ADMM Attack: An Enhanced Adversarial Attack for Deep Neural Networks with Undetectable Distortions

Pu Zhao[1], Kaidi Xu[1], Sijia Liu[2], Yanzhi Wang[1], Xue Lin[1]
[1] Department of Electrical and Computer Engineering, Northeastern University
[2] IBM Research AI
{zhao.pu, xu.kaid}@husky.neu.edu, Sijia.Liu@ibm.com, {yanz.wang, xue.lin}@northeastern.edu

Abstract— Many recent studies demonstrate that state-of-the-art Deep neural networks (DNNs) might be easily fooled by adversarial examples, generated by adding carefully crafted and visually imperceptible distortions onto original legal inputs through adversarial attacks. Adversarial examples can lead the DNN to misclassify them as any target labels. In the literature, various methods are proposed to minimize the different ℓ_p norms of the distortion. However, there lacks a versatile framework for all types of adversarial attacks. To achieve a better understanding for the security properties of DNNs, we propose a general framework for constructing adversarial examples by leveraging Alternating Direction Method of Multipliers (ADMM) to split the optimization approach for effective minimization of various ℓ_p norms of the distortion, including ℓ_0, ℓ_1, ℓ_2, and ℓ_∞ norms. Thus, the proposed general framework unifies the methods of crafting ℓ_0, ℓ_1, ℓ_2, and ℓ_∞ attacks. The experimental results demonstrate that the proposed ADMM attacks achieve both the high attack success rate and the minimal distortion for the misclassification compared with state-of-the-art attack methods.

I. Introduction

In recent years, deep learning is achieving extraordinary performance [1, 2] and penetrating into wide application domains [3–5], such as natural language processing, computer vision and speech processing. However, despite these success stories, many recent studies demonstrate that even state-of-the-art DNNs might be vulnerable or fooled by adversarial misclassification attacks [6, 7], which find visually imperceptible noises and essentially lead to the DNN misclassification after adding the noise to an original legitimate input image. Kurakin, Goodfellow, and Bengio have demonstrated the existence of adversarial attacks not only in theoretical models but also the physical world [8]. Thus it is essential to evaluate the DNN robustness under adversarial attacks, especially for some security-critical applications.

The robustness of DNNs under adversarial attacks can be evaluated and enhanced from two aspects: 1) adversarial defenses, such as detecting and pre-processing possible adversarial examples [9–11] and/or modifying DNN model structures [12–15] and 2) adversarial attacks, such as the L-BFGS and the C&W attacks [7, 16–19]. The two approaches mutually benefit each other towards hardening DNNs under adversarial attacks. Our work follows the latter approach, but we do appreciate any research efforts on the former approach.

Adversarial examples are constructed by adding negligible distortions onto original legal inputs, and usually the distortions are crafted by formulating and solving an optimization problem such as L-BFGS [7], C&W [16] and EAD [18]. The objectives of the optimization problem have two aspects: (1) misleading the DNN classifier to label the adversarial example as a target class, which is different from the original correct class, and (2) minimizing the ℓ_p norm of the added distortion to keep the noise imperceptible.

In the optimization problem of constructing adversarial examples, we consider and minimize various ℓ_p norm measures (ℓ_0, ℓ_1, ℓ_2 and ℓ_∞ norms) of the distortion, and generates the corresponding ℓ_0, ℓ_1, ℓ_2 and ℓ_∞ attacks. Usually it is not easy to design various ℓ_p norm attacks as one attack is often heavily customed for a certain ℓ_p norm and if switched to design another ℓ_p attack, some vital modifications are required. To solve the optimization problem with different ℓ_p norm minimization requirements, a powerful and efficient method from optimization theory i.e., ADMM (Alternating Direction Method of Multipliers) [20] is introduced, which provides (i) a general framework for various ℓ_p attacks, (ii) no additional sub-optimality besides the original commonly used gradient-based backpropagation method, and (iii) a faster second-order convergence rate than state-of-the-art iterative attacks [21, 22]. By leveraging ADMM, the original optimization problem is split into several correlated subproblems, which can be solved individually. Then their solutions are coordinated to construct a solution to the original problem. This decomposition-coordination procedure of ADMM blends the benefits of dual decomposition and augmented Lagrangian for solving problems with non-convex and combinatorial constraints.

ASPDAC '19, January 21–24, 2019, Tokyo, Japan

ACM ISBN 978-1-4503-6007-4/19/01…$15.00
https://doi.org/10.1145/3287624.3288750

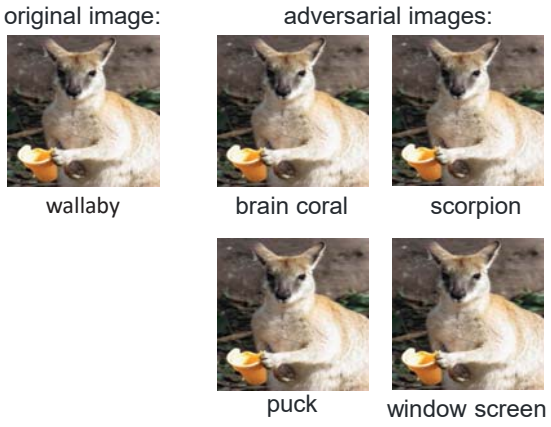

Fig. 1. Adversarial examples generated by our ADMM ℓ_2 attack. The original image is a wallaby from ImageNet dataset. It can be classified to other classes by adding visually undetectable noise.

