℓ_∞-PGD, together with their corresponding original images, as the training data for all detection methods. We report the results on adversarial images generated from three versions: high-confidence ℓ_∞-PGD ($\varepsilon = 0.03$), low-confidence ℓ_∞-PGD ($\varepsilon = 0.005$), and the mixed-confidence ℓ_∞-PGD that is used to generate the training data. The corresponding original images are different from the training images. Table 2 (Bottom) and Figure 4 (Right) show TPRs at different FPR thresholds, AUC, and the ROC curve. Mahalanobis, LID and KD+BU fail to detect adversarial examples of mixed-confidence effectively, while our method performs significantly better across the three settings.

Transferability

In this experiment, we evaluate the transferability of different methods by training detection methods on adversarial examples generated from one attacking method and carry out the evaluation on adversarial examples generated from different attacking methods. We trained all methods on adversarial examples generated by C&W attack and carried out the evaluation on adversarial examples generated by the rest of the attacking methods.

Experiments are carried out on MNIST, CIFAR-10, and CIFAR-100 data sets. AUC and TPRs at different FPR thresholds are reported in Table 3. All methods trained on C&W attack are capable of detecting adversarial examples generated from an unknown attack, even when the optimized distance is ℓ_∞, or the attack is not gradient-based. The same phenomenon has been observed in Lee et al. as well. This indicates attacks might share some common features. Our method yields a slightly higher AUC consistently, and has a significantly higher TPR when FPRs are controlled to be small.

White-box evaluation

The previous experiments are carried out in a "gray-box" threat model, where the attacker has access to the model details such as gradients, but does not have access to the design of the detector. The "white-box" setting assumes a stronger threat model, where an attacker knows exactly how our detector is constructed and its parameters. Such a setting is often missing in previous study of adversarial detection. Previous work such as LID and KD+BU has been shown to fail under this setting (Carlini and Wagner 2017a; Athalye, Carlini, and Wagner 2018). We evaluate the performance of ML-LOO in this setting.

We carried out the white-box attack on CIFAR-10 with the ResNet. The attacker aims to optimize the following objective

$$\min_w L(x') = \|x' - x\|^2 + c_1 L_{\text{C\&W}}(x') + c_2 L_{\text{DET}}(x'),$$

where $x' = 0.5(\tanh(w) + 1)$, the C&W loss $L_{\text{C\&W}} = \max\{F(x)_{y_{\text{true}}}, 0\}$, and L_{DET} aims at controlling the statistic used by the detector, which will be defined differently under different scenarios below. For each image, we increase c_2 gradually until adversarial images cannot be detected (at

FPR=0.05) at all. For each c_2, c_1 is selected via binary search. The loss is minimized with Adam (Kingma and Ba 2014). Under this scheme, the generated examples are expected to fool the detector, and we aim to check whether it fools the original model at an acceptable perturbation size as well. We will evaluate three variants of ML-LOO, including the simplest output layer standard deviation (SD) thresholding, ML-LOO (SD), and ML-LOO (IQR). We measure the performance by the success rate on the original model, the detector, and the rate of fooling both simultaneously. We also report the average ℓ_2 distance between original and successful adversarial images. The results are summarized in Table 4.

In the first scenario, we evaluate the robustness of the single-layer variate of ML-LOO (SD) to demonstrate the power of our attacker design, which only thresholds the SD of the probability of the predicted class alone. We define the detector loss as $L^1_{\text{DET}} := \max\{\text{SD}(\phi(x')) - \tau, 0\}$, which penalizes $\text{SD}(\phi(x'))$ over attribution scores if it is larger than τ. The threshold τ is chosen to keep the FPR at 0.05 when detecting adversarial examples generated by gray-box C&W attack from natural images. When LOO over all pixels is intractable, we sample pixels to estimate the SD. The attacker always fool the detector. However, the success rate of fooling the original model decreases from 100% to 93%, and the average distance increases from 0.31 to 0.43.

