Reinforced Adversarial Attacks on Deep Neural Networks Using ADMM

Pu Zhao[1], Kaidi Xu[1], Tianyun Zhang[2], Makan Fardad[2], Yanzhi Wang[1], Xue Lin[1]

[1] Department of Electrical and Computer Engineering, Northeastern University
[2] College of Engineering & Computer Science, Syracuse University
zhao.pu@husky.neu.edu xu.kaid@husky.neu.edu tzhan120@syr.edu
makan@syr.edu yanz.wang@northeastern.edu xue.lin@northeastern.edu

***Abstract*—As deep learning penetrates into wide application domains, it is essential to evaluate the robustness of deep neural networks (DNNs) under adversarial attacks, especially for some security-critical applications. To better understand the security properties of DNNs, we propose a general framework for constructing adversarial examples, based on ADMM (Alternating Direction Method of Multipliers). This general framework can be adapted to implement L2 and L0 attacks with minor changes. Our ADMM attacks require less distortion for incorrect classification compared with C&W attacks. Our ADMM attack is also able to break defenses such as defensive distillation and adversarial training, and provide strong attack transferability.**

***Index Terms*—Deep Neural Networks, Adversarial Attacks, ADMM (Alternating Direction Method of Multipliers)**

I. Introduction

As deep learning is achieving extraordinary performance [1, 2] and penetrating into wide application domains [3, 4, 5], it is essential to evaluate the robustness of deep neural networks (DNNs) under adversarial attacks, especially for some security-critical applications. Evidence has shown that audio/visual inputs sound/look like speech/objects to machine learning algorithms but non-sense to humans [6, 7]. Recently Kurakin, Goodfellow, and Bengio have demonstrated the existence of adversarial attacks not only in theoretical models but also the physical world [8].

To enhance the robustness of DNNs under adversarial attacks, there are in general two different research approaches: (i) defend by pre-processing possibly adversarially distorted inputs [9, 10, 11] and/or by modifying DNN model structures [12, 13, 14, 15] and (ii) evaluate the robustness by constructing adversarial examples with negligible distortions added onto original legal inputs [16, 17, 18, 19, 20]. The two approaches mutually benefit each other towards hardening DNNs. Although our work follows the latter approach, we do appreciate any research efforts using the former approach.

Adversarial examples are constructed by adding negligible distortions onto original legal inputs as shown in Fig. 1, and usually an optimization problem is formulated for that such as L-BFGS [16], JSMA [21], Deepfool [22], C&W [17], and EAD [19]. Currently C&W is the most powerful attack [17, 20] in that it achieves 100% attack success rate with the minimum distortion compared with other attacks and it defeats many state-of-the-art defenses.

This work considers two measures (L_2 and L_0 norms) of the distortion, namely, L_2 attack and L_0 attack. To solve the optimization problem of constructing adversarial examples, we introduce a powerful and efficient method from optimization theory i.e., ADMM (Alternating Direction Method of Multipliers) [23], which provides (i) a general framework for L_2 and L_0 attacks, (ii) no additional sub-optimality besides the original gradient-based backpropagation method commonly used in DNNs, and (iii) a faster second-order convergence rate than state-of-the-art iterative attacks [24, 25]. ADMM decomposes an original optimization problem into two correlated subproblems, which can be solved individually, and then coordinates solutions to the subproblems 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. Our contributions beyond what appears in C&W are summarized as follows:

- Thanks to ADMM, our L_2 and L_0 attacks are constructed through a general framework, while C&W L_0 attack needs to run their L_2 attack iteratively to find the pixels with the least effect and fix them, thereby identifying a minimal subset of pixels for modification to generate an adversarial example.
- Our attacks require less distortion for incorrect classification compared with C&W attacks, which themselves require less distortion compared to other methods in the literature.

Besides comparing with C&W and other attacks, we also test our L_2 and L_0 attacks against defenses such as defensive distillation [12] and adversarial training [26], demonstrating the success of our attacks. In addition, we validate the transferability of our attacks onto different defenses.