We show an example of the generated adversarial examples on ImageNet in Fig. 1.

Our contributions beyond what appears in the C&W attack are summarized as follows:

- By leveraging ADMM, we propose a general framework to craft ℓ_0, ℓ_1, ℓ_2, and ℓ_∞ attacks with minor modifications, which is quite different from the C&W attack. The C&W ℓ_2 attack is easy to solve while it is quite hard to minimize the ℓ_0 or ℓ_∞ norm due to their non-differential characteristics. Thus a much more complex method is proposed. For example, the C&W ℓ_0 attack needs to run their ℓ_2 attack iteratively to find and fix the least influencing pixels, thereby identifying a minimal subset of pixels which can be modified in an adversarial example.
- Our attacks achieves high success rate and smaller distortion for the misclassification compared with state-of-the-art attacks.

II. Related Work

The representative attacks and defenses are introduced in this section.

A. Adversarial Attacks

L-BFGS Attack [7] first proposes to minimize the ℓ_2 norm of the perturbation in the objective function to keep the distortion undetectable during adversarial attacks.

JSMA Attack [23] leverages a greedy algorithm to find out the most influential pixels through computing the Jacobian-based Saliency Map and modifies the pixels iteratively, leading to the minimization of the ℓ_0 distortion. However, the prohibitive computational complexity limits its application to large scale datasets like ImageNet.

FGSM [24] and IFGSM [8] Attacks utilize the sign of the gradients of the loss function to determine the direction for modifying the pixels. They are fast ℓ_∞ attacks but lack optimal guarantee. The improvements of FGSM and IFGSM, that is, the fast gradient method (FGM) and the iterative fast gradient method (IFGM), can perform ℓ_1, ℓ_2, and ℓ_∞ attacks.

C&W Attacks [16] can generate ℓ_0, ℓ_2, and ℓ_∞ attacks which is able to achieve 100% attack success rate with much lower distortions compared with the above-mentioned attacks. In particular, the superior performance of the C&W ℓ_2 attack compared with the L-BFGS attack (they are both optimization-based ℓ_2 attacks) is achieved by a more effective objective function.

EAD Attack [18] solves an elastic-net regularized optimization problem during adversarial attacks, that is, minimizing a linear combination of ℓ_1 and ℓ_2 norms in the objective function. It can craft ℓ_1-oriented adversarial attacks and includes the C&W ℓ_2 attack as a special case.

B. Representative Defenses

Defensive Distillation [12] first trains a teacher model to produce soft labels for the training dataset and then trains a distilled model using soft labels as target labels. During the testing phase, a new parameter, *temperature*, is introduced into the softmax layer to produce outputs with almost hard labels.

Adversarial Training [25] enlarges the training dataset by including the adversarial examples with their correct labels and then retrains the neural network, thus enhancing the robustness of DNNs.

III. Formulation: An Optimization Perspective

In this paper, we would like to investigate the problem of designing the perturbation, so that 1) the classification of the image after adding the perturbation can be changed to any target label and 2) the perturbation can be as small as possible under certain distance measure. The optimization problem can be formulated as below,

$$\begin{array}{ll} \underset{\boldsymbol{\delta}}{\text{minimize}} & D(\boldsymbol{\delta}) + g(\mathbf{x}+\boldsymbol{\delta}) \\ \text{subject to} & (\mathbf{x}+\boldsymbol{\delta}) \in [0,1]^n, \end{array} \tag{1}$$

where $D(\boldsymbol{\delta})$ denotes the distance measure of perturbation $\boldsymbol{\delta}$, and $g(\boldsymbol{x})$ has the following form,

$$g(\mathbf{x}) = \begin{cases} 0 & \text{if } \max_{i \neq t}\{Z(\mathbf{x})_i\} - Z(\mathbf{x})_t \leq 0 \\ \infty & \text{otherwise.} \end{cases} \tag{2}$$

$Z(\boldsymbol{x})_i$ represents the i-th element of the logits $Z(\boldsymbol{x})$, which is the output of all layers except the final softmax layer in the DNN model. D function is usually in the form of ℓ_p norm as the following,

$$D(\boldsymbol{\delta}) = \|\boldsymbol{\delta}\|_p = \left(\sum_{i=1}^{n} |\boldsymbol{\delta}_i|^p\right)^{\frac{1}{p}} \tag{3}$$

ℓ_0 norm measures the number of nonzero elements in $\boldsymbol{\delta}$. L_2 norm denotes the standard Euclidean distance of $\boldsymbol{\delta}$.

ℓ_∞ norm represents the maximum absolute value of δ. In the paper, we investigate the problem where D function takes the form of ℓ_0, ℓ_1, ℓ_2 and ℓ_∞.