In the second scenario, We use ML-LOO (with SD) as the detector (a Logistic Regression (LR) applied to SD of multi-layer feature attributions). The white-box attack in the first setting fails to fool this detector completely. Therefore, we define the detector loss as

$$L^2_{\text{DET}} := \max\{\sum\nolimits_n w_n \text{SD}(\phi_{f_n}(x)) - \tau, 0\},$$

where n loops over neurons of selected layers, $\phi_{f_n}(x)$ is the attribution score at neuron n, and w_n is the corresponding learned coefficients. The threshold τ is still chosen at FPR = 0.05. The success rate of fooling the model decreases from 100% to 52%, and the average distance increases to 1.23.

In the third scenario, we evaluate ML-LOO (IQR), which is non-differentiable. However, there is an approximately linear relationship between SD and IQR under the normality assumption (Royston 1982), which suggests that we can apply the same detector loss L^2_{DET} with the transformed threshold at FPR = 0.05, and the transformed coefficients learned in ML-LOO (IQR). The attacker achieves worse performance. In particular, only 78% generated images fool the detector, with a 58% success rate of fooling the model (over all the examples) and an average distance of 1.07.

We observe that ML-LOO (IQR) achieves competitive performance even under the strongest white-box threat model. The white-box attack fails to fool the model and the detector simultaneously for 64% of test images. The average size of successful perturbations also increases by over three times. We expect ML-LOO works better under a white-box attack for adversarially trained models with larger certified radii.

Discussion

In this paper, we introduce a new framework to detect adversarial examples with multi-layer feature attribution, by capturing the scaling difference of feature attribution scores between the original and adversarial examples. We show that our detection method outperforms other state-of-the-art methods in detecting various kinds of attacks. It also displays strong performance in detecting adversarial examples of varied confidence levels, in detecting transferred examples from other attacks, and when an attacker has complete access to the detector.

One limitation of ML-LOO is its query inefficiency. In the detection stage, the number of queries scales with the number of features for each input. Future work may address this issue by randomly sampling pixels to compute feature attribution scores.

References

Athalye, A.; Carlini, N.; and Wagner, D. 2018. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, 274–283.

Bach, S.; Binder, A.; Montavon, G.; Klauschen, F.; Müller, K.-R.; and Samek, W. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS One* 10(7):e0130140.

Bhagoji, A. N.and Cullina, D.; Sitawarin, C.; and Mittal, P. 2018. Enhancing robustness of machine learning systems via data transformations. In *CISS*, 1–5. IEEE.

Brendel, W.; Rauber, J.; and Bethge, M. 2018. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. In *ICLR*.

Carlini, N., and Wagner, D. 2017a. Adversarial examples are not easily detected: Bypassing ten detection methods. In *AISec*, 3–14. ACM.

Carlini, N., and Wagner, D. 2017b. Towards evaluating the robustness of neural networks. In *IEEE SP*, 39–57. IEEE.

Chalasani, P.; Jha, S.; Sadagopan, A.; and Wu, X. 2018. Adversarial learning and explainability in structured datasets. *arXiv preprint arXiv:1810.06583*.

Chen, P.-Y.; Zhang, H.; Sharma, Y.; Yi, J.; and Hsieh, C.-J. 2017. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *AISec*, 15–26. ACM.

Chen, J.; Song, L.; Wainwright, M. J.; and Jordan, M. I. 2019. L-shapley and C-shapley: Efficient model interpretation for structured data. In *ICLR*.

Chollet, F., et al. 2015. Keras. https://github.com/fchollet/keras.

Datta, A.; Sen, S.; and Zick, Y. 2016. Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *IEEE SP*, 598–617. IEEE.

Dvijotham, K.; Gowal, S.; Stanforth, R.; Arandjelovic, R.; O'Donoghue, B.; Uesato, J.; and Kohli, P. 2018. Training verified learners with learned verifiers. *arXiv preprint arXiv:1805.10265*.