II. Background

A. Notation

This work focuses on DNNs for image classification tasks. A gray-scale image with height h and width w is represented by a two dimensional vector $\boldsymbol{x} \in \mathrm{R}^{hw}$. Each element x_i represents the value of the i-th pixel and is scaled to the range $[0, 1]$. A colored RGB image with three channels is represented by a three dimensional tensor $\boldsymbol{x} \in \mathrm{R}^{3hw}$. We use model $F(\boldsymbol{x}) = \boldsymbol{y}$ to denote a neural network, where F accepts an input $\boldsymbol{x}$ and generates an output $\boldsymbol{y}$. We use trained neural networks in this work so the model F is fixed.

Suppose the neural network is an m-class classifier and the output layer performs softmax operation. Let $Z(\boldsymbol{x})$ denote the output of all layers except for the softmax layer, and we have $F(\boldsymbol{x}) = \mathrm{softmax}(Z(\boldsymbol{x})) = \boldsymbol{y}$. The input to the softmax layer, $Z(\boldsymbol{x})$, is called logits. The element y_i of the output vector $\boldsymbol{y}$ represents the probability that input $\boldsymbol{x}$ belongs to the i-th class.

Figure 1. Adversarial examples generated by our ADMM $L2$ attack. The original image is a koala from ImageNet dataset (the leftmost). The nine constructed adversarial examples are mis-classified as the target labels above the images

The output vector $\boldsymbol{y}$ is treated as a probability distribution and its elements satisfy $0 \leq y_i \leq 1$ and $y_1+y_2+\cdots+y_m = 1$. The neural network classifies input $\boldsymbol{x}$ according to the maximum probability i.e., $C(\boldsymbol{x}) = \arg\max_i y_i$.

B. Assumptions of Adversarial Attacks

When constructing adversarial examples, we assume that the neural network is completely accessible, which means we are able to use the architecture and all parameters in a white-box manner. The adversarial attack can be either targeted or untargeted. Give an original legal input $\boldsymbol{x}$ with its correct label t^* and a target label $t \neq t^*$, the targeted adversarial attack is to find an input $\boldsymbol{x}'$ such that $C(\boldsymbol{x}') = t$ and $\boldsymbol{x}$ and $\boldsymbol{x}'$ are close according to some distance measure between $\boldsymbol{x}$ and $\boldsymbol{x}'$. The input $\boldsymbol{x}'$ is then called as an adversarial example. For targeted attacks, we employ different ways to choose the target labels:

- *Average Case*: select at random the target label uniformly among all the labels that are not the correct label.
- *Best Case*: perform attacks using all incorrect labels, and report the target label that is the least difficult to attack.
- *Worst Case*: perform attacks using all incorrect labels, and report the label that is the most difficult to attack.

C. Measures of the Distortion

We need to measure the distortion between the original legal input $\boldsymbol{x}$ and the adversarial example $\boldsymbol{x}'$. L_p norms are the most commonly used measures in the literature. When generating adversarial examples, we use L_2 and L_0 norms, respectively, that is, in this work, using a general ADMM-based framework, we implement L_2 attack and L_0 attack, respectively, as they are representative attacks.

III. ADMM-based Framework for Adversarial Attacks

This section describes our ADMM-based framework for constructing adversarial examples. We begin by formally defining the *initial* problem of constructing adversarial examples as: **Given** an original legal input image $\boldsymbol{x}$, **find** an adversarial example $\boldsymbol{x}+\boldsymbol{\delta}$, **such that** $D(\boldsymbol{\delta})$ is minimized, $C(\boldsymbol{x}+\boldsymbol{\delta}) = t$, and $\boldsymbol{x}+\boldsymbol{\delta} \in [0,1]^n$. $\boldsymbol{\delta}$ is the distortion on $\boldsymbol{x}$. $\mathcal{D}(\boldsymbol{\delta})$ is a measure of the added distortion $\boldsymbol{\delta}$. $C(\cdot)$ is the DNN classification function and the target label is t.