It is known from (2) that $g(\mathbf{x}) = 0$ iff

$$Z(\mathbf{x})_t \geq Z(\mathbf{x})_i, \quad \text{for all } i, \tag{4}$$

otherwise it would be ∞. So the solution of problem (1) exists only if $g(\mathbf{x}) = 0$, that is, the classification of $\boldsymbol{x}+\boldsymbol{\delta}$ is changed to the target label t. Meanwhile, the perturbation is minimized as the D function is optimized in the problem.

Since the function $g(\boldsymbol{x})$ is non-differential, we use the following form inspired by [16]:

$$g(\boldsymbol{x}+\boldsymbol{\delta}) = c \cdot \max\left(\left(\max_{i\neq t}\left(Z(\boldsymbol{x}+\boldsymbol{\delta})\right) - Z(\boldsymbol{x}+\boldsymbol{\delta})_t\right), 0\right) \tag{5}$$

The original problem can be transformed into as follows,

$$\begin{aligned} \underset{\boldsymbol{\delta}}{\text{minimize}} \quad & D(\boldsymbol{\delta}) + \\ & c \cdot \max\left(\left(\max_{i\neq t}\left(Z(\boldsymbol{x}+\boldsymbol{\delta})\right) - Z(\boldsymbol{x}+\boldsymbol{\delta})_t\right), 0\right) \\ \text{subject to} \quad & (\mathbf{x}+\boldsymbol{\delta}) \in [0,1]^n, \end{aligned} \tag{6}$$

where $c \geq 0$ is a regularization parameter. Note that when $c = 0$, the solution attempts to minimize $D(\boldsymbol{\delta})$ only. In the other extreme case of $c \to \infty$, the solution moves towards the direction that $\boldsymbol{x}+\boldsymbol{\delta}$ must be an adversarial example.

IV. ADMM formulation

To apply ADMM, we introduce *two* auxiliary variables $\mathbf{z}$ and $\mathbf{w}$, so that problem (6) can be rewritten as,

$$\begin{aligned} \underset{\boldsymbol{\delta},\mathbf{z},\mathbf{w}}{\text{minimize}} \quad & D(\boldsymbol{\delta}) + g(\mathbf{x}+\mathbf{z}) + h(\mathbf{w}) \\ \text{subject to} \quad & \mathbf{z} = \boldsymbol{\delta} \\ & \mathbf{w} = \mathbf{x}+\mathbf{z}, \end{aligned} \tag{7}$$

where $h(\mathbf{w})$ is the indicator function,

$$h(\mathbf{w}) = \begin{cases} 0 & \mathbf{w} \in [0,1]^n \\ \infty & \text{otherwise.} \end{cases} \tag{8}$$

The augmented Lagrangian of problem (7) is given by

$$\begin{aligned} L(\boldsymbol{\delta},\mathbf{z},\mathbf{w},\mathbf{u},\mathbf{v}) = & D(\boldsymbol{\delta}) + g(\mathbf{x}+\mathbf{z}) + h(\mathbf{w}) \\ & + \mathbf{u}^T(\boldsymbol{\delta}-\mathbf{z}) + \mathbf{v}^T(\mathbf{w}-\mathbf{z}-\mathbf{x}) \\ & + \frac{\rho}{2}\|\boldsymbol{\delta}-\mathbf{z}\|_2^2 + \frac{\rho}{2}\|\mathbf{w}-\mathbf{z}-\mathbf{x}\|_2^2, \end{aligned} \tag{9}$$

where $\mathbf{u}$ and $\mathbf{v}$ are Lagrangian multipliers.

ADMM yields the following alternating steps

$$\{\boldsymbol{\delta}^{k+1}, \mathbf{w}^{k+1}\} = \arg\min L(\boldsymbol{\delta}, \mathbf{z}^k, \mathbf{w}, \mathbf{u}^k, \mathbf{v}^k) \tag{10}$$

$$\mathbf{z}^{k+1} = \arg\min L(\boldsymbol{\delta}^{k+1}, \mathbf{z}, \mathbf{w}^{k+1}, \mathbf{u}^k, \mathbf{v}^k) \tag{11}$$

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \rho(\boldsymbol{\delta}^{k+1} - \mathbf{z}^{k+1}) \tag{12}$$

$$\mathbf{v}^{k+1} = \mathbf{v}^k + \rho(\mathbf{w}^{k+1} - \mathbf{x}^{k+1} - \mathbf{z}^{k+1}). \tag{13}$$

In problem (10), the optimal $\boldsymbol{\delta}^{k+1}$ and $\boldsymbol{w}^{k+1}$ are obtained by minimizing the L function with fixed $\boldsymbol{z}^k$, $\boldsymbol{u}^k$ and $\boldsymbol{v}^k$. Similarly, the optimal $\boldsymbol{z}^{k+1}$ is achieved by solving problem (11) with fixed $\boldsymbol{\delta}^{k+1}$, $\boldsymbol{w}^{k+1}$, $\boldsymbol{u}^k$ and $\boldsymbol{v}^k$. In Eq. (12) and (13), we update $\boldsymbol{u}^k$ and $\boldsymbol{v}^k$ with $\boldsymbol{u}^{k+1}$ and $\boldsymbol{v}^{k+1}$, respectively.

