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

1AI Foundations Lab, IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, USA
2The Cooper Union, New York, NY 10003, USA
3University of California, Davis, Davis, CA 95616, USA
4Tencent 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 $L_{\infty}$ 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 elastic-net attacks to DNNs (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, which are classified as “safe”, “shoe shop” and “vacuum” by the Inception-v3 model.

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.

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...
DeepFool is an untargeted crafting adversarial examples in the Wagner 2017b) based on where the $L_q$ 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 elastic-net attacks to DNNs (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 $L_\infty$ 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 $L_\infty$ 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 $L_\infty$ 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 $x_0$ and $x$ denote the original and adversarial examples, respectively, and let $t$ denote the target class to attack. Fast gradient methods (FGM) use the gradient $\nabla J$ of the training loss $J$ with respect to $x_0$ for crafting adversarial examples (Goodfellow, Shlens, and Szegedy 2015). For $L_\infty$ attacks, $x$ is crafted by

$$x = x_0 - \epsilon \cdot \text{sign}(\nabla J(x_0, t)),$$

where $\epsilon$ specifies the $L_\infty$ distortion between $x$ and $x_0$, and $\text{sign}(\nabla J)$ takes the sign of the gradient. For $L_1$ and $L_2$ attacks, $x$ is crafted by

$$x = x_0 - \epsilon \cdot \frac{\nabla J(x_0, t)}{\|\nabla J(x_0, t)\|_q}$$

for $q = 1, 2$, where $\epsilon$ 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 $\epsilon$-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 (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:

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

(3)

where $z$ is a vector of $p$ optimization variables, $\mathbb{Z}$ indicates the set of feasible solutions, $f(z)$ denotes a loss function, $\|z\|_q$ denotes the $L_q$ norm of $z$, and $\lambda_1, \lambda_2 \geq 0$ are the $L_1$ and $L_2$ regularization parameters, respectively. The term $\lambda_1 \|z\|_1 + \lambda_2 \|z\|_2^2$ in (3) is called the elastic-net regularizer of $z$. For standard regression problems, the loss function $f(z)$ is the mean squared error, the vector $z$ represents the weights (coefficients) on the features, and the set $\mathbb{Z} = \mathbb{R}^p$. 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 $x_0$ and its correct label denoted by $t_0$, let $x$ denote the adversarial example of $x_0$ with a target class $t \neq t_0$. The loss function $f(x)$ for targeted attacks is defined as

$$f(x, t) = \max\{\max_{j \neq t} \text{Logit}(x)_j - \text{Logit}(x)_t, -\kappa\},$$  

(4)

where $\text{Logit}(x) = [\text{Logit}(x)_1, \ldots, \text{Logit}(x)_K] \in \mathbb{R}^K$ is the logit layer (the layer prior to the softmax layer) representation of $x$ 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 $\max_{j \neq t} \text{Logit}(x)_j$ and $\text{Logit}(x)_t$.

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

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

(5)

Consequently, the loss function in (4) aims to render the label $t$ the most probable class for $x$, and the parameter $\kappa$ 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

$$f(x) = \max\{\text{Logit}(x)_t - \max_{j \neq t} \text{Logit}(x)_j, -\kappa\}.$$  

(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 $f(x, t)$ in (4) with $f(x)$ 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 $(x, t)$ with respect to a labeled natural image $(x_0, t_0)$ is as follows:

$$\text{minimize}_{x} c \cdot f(x, t) + \beta \|x - x_0\|_1 + \|x - x_0\|_2^2,$$

subject to $x \in [0, 1]^p$,  

(7)

