The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

EAD: Elastic-Net Attacks to Deep Neural Networks via Adversarial Examples

Pin-Yu Chen,[1]* Yash Sharma,[2]*† Huan Zhang,[3]† Jinfeng Yi,[4]‡ Cho-Jui Hsieh,[3]
[1]AI Foundations Lab, IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, USA
[2]The Cooper Union, New York, NY 10003, USA
[3]University of California, Davis, Davis, CA 95616, USA
[4]Tencent AI Lab, Bellevue, WA 98004, USA
pin-yu.chen@ibm.com, ysharma1126@gmail.com, ecezhang@ucdavis.edu,
jinfengy@us.ibm.com, chohsieh@ucdavis.edu

Abstract

Recent studies have highlighted the vulnerability of deep neural networks (DNNs) to adversarial examples - a visually indistinguishable adversarial image can easily be crafted to cause a well-trained model to misclassify. Existing methods for crafting adversarial examples are based on L_2 and L_∞ distortion metrics. However, despite the fact that L_1 distortion accounts for the total variation and encourages sparsity in the perturbation, little has been developed for crafting L_1-based adversarial examples.

In this paper, we formulate the process of attacking DNNs via adversarial examples as an elastic-net regularized optimization problem. Our **e**lastic-net **a**ttacks to **D**NNs (EAD) feature L_1-oriented adversarial examples and include the state-of-the-art L_2 attack as a special case. Experimental results on MNIST, CIFAR10 and ImageNet show that EAD can yield a distinct set of adversarial examples with small L_1 distortion and attains similar attack performance to the state-of-the-art methods in different attack scenarios. More importantly, EAD leads to improved attack transferability and complements adversarial training for DNNs, suggesting novel insights on leveraging L_1 distortion in adversarial machine learning and security implications of DNNs.

Introduction

Deep neural networks (DNNs) achieve state-of-the-art performance in various tasks in machine learning and artificial intelligence, such as image classification, speech recognition, machine translation and game-playing. Despite their effectiveness, recent studies have illustrated the vulnerability of DNNs to adversarial examples (Szegedy et al. 2013; Goodfellow, Shlens, and Szegedy 2015). For instance, a carefully designed perturbation to an image can lead a well-trained DNN to misclassify. Even worse, effective adversarial examples can also be made virtually indistinguishable to human perception. For example, Figure 1 shows three adversarial examples of an ostrich image crafted by our algorithm,

(a) Image (b) Adversarial examples with target class labels

Figure 1: Visual illustration of adversarial examples crafted by EAD (Algorithm 1). The original example is an ostrich image selected from the ImageNet dataset (Figure 1 (a)). The adversarial examples in Figure 1 (b) are classified as the target class labels by the Inception-v3 model.

which are classified as "safe", "shoe shop" and "vacuum" by the Inception-v3 model (Szegedy et al. 2016), a state-of-the-art image classification model.

The lack of robustness exhibited by DNNs to adversarial examples has raised serious concerns for security-critical applications, including traffic sign identification and malware detection, among others. Moreover, moving beyond the digital space, researchers have shown that these adversarial examples are still effective in the physical world at fooling DNNs (Kurakin, Goodfellow, and Bengio 2016a; Evtimov et al. 2017). Due to the robustness and security implications, the means of crafting adversarial examples are called *attacks* to DNNs. In particular, *targeted attacks* aim to craft adversarial examples that are misclassified as specific target classes, and *untargeted attacks* aim to craft adversarial examples that are not classified as the original class. *Transfer attacks* aim to craft adversarial examples that are transferable from one DNN model to another. In addition to evaluating the robustness of DNNs, adversarial examples can be used to train a robust model that is resilient to adversarial perturbations, known as *adversarial training* (Madry et al. 2017). They have also been used in interpreting DNNs (Koh and Liang 2017; Dong et al. 2017).

Throughout this paper, we use adversarial examples to attack image classifiers based on deep convolutional neural networks. The rationale behind crafting effective adversarial examples lies in manipulating the prediction results while

*Pin-Yu Chen and Yash Sharma contribute equally to this work.
†This work was done during the internship of Yash Sharma and Huan Zhang at IBM T. J. Watson Research Center.
‡Part of the work was done when Jinfeng Yi was at AI Foundations Lab, IBM T. J. Watson Research Center.
Copyright © 2018, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

ensuring similarity to the original image. Specifically, in the literature the similarity between original and adversarial examples has been measured by different distortion metrics. One commonly used distortion metric is the L_q norm, where $\|\mathbf{x}\|_q = (\sum_{i=1}^{p} |\mathbf{x}_i|^q)^{1/q}$ denotes the L_q norm of a p-dimensional vector $\mathbf{x} = [\mathbf{x}_1, \ldots, \mathbf{x}_p]$ for any $q \geq 1$. In particular, when crafting adversarial examples, the L_∞ distortion metric is used to evaluate the maximum variation in pixel value changes (Goodfellow, Shlens, and Szegedy 2015), while the L_2 distortion metric is used to improve the visual quality (Carlini and Wagner 2017b). However, despite the fact that the L_1 norm is widely used in problems related to image denoising and restoration (Fu et al. 2006), as well as sparse recovery (Candès and Wakin 2008), L_1-based adversarial examples have not been rigorously explored. In the context of adversarial examples, L_1 distortion accounts for the total variation in the perturbation and serves as a popular convex surrogate function of the L_0 metric, which measures the number of modified pixels (i.e., sparsity) by the perturbation. To bridge this gap, we propose an attack algorithm based on elastic-net regularization, which we call **e**lastic-net **a**ttacks to **D**NNs (EAD). Elastic-net regularization is a linear mixture of L_1 and L_2 penalty functions, and it has been a standard tool for high-dimensional feature selection problems (Zou and Hastie 2005). In the context of attacking DNNs, EAD opens up new research directions since it generalizes the state-of-the-art attack proposed in (Carlini and Wagner 2017b) based on L_2 distortion, and is able to craft L_1-oriented adversarial examples that are more effective and fundamentally different from existing attack methods.