The major advantage of the formulation (7) lies in the step (10). We note that the minimization over $\boldsymbol{\delta}$ and $\mathbf{w}$ can be split into two problems, each of which has the *closed-form* solution. Specifically, problem (10) can be equivalently transformed into problem (14) and (15),

$$\underset{\boldsymbol{\delta}}{\text{minimize}} \;\; D(\boldsymbol{\delta}) + \frac{\rho}{2}\|\boldsymbol{\delta} - \mathbf{z}^k + (1/\rho)\mathbf{u}^k\|_2^2 \tag{14}$$

and

$$\underset{\mathbf{w}}{\text{minimize}} \;\; h(\mathbf{w}) + \frac{\rho}{2}\|\mathbf{w} - \mathbf{z}^k - \mathbf{x} + (1/\rho)\mathbf{v}^k\|_2^2. \tag{15}$$

In problem (11), we need to solve the following problem,

$$\begin{aligned} \underset{\mathbf{z}}{\text{minimize}} \quad & g(\mathbf{x}+\mathbf{z}) + \tfrac{\rho}{2}\|\boldsymbol{\delta}^{k+1} - \mathbf{z} + (1/\rho)\mathbf{u}^k\|_2^2 \\ & + \tfrac{\rho}{2}\|\mathbf{w}^{k+1} - \mathbf{z} - \mathbf{x} + (1/\rho)\mathbf{v}^k\|_2^2, \end{aligned} \tag{16}$$

As specified above, the original problem is split into three subproblems, (14), (15) and (16), through ADMM. Note that in each subproblem, we only need to deal with one constraint, which is much easier to solve compared with the original problem dealing with multiple constraints at the same time.

V. ADMM solution

A. δ-minimization: a universal solution framework

In what follows, we investigate how to obtain the solution to problem (14) through proximal operator. We first introduce the proximal operator [26] defined as,

$$\mathbf{prox}_{\lambda D}(\boldsymbol{s}) = \arg\min_{\boldsymbol{\delta}}\left(\lambda D(\boldsymbol{\delta}) + \frac{1}{2}\|\boldsymbol{\delta} - \boldsymbol{s}\|_2^2\right) \tag{17}$$

where $\mathbf{prox}_{\lambda D}(\mathbf{s})$ means the optimal solution $\boldsymbol{\delta}^*$ which can achieve the minimal value of $\lambda D(\boldsymbol{\delta}) + \frac{1}{2}\|\boldsymbol{\delta} - \boldsymbol{s}\|_2^2$ given $\boldsymbol{s}$.

In problem (14), multiple choices of $D(\boldsymbol{\delta})$ are taken into consideration: 1) $D(\boldsymbol{\delta}) = \|\boldsymbol{\delta}\|_0$, 2) $D(\boldsymbol{\delta}) = \|\boldsymbol{\delta}\|_1$, 3) $D(\boldsymbol{\delta}) = \|\boldsymbol{\delta}\|_2$, and 4) $D(\boldsymbol{\delta}) = \|\boldsymbol{\delta}\|_\infty$. Each choice of $D(\boldsymbol{\delta})$ yields an analytic solution of problem (14) by evaluating the corresponding proximal operator with $\boldsymbol{s} = \mathbf{z}^k - 1/\rho\mathbf{u}^k$ and $\lambda = 1/\rho$ as specified in the following.

A.1 ℓ_0 attack

For the case of ℓ_0 norm ($D(\boldsymbol{\delta}) = \|\boldsymbol{\delta}\|_0$), the proximal operator is,

$$\mathbf{prox}_{\lambda 0}(\mathbf{s}) = \arg\min_{\delta}\left(\lambda\|\boldsymbol{\delta}\|_0 + \frac{1}{2}\|\boldsymbol{\delta} - \boldsymbol{s}\|_2^2\right) \tag{18}$$

The solution can be obtained elementwise,

$$(\mathbf{prox}_{\lambda 0}(\mathbf{s}))_i = \begin{cases} 0 & |s_i| < \sqrt{2\lambda} \\ 0 \text{ or } s_i & |s_i| = \sqrt{2\lambda} \\ s_i & |s_i| > \sqrt{2\lambda} \end{cases} \tag{19}$$

where $\boldsymbol{s} = \mathbf{z}^k - 1/\rho \mathbf{u}^k$ and $\lambda = 1/\rho$.

A.2 ℓ_1 attack

For the case of L_1 norm, that is, $D(\boldsymbol{\delta}) = \|\boldsymbol{\delta}\|_1$, the proximal operator is,

$$\mathbf{prox}_{\lambda 1}(\mathbf{s}) = \arg\min_{\boldsymbol{\delta}} \left(\lambda \|\boldsymbol{\delta}\|_1 + \frac{1}{2} \|\boldsymbol{\delta} - \boldsymbol{s}\|_2^2 \right) \tag{20}$$

By performing the (elementwise) soft thresholding operator, we can get the solution,

$$(\mathbf{prox}_{\lambda 1}(\mathbf{s}))_i = \begin{cases} s_i - \lambda & s_i \geq \lambda \\ 0 & |s_i| < \lambda \\ s_i + \lambda & s_i \leq -\lambda \end{cases} \tag{21}$$

where $\boldsymbol{s} = \mathbf{z}^k - 1/\rho \mathbf{u}^k$ and $\lambda = 1/\rho$.