Fawzi, A.; Fawzi, O.; and Frossard, P. 2018. Analysis of classifiers' robustness to adversarial perturbations. *Machine Learning* 107(3):481–508.

Feinman, R.; Curtin, R. R.; Shintre, S.; and Gardner, A. B. 2017. Detecting adversarial samples from artifacts. *arXiv preprint arXiv:1703.00410*.

Ghorbani, A.; Abid, A.; and Zou, J. 2017. Interpretation of neural networks is fragile. *arXiv preprint arXiv:1710.10547*.

Gong, Z.; Wang, W.; and Ku, W.-S. 2017. Adversarial and clean data are not twins. *arXiv preprint arXiv:1704.04960*.

Goodfellow, I.; Papernot, N.; Huang, S.; Duan, Y.; and Abbeel, P. 2017. Attacking machine learning with adversarial examples. https://blog.openai.com/adversarial-example-research/.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2015. Explaining and harnessing adversarial examples. *ICLR*.

Grosse, K.; Manoharan, P.; Papernot, N.; Backes, M.; and McDaniel, P. 2017. On the (statistical) detection of adversarial examples. *arXiv preprint arXiv:1702.06280*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Identity mappings in deep residual networks. In *ECCV*, 630–645. Springer.

Hendrycks, D., and Gimpel, K. 2017. Early methods for detecting adversarial images. In *ICLR*.

Huang, G.; Liu, Z.; Van Der Maaten, L.; and Weinberger, K. Q. 2017. Densely connected convolutional networks. In *CVPR*, 4700–4708.

Ilyas, A.; Engstrom, L.; Athalye, A.; and Lin, J. 2018. Black-box adversarial attacks with limited queries and information. In *ICML*, 2142–2151.

Ilyas, A.; Engstrom, L.; and Madry, A. 2019. Prior convictions: Black-box adversarial attacks with bandits and priors. In *ICLR*.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. In *ICLR*.

Kurakin, A.; Goodfellow, I.; and Bengio, S. 2017. Adversarial machine learning at scale. In *ICLR*.

Lecuyer, M.; Atlidakis, V.; Geambasu, R.; Hsu, D.; and Jana, S. 2019. Certified robustness to adversarial examples with differential privacy. In *IEEE SP*.

Lee, K.; Lee, K.; Lee, H.; and Shin, J. 2018. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *NeurIPS*, 7167–7177.

Li, X., and Li, F. 2017. Adversarial examples detection in deep networks with convolutional filter statistics. In *ICCV*, 5764–5772.

Li, J.; Monroe, W.; and Jurafsky, D. 2016. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*.

Liu, X., and Hsieh, C.-J. 2019. Rob-gan: Generator, discriminator, and adversarial attacker. In *CVPR*.

Liu, X.; Cheng, M.; Zhang, H.; and Hsieh, C.-J. 2018. Towards robust neural networks via random self-ensemble. In *ECCV*, 369–385.

Liu, X.; Li, Y.; Wu, C.; and Hsieh, C.-J. 2019. Adv-bnn: Improved adversarial defense through robust bayesian neural network. In *ICLR*.

Lu, P.-H.; Chen, P.-Y.; and Yu, C.-M. 2018. On the limitation of local intrinsic dimensionality for characterizing the subspaces of adversarial examples. *arXiv preprint arXiv:1803.09638*.

Lundberg, S. M., and Lee, S.-I. 2017. A unified approach to interpreting model predictions. In *NeurIPS*, 4765–4774.

Ma, X.; Li, B.; Wang, Y.; Erfani, S. M.; Wijewickrema, S.; Schoenebeck, G.; Houle, M. E.; Song, D.; and Bailey, J. 2018. Characterizing adversarial subspaces using local intrinsic dimensionality. In *ICLR*.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2018. Towards deep learning models resistant to adversarial attacks. In *ICLR*.