To explore the utility of L_1-based adversarial examples crafted by EAD, we conduct extensive experiments on MNIST, CIFAR10 and ImageNet in different attack scenarios. Compared to the state-of-the-art L_2 and L_∞ attacks (Kurakin, Goodfellow, and Bengio 2016b; Carlini and Wagner 2017b), EAD can attain similar attack success rate when breaking undefended and defensively distilled DNNs (Papernot et al. 2016b). More importantly, we find that L_1 attacks attain superior performance over L_2 and L_∞ attacks in transfer attacks and complement adversarial training. For the most difficult dataset (MNIST), EAD results in improved attack transferability from an undefended DNN to a defensively distilled DNN, achieving nearly 99% attack success rate. In addition, joint adversarial training with L_1 and L_2 based examples can further enhance the resilience of DNNs to adversarial perturbations. These results suggest that EAD yields a distinct, yet more effective, set of adversarial examples. Moreover, evaluating attacks based on L_1 distortion provides novel insights on adversarial machine learning and security implications of DNNs, suggesting that L_1 may complement L_2 and L_∞ based examples toward furthering a thorough adversarial machine learning framework.

Related Work

Here we summarize related works on attacking and defending DNNs against adversarial examples.

Attacks to DNNs

FGM and I-FGM: Let $\mathbf{x}_0$ and $\mathbf{x}$ denote the original and adversarial examples, respectively, and let t denote the target class to attack. Fast gradient methods (FGM) use the gradient ∇J of the training loss J with respect to $\mathbf{x}_0$ for crafting adversarial examples (Goodfellow, Shlens, and Szegedy 2015). For L_∞ attacks, $\mathbf{x}$ is crafted by

$$\mathbf{x} = \mathbf{x}_0 - \epsilon \cdot \text{sign}(\nabla J(\mathbf{x}_0, t)), \tag{1}$$

where ϵ specifies the L_∞ distortion between $\mathbf{x}$ and $\mathbf{x}_0$, and $\text{sign}(\nabla J)$ takes the sign of the gradient. For L_1 and L_2 attacks, $\mathbf{x}$ is crafted by

$$\mathbf{x} = \mathbf{x}_0 - \epsilon \frac{\nabla J(\mathbf{x}_0, t)}{\|\nabla J(\mathbf{x}_0, t)\|_q} \tag{2}$$

for $q = 1, 2$, where ϵ specifies the corresponding distortion. Iterative fast gradient methods (I-FGM) were proposed in (Kurakin, Goodfellow, and Bengio 2016b), which iteratively use FGM with a finer distortion, followed by an ϵ-ball clipping. Untargeted attacks using FGM and I-FGM can be implemented in a similar fashion.

C&W attack: Instead of leveraging the training loss, Carlini and Wagner designed an L_2-regularized loss function based on the logit layer representation in DNNs for crafting adversarial examples (Carlini and Wagner 2017b). Its formulation turns out to be a special case of our EAD formulation, which will be discussed in the following section. The C&W attack is considered to be one of the strongest attacks to DNNs, as it can successfully break undefended and defensively distilled DNNs and can attain remarkable attack transferability.

JSMA: Papernot et al. proposed a Jacobian-based saliency map algorithm (JSMA) for characterizing the input-output relation of DNNs (Papernot et al. 2016a). It can be viewed as a greedy attack algorithm that iteratively modifies the most influential pixel for crafting adversarial examples.

DeepFool: DeepFool is an untargeted L_2 attack algorithm (Moosavi-Dezfooli, Fawzi, and Frossard 2016) based on the theory of projection to the closest separating hyperplane in classification. It is also used to craft a universal perturbation to mislead DNNs trained on natural images (Moosavi-Dezfooli et al. 2016).

Black-box attacks: Crafting adversarial examples in the black-box case is plausible if one allows querying of the target DNN. In (Papernot et al. 2017), JSMA is used to train a substitute model for transfer attacks. In (Chen et al. 2017), an effective black-box C&W attack is made possible using zeroth order optimization (ZOO). In the more stringent attack scenario where querying is prohibited, ensemble methods can be used for transfer attacks (Liu et al. 2016).

Defenses in DNNs

Defensive distillation: Defensive distillation (Papernot et al. 2016b) defends against adversarial perturbations by using the distillation technique in (Hinton, Vinyals, and Dean 2015) to retrain the same network with class probabilities predicted by the original network. It also introduces the temperature parameter T in the softmax layer to enhance the robustness to adversarial perturbations.

Adversarial training: Adversarial training can be implemented in a few different ways. A standard approach is augmenting the original training dataset with the label-corrected adversarial examples to retrain the network. Modifying the training loss or the network architecture to increase the robustness of DNNs to adversarial examples has been proposed in (Zheng et al. 2016; Madry et al. 2017; Tramèr et al. 2017; Zantedeschi, Nicolae, and Rawat 2017).

Detection methods: Detection methods utilize statistical tests to differentiate adversarial from benign examples (Feinman et al. 2017; Grosse et al. 2017; Lu, Issaranon, and Forsyth 2017; Xu, Evans, and Qi 2017). However, 10 different detection methods were unable to detect the C&W attack (Carlini and Wagner 2017a).