A.3 ℓ_2 attack

For the case of L_2 norm, that is, $D(\boldsymbol{\delta}) = \|\boldsymbol{\delta}\|_2$, the proximal operator is,

$$\mathbf{prox}_{\lambda 2}(\mathbf{s}) = \arg\min_{\boldsymbol{\delta}} \left(\lambda \|\boldsymbol{\delta}\|_2 + \frac{1}{2} \|\boldsymbol{\delta} - \boldsymbol{s}\|_2^2 \right) \tag{22}$$

By the 'block soft thresholding' operator [26], we can get the solution,

$$\mathbf{prox}_{\lambda 2}(\mathbf{s}) = \begin{cases} (1 - \lambda/\|\mathbf{s}\|_2)\mathbf{s} & \|\mathbf{s}\|_2 \geq \lambda \\ 0 & \|\mathbf{s}\|_2 < \lambda \end{cases} \tag{23}$$

where $\boldsymbol{s} = \mathbf{z}^k - 1/\rho \mathbf{u}^k$ and $\lambda = 1/\rho$.

A.4 ℓ_∞ attack

For the ℓ_∞ norm, problem (14) becomes

$$\underset{\boldsymbol{\delta}}{\text{minimize}} \ \|\boldsymbol{\delta}\|_\infty + \frac{\rho}{2} \|\boldsymbol{\delta} - \mathbf{s}\|_2^2, \tag{24}$$

where $\mathbf{s} = \mathbf{z}^k - (1/\rho)\mathbf{u}^k$.

By introducing epigraph variable r, then problem (24) becomes

$$\begin{array}{ll} \underset{\boldsymbol{\delta}, r}{\text{minimize}} & r + \frac{\rho}{2} \|\boldsymbol{\delta} - \mathbf{s}\|_2^2 \\ \text{subject to} & \delta_i \leq r, \ i = 1, 2, \ldots, n, \end{array} \tag{25}$$

where the solution is given by the KKT conditions. The Lagrangian is

$$L(\boldsymbol{\delta}, t, \boldsymbol{\mu}) = t + (\rho/2) \|\boldsymbol{\delta} - \boldsymbol{s}\|_2^2 + \boldsymbol{\mu}^T (\boldsymbol{\delta} - t\mathbf{1}) \tag{26}$$

where $\boldsymbol{\mu}$ is the dual variable, and the optimality conditions are

$$\begin{aligned} \boldsymbol{\delta}_i^* \leq t^*, \boldsymbol{\mu}_i^* \geq 0, \boldsymbol{\mu}_i^* (\boldsymbol{\delta}_i^* - t^*) = 0, \\ \rho(\boldsymbol{\delta}_i^* - \boldsymbol{s}_i) + \boldsymbol{\mu}_i^* = 0, \mathbf{1}^T \boldsymbol{\mu}^* = 1 \end{aligned} \tag{27}$$

If $\boldsymbol{\delta}_i^* < t^*$, then the third condition implies that $\boldsymbol{\mu}_i^* = 0$, and if $\boldsymbol{\delta}_i^* = t^*$, the fourth condition implies that $\boldsymbol{\mu}_i^* = \rho(\boldsymbol{s}_i - t^*)$. Since $\boldsymbol{\mu}_i^* \geq 0$, we have

$$\boldsymbol{\mu}_i^* = \rho(\boldsymbol{s}_i - t^*)_+ \tag{28}$$

Substituting for $\boldsymbol{\mu}_i^*$ in the fifth condition gives

$$\sum_{i=1}^{n} \rho(\boldsymbol{s}_i - t^*)_+ = 1 \tag{29}$$

This equation can be solved for t^* by bisection using the initial interval $[\min_i s_i - (1/n), \max_i s_i]$. After solving (29) and obtaining t^*, we recover the solution to the original problem (24) via the following,

$$\boldsymbol{\delta}_i^* = \min\{t^*, \boldsymbol{s}_i\} \tag{30}$$

This follows by applying the third and fourth conditions.

B. w-minimization step

We solve problem (15) in this section. Based on the definition of the indicator function h, problem (15) becomes a projection problem onto a box constraint,

$$\underset{\mathbf{w}}{\text{minimize}} \ \frac{\rho}{2} \|\mathbf{w} - \mathbf{z}^k - \mathbf{x} + (1/\rho)\mathbf{v}^k\|_2^2 \tag{31}$$

$$\text{subject to} \ \ \mathbf{w} \in [0, 1]^n. \tag{32}$$

The solution is given by

$$[\mathbf{w}^{k+1}]_i = \begin{cases} 0 & \text{if } [\mathbf{z}^k + \mathbf{x} - (1/\rho)\mathbf{v}^k]_i < 0 \\ 1 & \text{if } [\mathbf{z}^k + \mathbf{x} - (1/\rho)\mathbf{v}^k]_i > 1 \\ [\mathbf{z}^k + \mathbf{x} - (1/\rho)\mathbf{v}^k]_i & \text{otherwise,} \end{cases} \tag{33}$$

where $[\mathbf{w}]_i$ denotes the i-th entry of $\mathbf{w}$.