where $f(x, t)$ 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 $x \in [0, 1]^p$ restricts $x$ 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 $x$ relative to $x_0$ as $\delta = x - x_0$, the EAD formulation in (7) aims to find an adversarial example $x$ that will be classified as the target class $t$ while minimizing the distortion in $\delta$ in terms of the elastic-net loss $\beta \|\delta\|_1 + \|\delta\|_2^2$, which is a linear combination of $L_1$ and $L_2$ distortion metrics between $x$ and $x_0$. 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 $\delta$. However, the $L_1$ penalty is an intuitive regularizer for crafting adversarial examples, as $\|\delta\|_1 = \sum_{i=1}^{p} |\delta_i|$ 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 $x$ in order to remove the box constraint $x \in [0, 1]^p$ (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 $\beta$ (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 \( g(x) = c: f(x) + \|x-x_0\|^2 \) and let \( \nabla g(x) \) be the numerical gradient of \( g(x) \) computed by the DNN. At the \( k+1 \)-th iteration, the adversarial example \( x^{(k+1)} \) of \( x_0 \) is computed by
\[
x^{(k+1)} = S_\beta(x^{(k)}) - \alpha_k \nabla g(x^{(k)}),
\]
where \( \alpha_k \) denotes the step size at the \( k+1 \)-th iteration, and \( S_\beta : \mathbb{R}^p \rightarrow \mathbb{R}^p \) is an element-wise projected shrinkage-thresholding function, which is defined as
\[
[S_\beta(z)]_i = \begin{cases} 
\min\{z_i - \beta, 1\}, & \text{if } z_i - x_{0i} > \beta; \\
x_{0i}, & \text{if } |z_i - x_{0i}| \leq \beta; \\
\max\{z_i + \beta, 0\}, & \text{if } z_i - x_{0i} < -\beta,
\end{cases}
\]
for any \( i \in \{1, \ldots, p\} \). If \( |z_i - x_{0i}| > \beta \), it shrinks the element \( z_i \) by \( \beta \) and projects the resulting element to the feasible box constraint between 0 and 1. On the other hand, if \( |z_i - x_{0i}| \leq \beta \), it thresholds \( z_i \) by setting \( [S_\beta(z)]_i = x_{0i} \).

The proof of optimality of using (8) for solving the EAD formulation in (7) can be given in the supplementary material. Notably, since \( g(x) \) 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 \( \beta \), and keeps a pixel value unchanged if the deviation is less than \( \beta \).

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 \( y^{(k)} \) in Algorithm 1 incorporates the momentum in \( x^{(k)} \) 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 \( x^{(k)} \) is considered as a successful adversarial example of \( x_0 \) if the model predicts its most likely class to be the target class \( t \). The final adversarial example \( x \) is selected from all successful examples based on distortion metrics. In this paper we consider two decision rules for selecting \( x \): the least elastic-net (EN) and \( L_1 \) distortions relative to \( x_0 \). The influence of \( \beta, \kappa \) and the decision rules on EAD will be investigated in the following section.

### 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-\( L_\infty \).

#### 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-\( L_\infty \).

### Algorithm 1 Elastic-Net Attacks to DNNs (EAD)

**Input:** original labeled image \((x_0, t_0)\), target attack class \(t\), attack transferability parameter \(\kappa\), \(L_1\) regularization parameter \(\beta\), step size \(\alpha_k\), # of iterations \(I\)

**Output:** adversarial example \(x\)

Initialization: \(x^{(0)} = y^{(0)} = x_0\)

for \(k = 0\) to \(I - 1\) do
\[
x^{(k+1)} = S_\beta(y^{(k)}) - \alpha_k \nabla g(y^{(k)}))
\]
\[
y^{(k+1)} = x^{(k+1)} + \frac{1}{k+\lambda}(x^{(k+1)} - x^{(k)})
\]
end for

Decision rule: determine \(x\) from successful examples in \(\{x^{(k)}\}_{k=1}^I\) (EN rule or \(L_1\) rule).

---

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

2https://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 $\beta$ leads to less $L_1$-distorted adversarial examples for EAD, whereas the distortion of COV is insensitive to changes in $\beta$.

<table>
<thead>
<tr>
<th>Optimization method</th>
<th>Best case</th>
<th>Average case</th>
<th>Worst case</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\beta$</td>
<td>$L_1$</td>
<td>$L_2$</td>
</tr>
<tr>
<td>COV</td>
<td>0</td>
<td>100</td>
<td>13.93</td>
</tr>
<tr>
<td></td>
<td>$10^{-5}$</td>
<td>100</td>
<td>13.92</td>
</tr>
<tr>
<td></td>
<td>$10^{-4}$</td>
<td>100</td>
<td>13.91</td>
</tr>
<tr>
<td></td>
<td>$10^{-3}$</td>
<td>100</td>
<td>13.8</td>
</tr>
<tr>
<td></td>
<td>$10^{-2}$</td>
<td>100</td>
<td>12.98</td>
</tr>
<tr>
<td>EAD (EN rule)</td>
<td>0</td>
<td>100</td>
<td>14.04</td>
</tr>
<tr>
<td></td>
<td>$10^{-5}$</td>
<td>100</td>
<td>13.66</td>
</tr>
<tr>
<td></td>
<td>$10^{-4}$</td>
<td>100</td>
<td>12.79</td>
</tr>
<tr>
<td></td>
<td>$10^{-3}$</td>
<td>100</td>
<td>9.808</td>
</tr>
<tr>
<td></td>
<td>$10^{-2}$</td>
<td>100</td>
<td>7.271</td>
</tr>
</tbody>
</table>