EAD: Elastic-Net Attacks to DNNs

Preliminaries on Elastic-Net Regularization

Elastic-net regularization is a widely used technique in solving high-dimensional feature selection problems (Zou and Hastie 2005). It can be viewed as a regularizer that linearly combines L_1 and L_2 penalty functions. In general, elastic-net regularization is used in the following minimization problem:

$$\text{minimize}_{\mathbf{z}\in\mathcal{Z}}\ f(\mathbf{z}) + \lambda_1\|\mathbf{z}\|_1 + \lambda_2\|\mathbf{z}\|_2^2, \tag{3}$$

where $\mathbf{z}$ is a vector of p optimization variables, $\mathcal{Z}$ indicates the set of feasible solutions, $f(\mathbf{z})$ denotes a loss function, $\|\mathbf{z}\|_q$ denotes the L_q norm of $\mathbf{z}$, and $\lambda_1, \lambda_2 \geq 0$ are the L_1 and L_2 regularization parameters, respectively. The term $\lambda_1\|\mathbf{z}\|_1 + \lambda_2\|\mathbf{z}\|_2^2$ in (3) is called the elastic-net regularizer of $\mathbf{z}$. For standard regression problems, the loss function $f(\mathbf{z})$ is the mean squared error, the vector $\mathbf{z}$ represents the weights (coefficients) on the features, and the set $\mathcal{Z} = \mathbb{R}^p$. In particular, the elastic-net regularization in (3) degenerates to the LASSO formulation when $\lambda_2 = 0$, and becomes the ridge regression formulation when $\lambda_1 = 0$. It is shown in (Zou and Hastie 2005) that elastic-net regularization is able to select a group of highly correlated features, which overcomes the shortcoming of high-dimensional feature selection when solely using the LASSO or ridge regression techniques.

EAD Formulation and Generalization

Inspired by the C&W attack (Carlini and Wagner 2017b), we adopt the same loss function f for crafting adversarial examples. Specifically, given an image $\mathbf{x}_0$ and its correct label denoted by t_0, let $\mathbf{x}$ denote the adversarial example of $\mathbf{x}_0$ with a target class $t \neq t_0$. The loss function $f(\mathbf{x})$ for targeted attacks is defined as

$$f(\mathbf{x}, t) = \max\{\max_{j\neq t}[\mathbf{Logit}(\mathbf{x})]_j - [\mathbf{Logit}(\mathbf{x})]_t, -\kappa\}, \tag{4}$$

where $\mathbf{Logit}(\mathbf{x}) = [[\mathbf{Logit}(\mathbf{x})]_1, \ldots, [\mathbf{Logit}(\mathbf{x})]_K] \in \mathbb{R}^K$ is the logit layer (the layer prior to the softmax layer) representation of $\mathbf{x}$ in the considered DNN, K is the number of classes for classification, and $\kappa \geq 0$ is a confidence parameter that guarantees a constant gap between $\max_{j\neq t}[\mathbf{Logit}(\mathbf{x})]_j$ and $[\mathbf{Logit}(\mathbf{x})]_t$.

It is worth noting that the term $[\mathbf{Logit}(\mathbf{x})]_t$ is proportional to the probability of predicting $\mathbf{x}$ as label t, since by the softmax classification rule,

$$\text{Prob}(\text{Label}(\mathbf{x}) = t) = \frac{\exp([\mathbf{Logit}(\mathbf{x})]_t)}{\sum_{j=1}^K \exp([\mathbf{Logit}(\mathbf{x})]_j)}. \tag{5}$$

Consequently, the loss function in (4) aims to render the label t the most probable class for $\mathbf{x}$, and the parameter κ controls the separation between t and the next most likely prediction among all classes other than t. For untargeted attacks, the loss function in (4) can be modified as

$$f(\mathbf{x}) = \max\{[\mathbf{Logit}(\mathbf{x})]_{t_0} - \max_{j\neq t}[\mathbf{Logit}(\mathbf{x})]_j, -\kappa\}. \tag{6}$$

In this paper, we focus on targeted attacks since they are more challenging than untargeted attacks. Our EAD algorithm (Algorithm 1) can directly be applied to untargeted attacks by replacing $f(\mathbf{x}, t)$ in (4) with $f(\mathbf{x})$ in (6).

In addition to manipulating the prediction via the loss function in (4), introducing elastic-net regularization further encourages similarity to the original image when crafting adversarial examples. Our formulation of elastic-net attacks to DNNs (EAD) for crafting an adversarial example $(\mathbf{x}, t)$ with respect to a labeled natural image $(\mathbf{x}_0, t_0)$ is as follows:

$$\begin{aligned} &\text{minimize}_{\mathbf{x}}\ \ c\cdot f(\mathbf{x}, t) + \beta\|\mathbf{x}-\mathbf{x}_0\|_1 + \|\mathbf{x}-\mathbf{x}_0\|_2^2 \\ &\text{subject to}\ \ \mathbf{x}\in[0,1]^p, \end{aligned} \tag{7}$$

where $f(\mathbf{x}, t)$ is as defined in (4), $c, \beta \geq 0$ are the regularization parameters of the loss function f and the L_1 penalty, respectively. The box constraint $\mathbf{x} \in [0,1]^p$ restricts $\mathbf{x}$ to a properly scaled image space, which can be easily satisfied by dividing each pixel value by the maximum attainable value (e.g., 255). Upon defining the perturbation of $\mathbf{x}$ relative to $\mathbf{x}_0$ as $\boldsymbol{\delta} = \mathbf{x} - \mathbf{x}_0$, the EAD formulation in (7) aims to find an adversarial example $\mathbf{x}$ that will be classified as the target class t while minimizing the distortion in $\boldsymbol{\delta}$ in terms of the elastic-net loss $\beta\|\boldsymbol{\delta}\|_1 + \|\boldsymbol{\delta}\|_2^2$, which is a linear combination of L_1 and L_2 distortion metrics between $\mathbf{x}$ and $\mathbf{x}_0$. Notably, the formulation of the C&W attack (Carlini and Wagner 2017b) becomes a special case of the EAD formulation in (7) when $\beta = 0$, which disregards the L_1 penalty on $\boldsymbol{\delta}$. However, the L_1 penalty is an intuitive regularizer for crafting adversarial examples, as $\|\boldsymbol{\delta}\|_1 = \sum_{i=1}^p |\boldsymbol{\delta}_i|$ represents the total variation of the perturbation, and is also a widely used surrogate function for promoting sparsity in the perturbation. As will be evident in the performance evaluation section, including the L_1 penalty for the perturbation indeed yields a distinct set of adversarial examples, and it leads to improved attack transferability and complements adversarial learning.