C. z-minimization step

In this section, we solve problem (16). We note that the g function in problem (16) depends on the DNN model and is usually non-convex, thus it's almost impossible to obtain its closed-form solution. But we can try to find its solution through gradient descent method. Stochastic gradient descent methods have been widely applied in deep learning for non-convex optimization. There are several gradient descent optimizers including standard gradient descent, gradient descent with momentum [27], and Adam [28]. We use Adam optimizer to solve problem (16) as Adam converges more quickly.

TABLE I
ADVERSARIAL ATTACK SUCCESS RATE (ASR) AND DISTORTION OF DIFFERENT L_2 ATTACKS FOR DIFFERENT DATASETS

Data Set	Attack Method	Best Case				Average Case				Worst Case			
		ASR	$\boldsymbol{L_2}$	L_1	L_∞	ASR	$\boldsymbol{L_2}$	L_1	L_∞	ASR	$\boldsymbol{L_2}$	L_1	L_∞
MNIST	FGM(L_2)	99.3	2.158	23.7	0.562	43.2	3.18	37.6	0.761	0	N.A.	N.A.	N.A.
	IFGM(L_2)	100	1.61	18.2	0.393	99.7	2.43	31.8	0.574	99.3	3.856	54.1	0.742
	C&W(L_2)	100	1.356	13.32	0.394	100	1.9	21.11	0.533	99.6	2.52	30.44	0.673
	ADMM(L_2)	100	1.268	15.93	0.398	100	1.779	25.06	0.444	99.9	2.269	34.7	0.561
CIFAR-10	FGM(L_2)	99.7	0.418	13.85	0.05	40.6	1.09	37.4	0.62	1.2	4.17	119.3	0.43
	IFGM(L_2)	100	0.185	6.26	0.021	100	0.419	14.9	0.043	100	0.685	22.8	0.0674
	C&W(L_2)	100	0.170	5.721	0.0189	100	0.322	11.28	0.0347	100	0.445	15.79	0.0495
	ADMM(L_2)	100	0.163	5.66	0.0192	100	0.315	10.97	0.0354	100	0.427	15.05	0.0502
ImageNet	FGM(L_2)	15	2.37	815	0.129	3	7.51	2104	0.25	0	N.A.	N.A.	N.A.
	IFGM(L_2)	100	0.984	328	0.031	100	2.38	795	0.079	97.6	4.59	1354	0.177
	C&W(L_2)	100	0.449	126.8	0.0159	100	0.621	198	0.0218	100	0.81	272.3	0.031
	ADMM(L_2)	100	0.412	112.5	0.017	100	0.555	166.7	0.021	100	0.704	225.6	0.0356

VI. PERFORMANCE EVALUATION

We demonstrate the experimental results of the proposed ADMM attacks compared with state-of-the-art attacks, including C&W attacks [16], EAD attack [18], FGM and IFGM attacks [8], on three image classification datasets, MNIST [29], CIFAR-10 [30] and ImageNet [1].

A. Experiment Setup and Parameter Setting

Based on C&W attack setup[1], we train two networks for MNIST and CIFAR-10 datasets, respectively, and utilize a pre-trained network, Inception-v3 [31], for ImageNet.

For targeted attacks, we show the results of different methods to choose the target labels: 1) the average case randomly selects the target label from all the labels except the correct label, 2) the best case performs attacks using all incorrect labels, and report the target label that is the easiest to attack, and 3) the worst case performs attacks using all incorrect labels, and report the label that is the most difficult to attack.

The network architecture for MNIST and CIFAR-10 is the same with four convolutional layers, two max pooling layers, two fully connected layers and a softmax layer. It is able to achieve 99.5% accuracy on MNIST and 80% accuracy on CIFAR-10. On ImageNet, the Google Inception model can achieve 96% top-5 accuracy with input images of size $299 \times 299 \times 3$. We conduct all experiments on machines with NVIDIA GTX 1080 TI GPUs.

The implementations of FGM and IFGM are based on the CleverHans package [32]. The key distortion parameter ϵ is determined through a fine-grained grid search. For IFGM, there are 10 FGM iterations and the distortion parameter in each FGM iteration is set to $\epsilon' = \epsilon/10$, as demonstrated in [25] for its effectiveness.

The implementations of C&W attacks and EAD attack[2] are based on the GitHub code released by the authors. In the EAD attack, we use the ℓ_1 distortion measurement (ℓ_1) to select the final adversarial examples since it can usually obtain lower ℓ_1 distortion than the least elastic-net (EN) measurement. The key parameter β is set to 0.001.