Metzen, J. H.; Genewein, T.; Fischer, V.; and Bischoff, B. 2017. On detecting adversarial perturbations. In *ICLR*.

Miyato, T.; Maeda, S.-I.; Koyama, M.; Nakae, K.; and Ishii, S. 2016. Distributional smoothing with virtual adversarial training. In *ICLR*.

Moosavi-Dezfooli, S.-M.; Fawzi, A.; and Frossard, P. 2016. Deepfool: a simple and accurate method to fool deep neural networks. In *CVPR*, 2574–2582.

Papernot, N.; McDaniel, P.; Jha, S.; Fredrikson, M.; Celik, Z. B.; and Swami, A. 2016a. The limitations of deep learning in adversarial settings. In *EuroS&P*, 372–387. IEEE.

Papernot, N.; McDaniel, P.; Wu, X.; Jha, S.; and Swami, A. 2016b. Distillation as a defense to adversarial perturbations against deep neural networks. In *IEEE SP*, 582–597. IEEE.

Ribeiro, M. T.; Singh, S.; and Guestrin, C. 2016. Why should I trust you?: Explaining the predictions of any classifier. In *SIGKDD*, 1135–1144. ACM.

Royston, J. 1982. Algorithm as 177: Expected normal order statistics (exact and approximate). *Journal of the royal statistical society. Series C (Applied statistics)* 31(2):161–165.

Schmidt, L.; Santurkar, S.; Tsipras, D.; Talwar, K.; and Madry, A. 2018. Adversarially robust generalization requires more data. In *NeurIPS*, 5019–5031.

Shrikumar, A.; Greenside, P.; and Kundaje, A. 2017. Learning important features through propagating activation differences. In *ICML*, volume 70 of *PMLR*, 3145–3153. PMLR.

Simonyan, K.; Vedaldi, A.; and Zisserman, A. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.

Song, Y.; Kim, T.; Nowozin, S.; Ermon, S.; and Kushman, N. 2018. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. In *ICLR*.

Sundararajan, M.; Taly, A.; and Yan, Q. 2017. Axiomatic attribution for deep networks. In *ICML*, 3319–3328.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2014. Intriguing properties of neural networks. In *ICLR*.

Tanay, T., and Griffin, L. 2016. A boundary tilting persepective on the phenomenon of adversarial examples. *arXiv preprint arXiv:1608.07690*.

Tao, G.; Ma, S.; Liu, Y.; and Zhang, X. 2018. Attacks meet interpretability: Attribute-steered detection of adversarial samples. In *NeurIPS*, 7728–7739.

Tramèr, F.; Kurakin, A.; Papernot, N.; Goodfellow, I.; Boneh, D.; and McDaniel, P. 2018. Ensemble adversarial training: Attacks and defenses. In *ICLR*.

Tsipras, D.; Santurkar, S.; Engstrom, L.; Turner, A.; and Madry, A. 2018. There is no free lunch in adversarial robustness (but there are unexpected benefits). *arXiv preprint arXiv:1805.12152*.

Wong, E., and Kolter, J. Z. 2018. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *ICML*.

Xu, W.; Evans, D.; and Qi, Y. 2018. Feature squeezing: Detecting adversarial examples in deep neural networks. In *NDSS*.

Yeh, C.-K.; Hsieh, C.-Y.; Suggala, A. S.; Inouye, D.; and Ravikumar, P. 2019. How sensitive are sensitivity-based explanations? *arXiv preprint arXiv:1901.09392*.

Zeiler, M. D., and Fergus, R. 2014. Visualizing and understanding convolutional networks. In *ECCV*, 818–833. Springer.

Zhang, C.; Ye, Z.; Wang, Y.; and Yang, Z. 2018. Detecting adversarial perturbations with saliency. In *ICSIP*, 271–275. IEEE.