EAD Algorithm

When solving the EAD formulation in (7) without the L_1 penalty (i.e., $\beta = 0$), Carlini and Wagner used a change-of-variable (COV) approach via the $\tanh$ transformation on $\mathbf{x}$ in order to remove the box constraint $\mathbf{x} \in [0,1]^p$ (Carlini and Wagner 2017b). When $\beta > 0$, we find that the same COV approach is not effective in solving (7), since the corresponding adversarial examples are insensitive to the changes in β (see the performance evaluation section for details). Since the L_1 penalty is a non-differentiable, yet piece-wise linear,

function, the failure of the COV approach in solving (7) can be explained by its inefficiency in subgradient-based optimization problems (Duchi and Singer 2009).

To efficiently solve the EAD formulation in (7) for crafting adversarial examples, we propose to use the iterative shrinkage-thresholding algorithm (ISTA) (Beck and Teboulle 2009). ISTA can be viewed as a regular first-order optimization algorithm with an additional shrinkage-thresholding step on each iteration. In particular, let $g(\mathbf{x}) = c \cdot f(\mathbf{x}) + \|\mathbf{x} - \mathbf{x}_0\|_2^2$ and let $\nabla g(\mathbf{x})$ be the numerical gradient of $g(\mathbf{x})$ computed by the DNN. At the $k+1$-th iteration, the adversarial example $\mathbf{x}^{(k+1)}$ of $\mathbf{x}_0$ is computed by

$$\mathbf{x}^{(k+1)} = S_\beta(\mathbf{x}^{(k)} - \alpha_k \nabla g(\mathbf{x}^{(k)})), \tag{8}$$

where α_k denotes the step size at the $k+1$-th iteration, and $S_\beta : \mathbb{R}^p \mapsto \mathbb{R}^p$ is an element-wise projected shrinkage-thresholding function, which is defined as

$$[S_\beta(\mathbf{z})]_i = \begin{cases} \min\{\mathbf{z}_i - \beta, 1\}, & \text{if } \mathbf{z}_i - \mathbf{x}_{0i} > \beta; \\ \mathbf{x}_{0i}, & \text{if } |\mathbf{z}_i - \mathbf{x}_{0i}| \leq \beta; \\ \max\{\mathbf{z}_i + \beta, 0\}, & \text{if } \mathbf{z}_i - \mathbf{x}_{0i} < -\beta, \end{cases} \tag{9}$$

for any $i \in \{1, \ldots, p\}$. If $|\mathbf{z}_i - \mathbf{x}_{0i}| > \beta$, it shrinks the element $\mathbf{z}_i$ by β and projects the resulting element to the feasible box constraint between 0 and 1. On the other hand, if $|\mathbf{z}_i - \mathbf{x}_{0i}| \leq \beta$, it thresholds $\mathbf{z}_i$ by setting $[S_\beta(\mathbf{z})]_i = \mathbf{x}_{0i}$. The proof of optimality of using (8) for solving the EAD formulation in (7) is given in the supplementary material[1]. Notably, since $g(\mathbf{x})$ is the attack objective function of the C&W method (Carlini and Wagner 2017b), the ISTA operation in (8) can be viewed as a robust version of the C&W method that shrinks a pixel value of the adversarial example if the deviation to the original image is greater than β, and keeps a pixel value unchanged if the deviation is less than β.

Our EAD algorithm for crafting adversarial examples is summarized in Algorithm 1. For computational efficiency, a fast ISTA (FISTA) for EAD is implemented, which yields the optimal convergence rate for first-order optimization methods (Beck and Teboulle 2009). The slack vector $\mathbf{y}^{(k)}$ in Algorithm 1 incorporates the momentum in $\mathbf{x}^{(k)}$ for acceleration. In the experiments, we set the initial learning rate $\alpha_0 = 0.01$ with a square-root decay factor in k. During the EAD iterations, the iterate $\mathbf{x}^{(k)}$ is considered as a successful adversarial example of $\mathbf{x}_0$ if the model predicts its most likely class to be the target class t. The final adversarial example $\mathbf{x}$ is selected from all successful examples based on distortion metrics. In this paper we consider two decision rules for selecting $\mathbf{x}$: the least elastic-net (EN) and L_1 distortions relative to $\mathbf{x}_0$. The influence of β, κ and the decision rules on EAD will be investigated in the following section.

Algorithm 1 Elastic-Net Attacks to DNNs (EAD)

Input: original labeled image $(\mathbf{x}_0, t_0)$, target attack class t, attack transferability parameter κ, L_1 regularization parameter β, step size α_k, # of iterations I
Output: adversarial example $\mathbf{x}$
Initialization: $\mathbf{x}^{(0)} = \mathbf{y}^{(0)} = \mathbf{x}_0$
for $k = 0$ to $I - 1$ **do**
  $\mathbf{x}^{(k+1)} = S_\beta(\mathbf{y}^{(k)} - \alpha_k \nabla g(\mathbf{y}^{(k)}))$
  $\mathbf{y}^{(k+1)} = \mathbf{x}^{(k+1)} + \frac{k}{k+3}(\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)})$
end for
Decision rule: determine $\mathbf{x}$ from successful examples in $\{\mathbf{x}^{(k)}\}_{k=1}^{I}$ (EN rule or L_1 rule).