[1] https://github.com/carlini/nn_robust_attacks
[2] https://github.com/ysharma1126/EAD-Attack

TABLE II
ADVERSARIAL ATTACK SUCCESS RATE AND DISTORTION OF ADMM AND C&W L_0 ATTACKS FOR MNIST AND CIFAR-10

Dataset	Attack method	Best case		Average case		Worst case	
		ASR	L_0	ASR	L_0	ASR	L_0
MNIST	C&W(L_0)	100	7.88	100	16.58	100	29.84
	ADMM(L_0)	100	6.94	100	13.35	100	23.66
CIFAR	C&W(L_0)	100	8.16	100	20.82	100	35.07
	ADMM(L_0)	100	7.64	100	18.78	100	32.81

B. Attack Success Rate and Distortion for ADMM ℓ_2 attack

The attack success rate (ASR) is the percentage of the adversarial examples which are successfully misclassified to target labels by the DNN model. We report the average distortion of all successful adversarial examples and the distortion for zero ASR is not available (N.A.).

We perform adversarial attacks on MNIST, CIFAR-10 and ImageNet. For MNIST and CIFAR-10, 1000 correctly classified images are randomly selected from the test sets with 9 target labels for each image, so we craft 9000 adversarial examples for MNIST or CIFAR-10 using each attack method. For ImageNet, we randomly select 100 correctly classified images with 9 random target labels for each image.

We compare the ADMM ℓ_2 attack with FGM, IFGM and C&W ℓ_2 attacks. Table I shows the results on MNIST, CIFAR-10 and ImageNet. As we can see, FGM fails to generate adversarial examples with high success rate as it does not have any success guarantee. Among IFGM, C&W and ADMM ℓ_2 attacks, ADMM achieves the lowest

TABLE III
ADVERSARIAL ATTACK SUCCESS RATE (ASR) AND DISTORTION OF DIFFERENT L_1 ATTACKS FOR DIFFERENT DATASETS

Data Set	Methods	Best Case		Average Case		Worst Case	
		ASR	L_1	ASR	L_1	ASR	L_1
MNIST	IFGM(L_1)	100	17.3	100	34.6	100	58.4
	EAD(L_1)	100	7.74	100	14.16	100	21.38
	ADMM(L_1)	100	6.29	100	12.35	100	17.9
CIFAR-10	IFGM(L_1)	100	5.96	100	15.8	100	20.8
	EAD(L_1)	100	1.94	100	4.62	100	7.25
	ADMM(L_1)	100	1.75	100	3.750	100	5.92
ImageNet	IFGM(L_1)	100	298	100	580	100	685
	EAD(L_1)	100	60.98	100	112.7	100	185
	ADMM(L_1)	100	49.17	100	75.2	100	127

ℓ_2 distortion for the best case, average case and worst case. IFGM has larger ℓ_2 distortions compared with C&W and ADMM attacks on the three datasets, especially on ImageNet. For the worst case, the ADMM attack can reduce the ℓ_2 distortion by about 10% compared with C&W ℓ_2 attack on MNIST and 12.5% on ImageNet.

C. ASR and Distortion for ADMM ℓ_0 attack

We demonstrate the performance of ADMM ℓ_0 attack in terms of attack success rate and ℓ_0 norm distortion in this section. The ADMM ℓ_0 attack is compared with C&W ℓ_0 attack on MNIST and CIFAR-10. 500 images are randomly selected from the test sets of MNIST and CIFAR-10, respectively, each with 9 target labels.

As observed from Table II, both C&W and ADMM ℓ_0 attacks can achieve 100% attack success rate and ADMM ℓ_0 attack can achieve lower ℓ_0 distortion than the C&W ℓ_0 attack. For the worst case, the ADMM attack can reduce the ℓ_0 distortion by about 20% on MNIST compared with the C&W ℓ_0 attack.

D. ASR and Distortion for ADMM ℓ_1 attack

We compare the ADMM ℓ_1 attack with IFGM and EAD ℓ_1 [18] attacks. We report The ASR and the average distortion of all successful adversarial examples.

The results of the ℓ_1 attack are shown in Table III. We can observe that the IFGM, EAD and ADMM ℓ_1 attacks can achieve 100% attack success rate. ADMM ℓ_1 attack can achieve the lowest ℓ_1 distortion. As demonstrated in Table III, in the best case, the ADMM ℓ_1 attack can craft adversarial examples with a ℓ_1 norm about 19% smaller than that of the EAD ℓ_1 attack on MNIST and ImageNet. For the worst case, the ℓ_1 norm of ADMM ℓ_1 attack is about 33% lower on CIFAR-10 or ImageNet compared with the EAD ℓ_1 attack.

E. ASR and Distortion for ADMM ℓ_∞ attack

We compare the ADMM ℓ_∞ attack with the IFGM ℓ_∞ attack and report the ASR and the average distortion of all successful adversarial examples.