ADMM provides a systematic way to deal with combinatorial constraints by breaking the initial problem into two subproblems. To do this, we first transform the initial problem into the following one, introducing an auxiliary variable $\boldsymbol{z}$:

$$\begin{aligned} \min_{\boldsymbol{\delta},\boldsymbol{z}} \quad & D(\boldsymbol{\delta}) + g(\boldsymbol{x}+\boldsymbol{z}) \\ s.t. \quad & \boldsymbol{\delta} = \boldsymbol{z} \\ & (\boldsymbol{x}+\boldsymbol{z}) \in [0,1]^n \end{aligned} \tag{1}$$

where $g(\boldsymbol{x})$ has the form:

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

Here $Z(\boldsymbol{x})$ is the logits before the softmax layer. The augmented Lagrangian function of problem (1) is as follows:

$$L_\rho(\boldsymbol{\delta},\boldsymbol{z},\boldsymbol{u}) = D(\boldsymbol{\delta}) + g(\boldsymbol{x}+\boldsymbol{z}) + \boldsymbol{u}^T(\boldsymbol{\delta}-\boldsymbol{z}) + \frac{\rho}{2}\|\boldsymbol{\delta}-\boldsymbol{z}\|_2^2 \tag{3}$$

where $\boldsymbol{u}$ is the dual variable or Lagrange multiplier and $\rho > 0$ is called the penalty parameter. Using the scaled form of ADMM by defining $\boldsymbol{u} = \rho\boldsymbol{s}$, we have:

$$L_\rho(\boldsymbol{\delta},\boldsymbol{z},\boldsymbol{s}) = D(\boldsymbol{\delta}) + g(\boldsymbol{x}+\boldsymbol{z}) + \frac{\rho}{2}\|\boldsymbol{\delta}-\boldsymbol{z}+\boldsymbol{s}\|_2^2 - \frac{\rho}{2}\|\boldsymbol{s}\|_2^2 \tag{4}$$

Problem (1) is solved through iterations. In the k-th iteration, we follow the steps:

$$\boldsymbol{\delta}^{k+1} = \arg\min_{\boldsymbol{\delta}} \quad L_\rho(\boldsymbol{\delta},\boldsymbol{z}^k,\boldsymbol{s}^k) \tag{5}$$

$$\boldsymbol{z}^{k+1} = \arg\min_{\boldsymbol{z}} \quad L_\rho(\boldsymbol{\delta}^{k+1},\boldsymbol{z},\boldsymbol{s}^k) \tag{6}$$

$$\boldsymbol{s}^{k+1} = \boldsymbol{s}^k + \boldsymbol{\delta}^{k+1} - \boldsymbol{z}^{k+1} \tag{7}$$

In Eqn. (5), we find $\boldsymbol{\delta}^{k+1}$ which minimizes L_ρ with fixed $\boldsymbol{z}^k$ and $\boldsymbol{s}^k$. Similarly, in Eqn. (6), $\boldsymbol{\delta}^{k+1}$ and $\boldsymbol{s}^k$ are fixed and we find $\boldsymbol{z}^{k+1}$ minimizing L_ρ. $\boldsymbol{s}^{k+1}$ is then updated accordingly. Note that the two variables $\boldsymbol{\delta}$ and $\boldsymbol{z}$ are updated in an alternating or sequential fashion, from which the term *alternating direction* comes. It converges when:

$$\left\|\boldsymbol{\delta}^{k+1}-\boldsymbol{z}^{k+1}\right\|_2^2 \leq \varepsilon, \quad \left\|\boldsymbol{z}^{k+1}-\boldsymbol{z}^k\right\|_2^2 \leq \varepsilon \tag{8}$$