Performance Evaluation

In this section, we compare the proposed EAD with the state-of-the-art attacks to DNNs on three image classification datasets - MNIST, CIFAR10 and ImageNet. We would like to show that (i) EAD can attain attack performance similar to the C&W attack in breaking undefended and defensively distilled DNNs, since the C&W attack is a special case of EAD when $\beta = 0$; (ii) Comparing to existing L_1-based FGM and I-FGM methods, the adversarial examples using EAD can lead to significantly lower L_1 distortion and better attack success rate; (iii) The L_1-based adversarial examples crafted by EAD can achieve improved attack transferability and complement adversarial training.

Comparative Methods

We compare EAD with the following targeted attacks, which are the most effective methods for crafting adversarial examples in different distortion metrics.

C&W attack: The state-of-the-art L_2 targeted attack proposed by Carlini and Wagner (Carlini and Wagner 2017b), which is a special case of EAD when $\beta = 0$.

FGM: The fast gradient method proposed in (Goodfellow, Shlens, and Szegedy 2015). The FGM attacks using different distortion metrics are denoted by FGM-L_1, FGM-L_2 and FGM-L_∞.

I-FGM: The iterative fast gradient method proposed in (Kurakin, Goodfellow, and Bengio 2016b). The I-FGM attacks using different distortion metrics are denoted by I-FGM-L_1, I-FGM-L_2 and I-FGM-L_∞.

Experiment Setup and Parameter Setting

Our experiment setup is based on Carlini and Wagner's framework[2]. For both the EAD and C&W attacks, we use the default setting[1], which implements 9 binary search steps on the regularization parameter c (starting from 0.001) and runs $I = 1000$ iterations for each step with the initial learning rate $\alpha_0 = 0.01$. For finding successful adversarial examples, we use the reference optimizer[1] (ADAM) for the C&W attack and implement the projected FISTA (Algorithm 1) with the square-root decaying learning rate for EAD. Similar to the C&W attack, the final adversarial example of EAD is selected by the least distorted example among all the successful examples. The sensitivity analysis of the L_1 parameter β and the effect of the decision rule on EAD will be investigated in the forthcoming paragraph. Unless specified, we set the attack transferability parameter $\kappa = 0$ for both attacks.

[1] https://arxiv.org/abs/1709.04114

[2] https://github.com/carlini/nn_robust_attacks

Table 1: Comparison of the change-of-variable (COV) approach and EAD (Algorithm 1) for solving the elastic-net formulation in (7) on MNIST. ASR means attack success rate (%). Although these two methods attain similar attack success rates, COV is not effective in crafting L_1-based adversarial examples. Increasing β leads to less L_1-distorted adversarial examples for EAD, whereas the distortion of COV is insensitive to changes in β.

		Best case				Average case				Worst case			
Optimization method	β	ASR	L_1	L_2	L_∞	ASR	L_1	L_2	L_∞	ASR	L_1	L_2	L_∞
COV	0	100	13.93	1.377	0.379	100	22.46	1.972	0.514	99.9	32.3	2.639	0.663
	10^{-5}	100	13.92	1.377	0.379	100	22.66	1.98	0.508	99.5	32.33	2.64	0.663
	10^{-4}	100	13.91	1.377	0.379	100	23.11	2.013	0.517	100	32.32	2.639	0.664
	10^{-3}	100	13.8	1.377	0.381	100	22.42	1.977	0.512	99.9	32.2	2.639	0.664
	10^{-2}	100	12.98	1.38	0.389	100	22.27	2.026	0.53	99.5	31.41	2.643	0.673
EAD (EN rule)	0	100	14.04	1.369	0.376	100	22.63	1.953	0.512	99.8	31.43	2.51	0.644
	10^{-5}	100	13.66	1.369	0.378	100	22.6	1.98	0.515	99.9	30.79	2.507	0.648
	10^{-4}	100	12.79	1.372	0.388	100	20.98	1.951	0.521	100	29.21	2.514	0.667
	10^{-3}	100	9.808	1.427	0.452	100	17.4	2.001	0.594	100	25.52	2.582	0.748
	10^{-2}	100	7.271	1.718	0.674	100	13.56	2.395	0.852	100	20.77	3.021	0.976

We implemented FGM and I-FGM using the CleverHans package[3]. The best distortion parameter ϵ is determined by a fine-grained grid search - for each image, the smallest ϵ in the grid leading to a successful attack is reported. For I-FGM, we perform 10 FGM iterations (the default value) with ϵ-ball clipping. The distortion parameter ϵ' in each FGM iteration is set to be $\epsilon/10$, which has been shown to be an effective attack setting in (Tramèr et al. 2017). The range of the grid and the resolution of these two methods are specified in the supplementary material[1].

The image classifiers for MNIST and CIFAR10 are trained based on the DNN models provided by Carlini and Wagner[1]. The image classifier for ImageNet is the Inception-v3 model (Szegedy et al. 2016). For MNIST and CIFAR10, 1000 correctly classified images are randomly selected from the test sets to attack an incorrect class label. For ImageNet, 100 correctly classified images and 9 incorrect classes are randomly selected to attack. All experiments are conducted on a machine with an Intel E5-2690 v3 CPU, 40 GB RAM and a single NVIDIA K80 GPU. Our EAD code is publicly available for download[4].

Evaluation Metrics

Following the attack evaluation criterion in (Carlini and Wagner 2017b), we report the attack success rate and distortion of the adversarial examples from each method. The attack success rate (ASR) is defined as the percentage of adversarial examples that are classified as the target class (which is different from the original class). The average L_1, L_2 and L_∞ distortion metrics of successful adversarial examples are also reported. In particular, the ASR and distortion of the following attack settings are considered:

Best case: The least difficult attack among targeted attacks to all incorrect class labels in terms of distortion.

Average case: The targeted attack to a randomly selected incorrect class label.

Worst case: The most difficult attack among targeted attacks to all incorrect class labels in terms of distortion.