We implemented FGM and I-FGM using the CleverHans package\(^3\). The best distortion parameter $\epsilon$ is determined by a fine-grained grid search - for each image, the smallest $\epsilon$ in the grid leading to a successful attack is reported. For I-FGM, we perform 10 FGM iterations (the default value) with $\epsilon$-ball clipping. The distortion parameter $\epsilon'$ in each FGM iteration is set to be $\epsilon/10$, which has been shown to be an effective attack setting in (Tramer 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 $L_\infty$ 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 $x \in [0, 1]^p$ by replacing $x$ with $\frac{1 + \text{tanh}(w)}{2}$, where $w \in \mathbb{R}^p$ and $1 \in \mathbb{R}^p$ is a vector of ones. The default ADAM optimizer (Kingma and Ba 2014) is then used to solve $w$ and obtain $x$. We apply this COV approach to (7) and compare with EAD on MNIST with different orders of the $L_1$ regularization parameter $\beta$ 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 $\beta$ leads to less $L_1$-distorted adversarial examples for EAD, whereas the distortion ($L_1$, $L_2$ and $L_\infty$) of COV is insensitive to changes in $\beta$. Similar insensitivity of COV on $\beta$ 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 $\text{tanh}$ 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 $L_\infty$ distortion. The trade-off can be explained by the fact that increasing $\beta$ further encourages sparsity in the perturbation, and hence results in increased $L_\infty$ 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 $\{x^{(k)}\}_{k=1}^\ell$, which we call the elastic-net (EN) decision rule. Alternatively, we can

\(^1\)https://github.com/tensorflow/cleverhans
\(^2\)https://github.com/ysharma1126/EAD-Attack
\(^3\)https://github.com/ysharma1126/EAD-Attack
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 $\beta$ on MNIST. Both decision rules yield 100% ASR for a wide range of $\beta$ values. For the same $\beta$, 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 $L_\infty$ distortions. Similar trends are observed on CIFAR10 (see supplementary material1). The complete results of these two rules on MNIST and CIFAR10 are given in the supplementary material1. 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 $\beta$ value significantly reduces the $L_1$ distortion while having comparable $L_2$ and $L_\infty$ 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-$L_\infty$ attain the least $L_1$, $L_2$, and $L_\infty$ 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 $L_\infty$ distortion metrics.

Notably, despite having large $L_2$ and $L_\infty$ 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 $L_\infty$ 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 $L_\infty$ based examples. Similar to the C&W method and I-FGM, the adversarial examples from EAD are also visually indistinguishable (see supplementary material1).

### 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
We also find that setting \( \kappa \) 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 \( \kappa \). The complete attack transferability 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 \( \kappa \) 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 \( \kappa \), 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 attack 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 \( \kappa \) 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 \( \kappa \). 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 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 labels, leading to 9000 adversarial examples in total for each method. We then separately augment the original training labels, leading to 9000 adversarial examples in total for each method. We then separately augment the original training labels, leading to 9000 adversarial examples in total for each method. We then separately augment the original training labels, leading to 9000 adversarial examples in total for each method. We then separately augment the original training labels, leading to 9000 adversarial examples in total for each method. We then separately augment the original training labels, leading to 9000 adversarial examples in total for each method. We then separately augment the original training labels, leading to 9000 adversarial examples in total for each method. We then separately augment the original training labels, leading to 9000 adversarial examples in total for each method. We then separately augment the original training labels, leading to 9000 adversarial examples in total for each method.

**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 $L_{\infty}$ attacks in breaking undefended and defensively distantly trained 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