Equivalently, in each iteration, we solve two optimization subproblems corresponding to Eqns. (5) and (6), respectively:

$$\min_{\boldsymbol{\delta}} \quad D(\boldsymbol{\delta}) + \frac{\rho}{2}\|\boldsymbol{\delta}-\boldsymbol{z}+\boldsymbol{s}\|_2^2 \tag{9}$$

and

$$\min_{\boldsymbol{z}} \quad g(\boldsymbol{x}+\boldsymbol{z}) + \frac{\rho}{2}\|\boldsymbol{\delta}-\boldsymbol{z}+\boldsymbol{s}\|_2^2 \tag{10}$$

The non-differentiable $g(\boldsymbol{x})$ makes it difficult to solve the second subproblem (10). Therefore, we use a new differentiable $g(\boldsymbol{x})$ as follows inspired by [17]:

$$g(\boldsymbol{x}) = c \cdot \max\left(\left(\max_{i\neq t}(Z(\boldsymbol{x})_i) - Z(\boldsymbol{x})_t\right), -\kappa\right) \tag{11}$$

Then, stochastic gradient decent methods can be used to solve this subproblem. We use the Adam optimizer [27] due to its fast and robust convergence behavior. The new $g(\boldsymbol{x})$ of Eqn. (11) is inspired by [17], where κ is a confidence parameter denoting the strength of adversarial example transferability.

A. Box Constraint

The constraint on $\boldsymbol{z}$ i.e., $\boldsymbol{x}+\boldsymbol{z} \in [0,1]^n$ is known as a "box constraint" in the optimization literature. We use a new variable $\boldsymbol{w}$ and instead of optimizing over $\boldsymbol{z}$ defined above, we optimize over $\boldsymbol{w}$, based on:

$$\boldsymbol{z} = \frac{1}{2}\big(\tanh(\boldsymbol{w})+1\big) - \boldsymbol{x} \tag{12}$$

Since $-1 \leq \tanh(w_i) \leq 1$ and $0 \leq x_i + z_i \leq 1$, the method will automatically satisfy the box constraint.

B. Discussion on Constants

There are two constants c and ρ in the two subproblems (9) and (10). We adopt different policies for choosing appropriate c and ρ in L_2 and L_0 attacks. In L_2 attack, since ρ acts in both problems (9) and (10), we fix ρ and change c to improve the solutions. We find that the best choice of $c > 0$ is the smallest one that can help achieve $g(\boldsymbol{x}) = 0$ in the subproblem (10). Thus, a modified binary search is used to find a satisfying c. In addition, for the ADMM L_2 attack, due to the adaptive search of c, we find in experiments that changing ρ does not affect the results significantly. For the ADMM L_0 attack, ρ has stronger and more direct influence on the solutions, so c is fixed and adaptive search of ρ is utilized. More details are provided in Section IV-B.

IV. Instantiations of L_2 and L_0 Attacks Using ADMM Framework

The ADMM framework for adversarial attacks now reduces to two subproblems (9) and (10). The difference between L_2 and L_0 attacks lies in the subproblem (9) and for both attacks the subproblem (10) is solved using stochastic gradient descent.

A. L_2 Attack

For L_2 attack, the subproblem (9) has the form:

$$\min_{\boldsymbol{\delta}} \quad \|\boldsymbol{\delta}\|_2^2 + \frac{\rho}{2}\|\boldsymbol{\delta} - \boldsymbol{z} + \boldsymbol{s}\|_2^2 \tag{13}$$

the solution to which can be directly derived in an analytical format:

$$\boldsymbol{\delta} = \frac{\rho}{2+\rho}(\boldsymbol{z} - \boldsymbol{s}) \tag{14}$$