Sensitivity Analysis and Decision Rule for EAD

We verify the necessity of using Algorithm 1 for solving the elastic-net regularized attack formulation in (7) by comparing it to a naive change-of-variable (COV) approach. In (Carlini and Wagner 2017b), Carlini and Wagner remove the box constraint $\mathbf{x} \in [0,1]^p$ by replacing $\mathbf{x}$ with $\frac{\mathbf{1}+\tanh \mathbf{w}}{2}$, where $\mathbf{w} \in \mathbb{R}^p$ and $\mathbf{1} \in \mathbb{R}^p$ is a vector of ones. The default ADAM optimizer (Kingma and Ba 2014) is then used to solve $\mathbf{w}$ and obtain $\mathbf{x}$. We apply this COV approach to (7) and compare with EAD on MNIST with different orders of the L_1 regularization parameter β in Table 1. Although COV and EAD attain similar attack success rates, it is observed that COV is not effective in crafting L_1-based adversarial examples. Increasing β leads to less L_1-distorted adversarial examples for EAD, whereas the distortion (L_1, L_2 and L_∞) of COV is insensitive to changes in β. Similar insensitivity of COV on β is observed when one uses other optimizers such as AdaGrad, RMSProp or built-in SGD in TensorFlow. We also note that the COV approach prohibits the use of ISTA due to the subsequent $\tanh$ term in the L_1 penalty. The insensitivity of COV suggests that it is inadequate for elastic-net optimization, which can be explained by its inefficiency in subgradient-based optimization problems (Duchi and Singer 2009). For EAD, we also find an interesting trade-off between L_1 and the other two distortion metrics - adversarial examples with smaller L_1 distortion tend to have larger L_2 and L_∞ distortions. This trade-off can be explained by the fact that increasing β further encourages sparsity in the perturbation, and hence results in increased L_2 and L_∞ distortion. Similar results are observed on CIFAR10 (see supplementary material[1]).

In Table 1, during the attack optimization process the final adversarial example is selected based on the elastic-net loss of all successful adversarial examples in $\{\mathbf{x}^{(k)}\}_{k=1}^{I}$, which we call the *elastic-net (EN) decision rule*. Alternatively, we can

[3]https://github.com/tensorflow/cleverhans

[4]https://github.com/ysharma1126/EAD-Attack

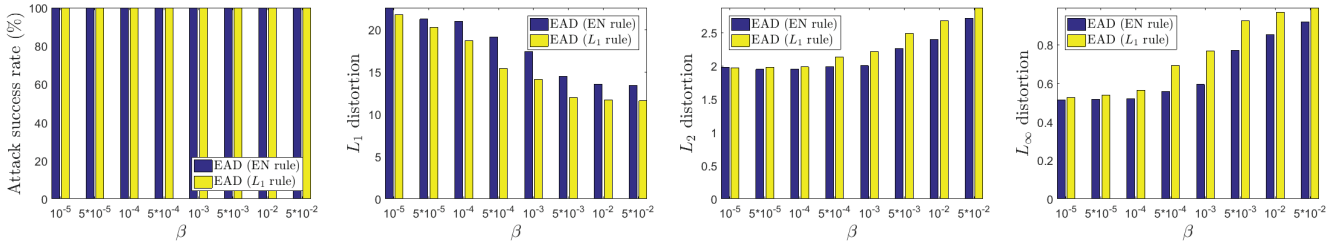

Figure 2: Comparison of EN and L_1 decision rules in EAD on MNIST with varying L_1 regularization parameter β (average case). Comparing to the EN rule, for the same β the L_1 rule attains less L_1 distortion but may incur more L_2 and L_∞ distortions.

Table 2: Comparison of different attacks on MNIST, CIFAR10 and ImageNet (average case). ASR means attack success rate (%). The distortion metrics are averaged over successful examples. EAD, the C&W attack, and I-FGM-L_∞ attain the least L_1, L_2, and L_∞ distorted adversarial examples, respectively. The complete attack results are given in the supplementary material[1].

	MNIST				CIFAR10				ImageNet			
Attack method	ASR	L_1	L_2	L_∞	ASR	L_1	L_2	L_∞	ASR	L_1	L_2	L_∞
C&W (L_2)	**100**	22.46	**1.972**	0.514	**100**	13.62	**0.392**	0.044	**100**	232.2	**0.705**	0.03
FGM-L_1	39	53.5	4.186	0.782	48.8	51.97	1.48	0.152	1	61	0.187	0.007
FGM-L_2	34.6	39.15	3.284	0.747	42.8	39.5	1.157	0.136	1	2338	6.823	0.25
FGM-L_∞	42.5	127.2	6.09	0.296	52.3	127.81	2.373	0.047	3	3655	7.102	0.014
I-FGM-L_1	**100**	32.94	2.606	0.591	**100**	17.53	0.502	0.055	77	526.4	1.609	0.054
I-FGM-L_2	**100**	30.32	2.41	0.561	**100**	17.12	0.489	0.054	**100**	774.1	2.358	0.086
I-FGM-L_∞	**100**	71.39	3.472	**0.227**	**100**	33.3	0.68	**0.018**	**100**	864.2	2.079	**0.01**
EAD (EN rule)	**100**	**17.4**	2.001	0.594	**100**	**8.18**	0.502	0.097	**100**	**69.47**	1.563	0.238
EAD (L_1 rule)	**100**	**14.11**	2.211	0.768	**100**	**6.066**	0.613	0.17	**100**	**40.9**	1.598	0.293

select the final adversarial example with the least L_1 distortion, which we call the L_1 *decision rule*. Figure 2 compares the ASR and average-case distortion of these two decision rules with different β on MNIST. Both decision rules yield 100% ASR for a wide range of β values. For the same β, the L_1 rule gives adversarial examples with less L_1 distortion than those given by the EN rule at the price of larger L_2 and L_∞ distortions. Similar trends are observed on CIFAR10 (see supplementary material[1]). The complete results of these two rules on MNIST and CIFAR10 are given in the supplementary material[1]. In the following experiments, we will report the results of EAD with these two decision rules and set $\beta = 10^{-3}$, since on MNIST and CIFAR10 this β value significantly reduces the L_1 distortion while having comparable L_2 and L_∞ distortions to the case of $\beta = 0$ (i.e., without L_1 regularization).