TABLE IV
ADVERSARIAL ATTACK SUCCESS RATE (ASR) AND DISTORTION OF DIFFERENT L_∞ ATTACKS FOR DIFFERENT DATASETS

Data Set	Methods	Best Case		Average Case		Worst Case	
		ASR	L_∞	ASR	L_∞	ASR	L_∞
MNIST	IFGM(L_∞)	100	0.1535	100	0.234	100	0.367
	ADMM(L_∞)	100	0.1439	100	0.191	100	0.234
CIFAR-10	IFGM(L_∞)	100	0.00655	100	0.0149	100	0.0262
	ADMM(L_∞)	100	0.00548	100	0.011	100	0.0161
ImageNet	IFGM(L_∞)	100	0.0035	100	0.01	100	0.0152
	ADMM(L_∞)	100	0.00268	100	0.00466	100	0.0065

The results of the ADMM ℓ_∞ attack are demonstrated in Table IV. We can observe that both IFGM and ADMM ℓ_∞ attacks can achieve 100% attack success rate. ADMM ℓ_∞ attack can achieve lower ℓ_∞ norm compared with the IFGM ℓ_∞ attack. In the worst case, the improvement of the ADMM ℓ_∞ attack over the IFGM ℓ_∞ attack is much more obvious. The ℓ_∞ distortion measure of the ADMM attack is about 40% smaller than the IFGM attack on MNIST or CIFAR-10 dataset for the worst case. On ImageNet, the ℓ_∞ norm of ADMM attack is 58% lower than that of IFGM attack.

VII. Conclusion

In this paper, we propose the ADMM attack for DNNs with undetectable distortions. Under the general ADMM attack framework, ℓ_0, ℓ_1, ℓ_2 and ℓ_∞ attacks are proposed and implemented to minimize various ℓ_p norms. We compare the ADMM attacks with state-of-the-art adversarial attacks, demonstrating the effectiveness of the ADMM attacks.

Acknowledgements

This work is partly supported by the National Science Foundation (CCF-1733701, CNS-1704662, and CNS-1739748), Air Force Research Laboratory FA8750-18-2-0058, and U.S. Office of Naval Research.

References

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 248–255, IEEE, 2009.

[2] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1701–1708, 2014.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[4] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath,

et al., "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[5] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[6] I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *2015 ICLR*, vol. arXiv preprint arXiv:1412.6572, 2015.

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

[8] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," *arXiv preprint arXiv:1607.02533*, 2016.

[9] C. Guo, M. Rana, M. Cissé, and L. van der Maaten, "Countering adversarial images using input transformations," *arXiv preprint arXiv:1711.00117*, 2017.

[10] G. K. Dziugaite, Z. Ghahramani, and D. M. Roy, "A study of the effect of jpg compression on adversarial images," *arXiv preprint arXiv:1608.00853*, 2016.

[11] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. Yuille, "Mitigating adversarial effects through randomization," *arXiv preprint arXiv:1711.01991*, 2017.

[12] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *Security and Privacy (SP), 2016 IEEE Symposium on*, pp. 582–597, IEEE, 2016.

[13] G. S. Dhillon, K. Azizzadenesheli, Z. C. Lipton, J. Bernstein, J. Kossaifi, A. Khanna, and A. Anandkumar, "Stochastic activation pruning for robust adversarial defense," *arXiv preprint arXiv:1803.01442*, 2018.

[14] S. Wang, X. Wang, P. Zhao, W. Wen, D. Kaeli, P. Chin, and X. Lin, "Defensive Dropout for Hardening Deep Neural Networks under Adversarial Attacks," *ArXiv e-prints*, Sept. 2018.

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

[16] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Security and Privacy (SP), 2017 IEEE Symposium on*, pp. 39–57, IEEE, 2017.

[17] P. Zhao, S. Liu, Y. Wang, and X. Lin, "An admm-based universal framework for adversarial attacks on deep neural networks," *CoRR*, vol. abs/1804.03193, 2018.

[18] P.-Y. Chen, Y. Sharma, H. Zhang, J. Yi, and C.-J. Hsieh, "Ead: elastic-net attacks to deep neural networks via adversarial examples," *arXiv preprint arXiv:1709.04114*, 2017.

[19] A. Athalye, N. Carlini, and D. Wagner, "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples," *arXiv preprint arXiv:1802.00420*, 2018.

[20] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[21] M. Hong and Z.-Q. Luo, "On the linear convergence of the alternating direction method of multipliers," *Mathematical Programming*, vol. 162, pp. 165–199, Mar 2017.

[22] H. Wang and A. Banerjee, "Bregman alternating direction method of multipliers," in *Advances in Neural Information Processing Systems 27* (Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, eds.), pp. 2816–2824, Curran Associates, Inc., 2014.

[23] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*, pp. 372–387, IEEE, 2016.

[24] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[25] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "Ensemble adversarial training: Attacks and defenses," *2018 ICLR*, vol. arXiv preprint arXiv:1705.07204, 2018.

[26] N. Parikh, S. Boyd, *et al.*, "Proximal algorithms," *Foundations and Trends® in Optimization*, vol. 1, no. 3, pp. 127–239, 2014.

[27] N. Qian, "On the momentum term in gradient descent learning algorithms," *Neural Networks*, vol. 12, no. 1, pp. 145 – 151, 1999.

[28] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *2015 ICLR*, vol. arXiv preprint arXiv:1412.6980, 2015.

[29] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, pp. 2278–2324, Nov 1998.

[30] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

[31] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2818–2826, 2016.

[32] N. Papernot, I. Goodfellow, R. Sheatsley, R. Feinman, and P. McDaniel, "cleverhans v1.0.0: an adversarial machine learning library," *arXiv preprint arXiv:1610.00768*, 2016.