Then the complete solution to the L_2 attack problem using the ADMM framework is as follows: for each k-th iteration,

$$\boldsymbol{\delta}^{k+1} = \frac{\rho}{2+\rho}\left(\left(\frac{1}{2}\big(\tanh(\boldsymbol{w}^k)+1\big) - \boldsymbol{x}\right) - \boldsymbol{s}^k\right) \tag{15}$$

$$\begin{aligned}\boldsymbol{w}^{k+1} = &\min_{\boldsymbol{w}}\ g\Big(\frac{1}{2}\big(\tanh(\boldsymbol{w})+1\big)\Big) \\ &+ \frac{\rho}{2}\left\|\boldsymbol{\delta}^{k+1} - \Big(\frac{1}{2}\big(\tanh(\boldsymbol{w})+1\big) - \boldsymbol{x}\Big) + \boldsymbol{s}^k\right\|_2^2\end{aligned} \tag{16}$$

$$\boldsymbol{s}^{k+1} = \boldsymbol{s}^k + \boldsymbol{\delta}^{k+1} - \left(\frac{1}{2}\big(\tanh(\boldsymbol{w}^{k+1})+1\big) - \boldsymbol{x}\right) \tag{17}$$

Eqn. (15) corresponds to the analytical solution to the subproblem (9) i.e., problem (13) with Eqn. (12) replacing $\boldsymbol{z}$ in Eqn. (14). Eqn. (16) corresponds to the subproblem (10) with Eqn. (12) replacing $\boldsymbol{z}$ and g taking the form of Eqn. (11). The solution to Eqn. (16) is derived through the Adam optimizer for the stochastic gradient descent.

B. L_0 Attack

For L_0 attack, the subproblem (9) has the form:

$$\min_{\boldsymbol{\delta}} \quad \|\boldsymbol{\delta}\|_0 + \frac{\rho}{2}\|\boldsymbol{\delta} - \boldsymbol{z} + \boldsymbol{s}\|_2^2 \tag{18}$$

the solution to which can be derived in this way: let $\boldsymbol{\delta}$ be equal to $\boldsymbol{z} - \boldsymbol{s}$ first, then for each element in $\boldsymbol{\delta}$, if its square is smaller than $\frac{2}{\rho}$, make it zero.

When solving the subproblem (9), we enforce a hidden constraint on the distortion $\boldsymbol{\delta}$, that the square of each non-zero element in $\boldsymbol{\delta}$ must be larger than $\frac{2}{\rho}$. Therefore, a smaller ρ would push ADMM method to find $\boldsymbol{\delta}$ with larger non-zero elements, thus reducing the number of non-zero elements and decreasing L_0 norm. Empirically, we find the constant ρ represents a trade-off between attack success rate and L_0 norm of the distortion, i.e., a larger ρ can help find solutions with higher attack success rate at the cost of larger L_0 norm.

Then the complete solution to the L_0 attack problem using the ADMM framework can be derived similar to the L_2 attack Eqns. (15), (16), and (17), except that the optimal distortion $\boldsymbol{\delta}$ is a little different in each iteration.

V. Performance Evaluation

The proposed ADMM attacks are compared with state-of-the-art attacks, especially with C&W [17], on three image classification datasets, MNIST [28], CIFAR-10 [29] and ImageNet [1]. We compare our ADMM L_2 attack with C&W L_2 attack, FGM [30] and IFGM [8] L_2 attacks in terms of attack success rate (ASR) and distortion. Our ADMM L_0 attack is compared with C&W L_0 attack. We also test our attacks against two defenses, defensive distillation [12] and adversarial training [26]. The transferability of ADMM attacks are evaluated too.

We train two networks for MNIST and CIFAR-10 datasets, respectively, which can achieve 99.5% accuracy on MNIST and 80% accuracy on CIFAR-10. For ImageNet, we utilize a pre-trained Inception v3 network [31] which can achieve 96% top-5 accuracy.