Attack Success Rate and Distortion on MNIST, CIFAR10 and ImageNet

We compare EAD with the comparative methods in terms of attack success rate and different distortion metrics on attacking the considered DNNs trained on MNIST, CIFAR10 and ImageNet. Table 2 summarizes their average-case performance. It is observed that FGM methods fail to yield successful adversarial examples (i.e., low ASR), and the corresponding distortion metrics are significantly larger than other methods. On the other hand, the C&W attack, I-FGM and EAD all lead to 100% attack success rate. Furthermore, EAD, the C&W method, and I-FGM-L_∞ attain the least L_1, L_2, and L_∞ distorted adversarial examples, respectively. We note that EAD significantly outperforms the existing L_1-based method (I-FGM-L_1). Compared to I-FGM-L_1, EAD with the EN decision rule reduces the L_1 distortion by roughly 47% on MNIST, 53% on CIFAR10 and 87% on ImageNet. We also observe that EAD with the L_1 decision rule can further reduce the L_1 distortion but at the price of noticeable increase in the L_2 and L_∞ distortion metrics.

Notably, despite having large L_2 and L_∞ distortion metrics, the adversarial examples crafted by EAD with the L_1 rule can still attain 100% ASRs in all datasets, which implies the L_2 and L_∞ distortion metrics are insufficient for evaluating the robustness of neural networks. Moreover, the attack results in Table 2 suggest that EAD can yield a set of distinct adversarial examples that are fundamentally different from L_2 or L_∞ based examples. Similar to the C&W method and I-FGM, the adversarial examples from EAD are also visually indistinguishable (see supplementary material[1]).

Breaking Defensive Distillation

In addition to breaking undefended DNNs via adversarial examples, here we show that EAD can also break defensively distilled DNNs. Defensive distillation (Papernot et al. 2016b) is a standard defense technique that retrains the network with class label probabilities predicted by the original network, soft labels, and introduces the temperature parameter T in the softmax layer to enhance its robustness to adversarial perturbations. Similar to the state-of-the-art attack (the C&W method), Figure 3 shows that EAD can attain 100% attack

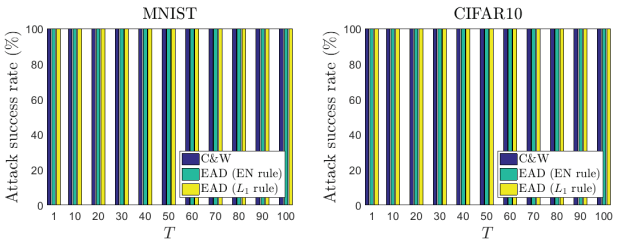

Figure 3: Attack success rate (average case) of the C&W method and EAD on MNIST and CIFAR10 with respect to varying temperature parameter T for defensive distillation. Both methods can successfully break defensive distillation.

success rate for different values of T on MNIST and CIFAR10. Moreover, since the C&W attack formulation is a special case of the EAD formulation in (7) when $\beta = 0$, successfully breaking defensive distillation using EAD suggests new ways of crafting effective adversarial examples by varying the L_1 regularization parameter β. The complete attack results are given in the supplementary material[1].

Improved Attack Transferability

It has been shown in (Carlini and Wagner 2017b) that the C&W attack can be made highly transferable from an undefended network to a defensively distilled network by tuning the confidence parameter κ in (4). Following (Carlini and Wagner 2017b), we adopt the same experiment setting for attack transferability on MNIST, as MNIST is the most difficult dataset to attack in terms of the average distortion per image pixel from Table 2.

Fixing κ, adversarial examples generated from the original (undefended) network are used to attack the defensively distilled network with the temperature parameter $T = 100$ (Papernot et al. 2016b). The attack success rate (ASR) of EAD, the C&W method and I-FGM are shown in Figure 4. When $\kappa = 0$, all methods attain low ASR and hence do not produce transferable adversarial examples. The ASR of EAD and the C&W method improves when we set $\kappa > 0$, whereas I-FGM's ASR remains low (less than 2%) since the attack does not have such a parameter for transferability.

Notably, EAD can attain nearly 99% ASR when $\kappa = 50$, whereas the top ASR of the C&W method is nearly 88% when $\kappa = 40$. This implies improved attack transferability when using the adversarial examples crafted by EAD, which can be explained by the fact that the ISTA operation in (8) is a robust version of the C&W attack via shrinking and thresholding. We also find that setting κ too large may mitigate the ASR of transfer attacks for both EAD and the C&W method, as the optimizer may fail to find an adversarial example that minimizes the loss function f in (4) for large κ. The complete attack transferability results are given in the supplementary material[1].

Complementing Adversarial Training

To further validate the difference between L_1-based and L_2-based adversarial examples, we test their performance in

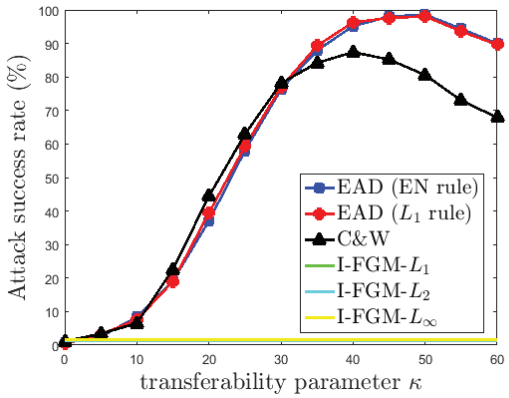

Figure 4: Attack transferability (average case) from the undefended network to the defensively distilled network on MNIST by varying κ. EAD can attain nearly 99% attack success rate (ASR) when $\kappa = 50$, whereas the top ASR of the C&W attack is nearly 88% when $\kappa = 40$.

Table 3: Adversarial training using the C&W attack and EAD (L_1 rule) on MNIST. ASR means attack success rate. Incorporating L_1 examples complements adversarial training and enhances attack difficulty in terms of distortion. The complete results are given in the supplementary material[1].

Attack method	Adversarial training	Average case ASR	L_1	L_2	L_∞
C&W (L_2)	None	100	22.46	1.972	0.514
	EAD	100	26.11	2.468	0.643
	C&W	100	24.97	2.47	0.684
	EAD + C&W	100	27.32	2.513	0.653
EAD (L_1 rule)	None	100	14.11	2.211	0.768
	EAD	100	17.04	2.653	0.86
	C&W	100	15.49	2.628	0.892
	EAD + C&W	100	16.83	2.66	0.87

adversarial training on MNIST. We randomly select 1000 images from the training set and use the C&W attack and EAD (L_1 rule) to generate adversarial examples for all incorrect labels, leading to 9000 adversarial examples in total for each method. We then separately augment the original training set with these examples to retrain the network and test its robustness on the testing set, as summarized in Table 3. For adversarial training with any single method, although both attacks still attain a 100% success rate in the average case, the network is more tolerable to adversarial perturbations, as all distortion metrics increase significantly when compared to the null case. We also observe that joint adversarial training with EAD and the C&W method can further increase the L_1 and L_2 distortions against the C&W attack and the L_2 distortion against EAD, suggesting that the L_1-based examples crafted by EAD can complement adversarial training.

Conclusion

We proposed an elastic-net regularized attack framework for crafting adversarial examples to attack deep neural networks.

Experimental results on MNIST, CIFAR10 and ImageNet show that the L_1-based adversarial examples crafted by EAD can be as successful as the state-of-the-art L_2 and L_∞ attacks in breaking undefended and defensively distilled networks. Furthermore, EAD can improve attack transferability and complement adversarial training. Our results corroborate the effectiveness of EAD and shed new light on the use of L_1-based adversarial examples toward adversarial learning and security implications of deep neural networks.

Acknowledgment Cho-Jui Hsieh and Huan Zhang acknowledge the support of NSF via IIS-1719097.

References

Beck, A., and Teboulle, M. 2009. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences* 2(1):183–202.

Candès, E. J., and Wakin, M. B. 2008. An introduction to compressive sampling. *IEEE signal processing magazine* 25(2):21–30.

Carlini, N., and Wagner, D. 2017a. Adversarial examples are not easily detected: Bypassing ten detection methods. *arXiv preprint arXiv:1705.07263*.

Carlini, N., and Wagner, D. 2017b. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy (SP)*, 39–57.

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 *ACM Workshop on Artificial Intelligence and Security*, 15–26.

Dong, Y.; Su, H.; Zhu, J.; and Bao, F. 2017. Towards interpretable deep neural networks by leveraging adversarial examples. *arXiv preprint arXiv:1708.05493*.

Duchi, J., and Singer, Y. 2009. Efficient online and batch learning using forward backward splitting. *Journal of Machine Learning Research* 10(Dec):2899–2934.

Evtimov, I.; Eykholt, K.; Fernandes, E.; Kohno, T.; Li, B.; Prakash, A.; Rahmati, A.; and Song, D. 2017. Robust physical-world attacks on machine learning models. *arXiv preprint arXiv:1707.08945*.

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

Fu, H.; Ng, M. K.; Nikolova, M.; and Barlow, J. L. 2006. Efficient minimization methods of mixed l2-l1 and l1-l1 norms for image restoration. *SIAM Journal on scientific computing* 27(6):1881–1902.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2015. Explaining and harnessing adversarial examples. *ICLR'15; arXiv preprint arXiv:1412.6572*.

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

Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Kingma, D., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Koh, P. W., and Liang, P. 2017. Understanding black-box predictions via influence functions. *ICML; arXiv preprint arXiv:1703.04730*.

Kurakin, A.; Goodfellow, I.; and Bengio, S. 2016a. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*.

Kurakin, A.; Goodfellow, I.; and Bengio, S. 2016b. Adversarial machine learning at scale. *ICLR'17; arXiv preprint arXiv:1611.01236*.

Liu, Y.; Chen, X.; Liu, C.; and Song, D. 2016. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*.

Lu, J.; Issaranon, T.; and Forsyth, D. 2017. Safetynet: Detecting and rejecting adversarial examples robustly. *arXiv preprint arXiv:1704.00103*.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Moosavi-Dezfooli, S.-M.; Fawzi, A.; Fawzi, O.; and Frossard, P. 2016. Universal adversarial perturbations. *arXiv preprint arXiv:1610.08401*.

Moosavi-Dezfooli, S.-M.; Fawzi, A.; and Frossard, P. 2016. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 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 *IEEE European Symposium on Security and Privacy (EuroS&P)*, 372–387.

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 Symposium on Security and Privacy (SP)*, 582–597.

Papernot, N.; McDaniel, P.; Goodfellow, I.; Jha, S.; Celik, Z. B.; and Swami, A. 2017. Practical black-box attacks against machine learning. In *ACM Asia Conference on Computer and Communications Security*, 506–519.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; and Wojna, Z. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2818–2826.

Tramèr, F.; Kurakin, A.; Papernot, N.; Boneh, D.; and McDaniel, P. 2017. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*.

Xu, W.; Evans, D.; and Qi, Y. 2017. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*.

Zantedeschi, V.; Nicolae, M.-I.; and Rawat, A. 2017. Efficient defenses against adversarial attacks. *arXiv preprint arXiv:1707.06728*.

Zheng, S.; Song, Y.; Leung, T.; and Goodfellow, I. 2016. Improving the robustness of deep neural networks via stability training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4480–4488.

Zou, H., and Hastie, T. 2005. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67(2):301–320.