A. Attack Success Rate and Distortion for ADMM L_2 attack

We compare our ADMM L_2 attack with FGM, IFGM and C&W L_2 attacks. The attack success rate (ASR) represents the percentage of the constructed adversarial examples that are successfully classified as target labels. The average distortion of all successful adversarial examples is reported. For zero ASR, its distortion is not available (N.A.). We perform the adversarial attacks on MNIST, CIFAR-10 and ImageNet. For MNIST and CIFAR-10, 1000 correctly classified images are randomly selected from the test sets and 9 target labels are tested for each image, so we perform 9000 attacks for each dataset using each attack method. For ImageNet, 100 correctly classified images and 9 target labels are utilized.

The parameter ρ is fixed to 20. The number of ADMM iterations is 10. When using Adam optimizer in each ADMM iteration, we do binary search for 9 steps on the parameter c (starting from 0.001) and runs 1000 learning iterations for each c with initial learning rate 0.02 for MNIST and 0.002 for CIFAR-10 and ImageNet.

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 since it is designed to be

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	L_2	ASR	L_2	ASR	L_2
MNIST	FGM(L_2)	99.4	2.245	34.6	3.284	0	N.A.
	IFGM(L_2)	100	1.58	99.9	2.50	99.6	3.958
	C&W(L_2)	100	1.393	100	2.002	99.9	2.598
	ADMM(L_2)	100	**1.288**	100	**1.873**	100	**2.445**
CIFAR-10	FGM(L_2)	99.5	0.421	42.8	1.157	0.7	3.115
	IFGM(L_2)	100	0.191	100	0.432	100	0.716
	C&W(L_2)	100	0.178	100	0.347	99.9	0.481
	ADMM(L_2)	100	**0.173**	100	**0.337**	100	**0.476**
ImageNet	FGM(L_2)	12	2.29	1	6.823	0	N.A.
	IFGM(L_2)	100	f1.057	100	2.461	98	4.448
	C&W(L_2)	100	0.48	100	0.681	100	0.866
	ADMM(L_2)	100	**0.416**	100	**0.568**	97	**0.701**

Table II
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	8.1	100	17.48	100	31.48
	ADMM(L_0)	100	8	100	15.71	100	25.87
CIFAR	C&W(L_0)	100	8.6	100	19.6	100	34.4
	ADMM(L_0)	100	8.25	100	18.8	100	31.2

fast, rather than optimal. Among IFGM, C&W and ADMM L_2 attacks, ADMM achieves the lowest 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 MNIST, the ADMM attack can reduce the L_2 distortion by about 7% compared with C&W L_2 attack. This becomes more prominent on ImageNet that ADMM reduces L_2 distortion by 19% comparing with C&W in the worst case.

B. Attack Success Rate and Distortion for ADMM L_0 attack

The performance of ADMM L_0 attack in terms of attack success rate and L_0 norm of distortion is demonstrated in this section. We compare our ADMM L_0 attack with C&W L_0 attack on MNIST and CIFAR-10.

For ADMM L_0 attack, 9 binary search steps are performed to search for the parameter ρ while c is fixed to 20 for MNIST and 200 for CIFAR-10. The initial value of ρ is set to 3 for MNIST and 40 for CIFAR-10, respectively. The number of ADMM iterations is 10.

The results of the L_0 attacks are shown in Table II. We can observe that both C&W and ADMM L_0 attacks can achieve 100% attack success rate. For the best case, C&W L_0 attack and ADMM L_0 attack have relatively close performance in terms of L_0 distortion. For the worst case, ADMM L_0 attack can achieve lower L_0 distortion than C&W. ADMM L_0 attack reduces the L_0 distortion up to 17% on MNIST.

C. ADMM Attack Against Defensive Distillation and Adversarial Training

ADMM attacks can break the undefended DNNs with high success rate. It is also able to break DNNs with defensive distillation. We perform C&W L_2 attack, ADMM L_2 attack and ADMM L_0 attack for different temperature parameters on MNIST and CIFAR-10. We find that the attack success rates of C&W L_2 attack, ADMM L_2 and L_0 attacks for different temperature T are all 100%. Since distillation at temperature T causes the value of logits to be approximately T times larger while the relative values of logits remain unchanged, C&W attack and ADMM attack which work on the relative values of logits do not fail.

We further test ADMM attack against adversarial training on MNIST. C&W L_2 attack and ADMM L_2 attack are utilized to separately generate 9000 adversarial examples with 1000 randomly selected images from the training set as sources. Then we add the adversarial examples with correct labels into the training dataset and retrain the network with the enlarged training dataset. Then we perform ADMM attack on the adversarially trained networks (one with C&W adversarial examples, and one with ADMM adversarial examples), as shown in Table III. ADMM L_2 attack can break all three networks with 100% success rate. L_2 distortions on the latter two networks are higher than that on the first network, showing some defense effect of adversarial training.

Table III
ADMM L_2 ATTACK AGAINST ADVERSARIAL TRAINING ON MNIST

Adversarial training	Best case		Average case		Worst case	
	ASR	L_2	ASR	L_2	ASR	L_2
None	100	1.35	100	2.07	100	2.63
C&W L_2	100	1.7	100	2.66	100	3.2
ADMM L_2	100	1.77	100	2.67	100	3.2

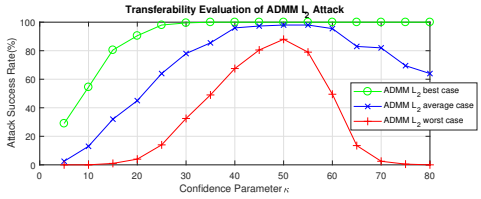

Figure 2. Transferiablity Evaluation of ADMM L_2 Attack on MNIST

D. Attack Transferability

Here we test the transferability of ADMM adversarial attack. For each value of the parameter κ, we use ADMM L_2 attack to generate 9000 adversarial examples with 1000 random images from MNIST as source. Then these examples are applied to attack the defensively distilled network with temperature $T = 100$. The ASR is reported in Fig. 2.

As demonstrated in Fig. 2, when $\kappa = 0$, the generated adversarial examples are not transferable or not strong enough to break the defended network. As κ increases, the ASRs of the three cases also increase. When $\kappa = 50$, the ASRs of three cases can achieve the maximum value and most of the generated adversarial examples on the undefended network can also break the defensively distilled network. We also find that when $\kappa > 50$, the ASRs of average case and worst case decrease as κ increases. The reason is that when κ is very large, it is quite difficult to generate adversarial examples even for the undefended network. Thus an decrease on the ASR is observed for average case and worst case, and the advantages of strong transferable adversarial examples are mitigated by the difficulty to generate such strong attacks.

VI. CONCLUSION

In this paper, we propose an ADMM-based general framework for adversarial L_2 and L_0 attacks. We compare our ADMM attacks with state-of-the-art adversarial attacks, showing the effectiveness of the proposed ADMM attacks against the defensive distillation and adversarial training.

ACKNOWLEDGEMENTS

This work is supported by the National Science Foundation (CCF-1733701, CNS-1704662, CNS-1739748, CMMI-1750531 and ECCS-1609916), Air Force Research Laboratory FA8750-18-2-0058, and U.S. Office of Naval Research. The fourth author gratefully acknowledges financial support from the National Science Foundation under awards CAREER CMMI-1750531 and ECCS-1609916. We thank researchers at the US Naval Research Laboratory for their comments on previous drafts of this paper.

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] N. Carlini, P. Mishra, T. Vaidya, Y. Zhang, M. Sherr, C. Shields, D. Wagner, and W. Zhou, "Hidden voice commands.," in *USENIX Security Symposium*, pp. 513–530, 2016.

[7] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 427–436, 2015.

[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.

[22] S. M. Moosavi Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, no. EPFL-CONF-218057, 2016.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[23] 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.

[24] 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.

[25] 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.

[26] 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.

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

[28] 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.

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

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

[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.