It is double pleasure to deceive the deceiver: disturbing classifiers against adversarial attacks

1st João G. Zago
Department of Automation and Systems, UFSC
Florianópolis, Brazil
joao.zago@posgrad.ufsc.br

2nd Eric A. Antonelo
Department of Automation and Systems, UFSC
Florianópolis, Brazil
eric.antonelo@ufsc.br

3rd Fabio L. Baldissera
Department of Automation and Systems, UFSC
Florianópolis, Brazil
fabio.baldissera@ufsc.br

4th Rodrigo T. Saad
Department of Automation and Systems, UFSC
Florianópolis, Brazil
rsaad@das.ufsc.br

***Abstract*—Convolutional neural networks (CNNs) for image classification can be fragile to small perturbations in the images they ought to classify. This fragility exposes CNNs to malicious attacks, resulting in safety concerns in many application domains. In this paper, we propose a simple yet efficient strategy for decreasing the effectiveness of black-box attacks that need to sequentially query the classifier network in order to build an attack. The general idea consists of applying controlled random disturbances (noise) at the softmax output layer of neural network classifiers, changing the confidence scores according to a set of design requirements. To evaluate this defense strategy, we employ a CNN, trained on the MNIST data set, and attack it with a black-box attack method from the literature called ZOO. The results show that our defense strategy: a) decreases the attack success rate of the adversarial examples; and b) forces the attack algorithm to insert larger perturbations in the input images.**

***Index Terms*—convolutional neural networks, adversarial attacks, defense strategies.**

I. Introduction

Despite achieving great performance in image classification [1], state-of-the-art convolutional neural networks (CNNs) are still vulnerable to small perturbations in the inputs they ought to classify [3]. One example is that of a neural network that mistakes a slightly perturbed image of a panda for a gibbon [4].

To better study this vulnerability phenomenon, many researchers have recently developed algorithms that generate so-called adversarial examples. These are inputs purposefully designed to induce a misclassification by the neural network (though not by a human being).

One can divide the attack algorithms into two main categories according to the attacker's knowledge about the network: white-box and black-box methods. White-box procedures are those that use at least one internal parameter of the neural network (e.g., its number of layers or values of its synaptic weights) to generate their attacks [3]–[12]. Black-box algorithms, on the other hand, need only the outputs of the neural network for some set of input images [13]–[17]. Usually, the latter makes use of the the confidence scores obtained from consecutive queries to the network in order to compute a small perturbation that is added to the input to make it adversarial.

To counteract such attacks, researchers have devised a wide range of defense strategies in the last few years [3], [4], [7], [12], [18]–[25]. Up to now, the attackers are winning this digital arms race, since the CNN-based classifiers seem to be inherently vulnerable to perturbed images that lie outside their training set probability distributions. Even though one can enlarge the training set with adversarial examples, as in adversarial training [3], [4], [7], [12], [19] or ensemble methods [20], it seems unfeasible to cover the training set with all possible adversarial examples.

In Section III of this paper we present an alternative approach to defend CNNs against black-box attacks that need to query the network under attack multiple times and have direct access to its softmax output layer. This layer returns the probability (or confidence score) associated with each possible output class. The goal of our methods is to confuse the attackers regarding the direction they should perturb the image to make it adversarial. To achieve that, we apply controlled random disturbances (noise) at the softmax output layer of the CNN, in such a way that the output probabilities keep some of their useful properties, such as score order and relative in-between class score distance as described in the problem formulation from Section II. To evaluate our defense strategies, we employ a CNN trained on the MNIST data set and attack it with a method called ZOO, the strongest black-box attack method proposed so far in the literature [15].

Section IV shows and discusses our experimental results. Depending on the parameter configurations, our defense method decreases the success rate of the adversarial images from 99.8% to 65.3%. It also forces the attacker to increase

This paper is a draft to be published in the IEEE International Conference on Systems, Man and Cybernetics (SMC 2020).

This work has been partially supported by CAPES - The Brazilian Agency for Higher Education (Finance Code 001), project PrInt CAPES-UFSC "Automation 4.0".

the magnitude of the perturbation added to the images.

Furthermore, the proposed method relies on the application of a controlled random disturbance during the network prediction step (after training), making it independent from the probability distribution that generates the training set, unlike previous work in training set augmentation or network distillation [18], [20].

II. Problem Formulation

Let N be any convolutional neural network for image classification and A be a black-box adversarial attack. We shall use $S(N)$ to denote the set of adversarial images generated by A for N, and $F(x) \in \mathbb{R}^m$ to represent the m-dimensional output of the softmax layer of N when presented with an n-dimensional image vector $x \in [0,1]^n$. This input x has to be classified into one of m available classes. We assume that some coding mechanism is available to convert matrices representing images to column vectors.

Let us now call N' the network that is equal to N, except for the output of its softmax layer, which is now given by $G(F(x)+\mathbf{d})$, where $\mathbf{d} \in \mathbb{R}^m$ is a controlled disturbance vector, and $G : \mathbb{R}^m \to \mathbb{R}^m$ is a normalization function, given by Equation (1). G makes the outputs of the network N' sum up to one, retaining the form of a probability distribution over the classes $i \in \{0, \ldots, m-1\}$.

$$G(F(x)+\mathbf{d})_i = \frac{(F(x)+\mathbf{d})_i}{\sum_{j=0}^{m-1}(F(x)+\mathbf{d})_j} \tag{1}$$

Our goal is to design such $\mathbf{d}$ in a way that:

1) the success rate of N' on the classification of $S(N')$ is higher than that of N for $S(N)$; and
2) for any $i, j \in \{0, \ldots, m-1\}$, if $F_i(x) \leq F_j(x)$, then $d_i + F_i(x) \leq d_j + F_j(x)$. That is, the disturbance vector $\mathbf{d}$ does not affect the ordering of the values of $F(x)$; and
3) for any $i \in \{0, \ldots, m-1\}$, the inequality $|d_i| \leq \delta * F_i(x)$ holds, where δ is a design parameter satisfying $0 < \delta < 1$. In other words, it limits the designed disturbance to be within the closed interval $[-\delta * F_i(x), \delta * F_i(x)]$.

III. Defense Proposals

We describe in this section three different approaches to design a disturbance vector $\mathbf{d}$ to be added to the output layer of a neural network.

The only one that satisfies all the requirements (1), (2), and (3) previously stated in the problem formulation is the last method, termed Limited Disturbed Classes with Order Preservation (Subsection III-C).

We present the other two, though, as a means to later evaluate the performance of our ideas when we relax the constraints on $\mathbf{d}$ posed by the problem formulation.

A. Homogeneously Disturbed Classes (HDC)

This first method generates a scalar disturbance $\mathbf{d}$ (or noise) that is homogeneously applied to each $F(x)_i$ using the same normal distribution $\mathcal{N}(0, \sigma_f)$ with 0 mean and standard deviation σ_f. The standard deviation σ_f, was defined to be equal to $\Delta \div r$, where Δ is equal to half of the difference between the largest and the second largest value of the vector $F(x)$. We divided Δ by r so that the random behavior of the disturbance does not get affected by the saturation function, defined further. For all the experiments presented in this paper, we employed $r = 3$.

In Algorithm 1, which describes the HDC method, the function $sort(F(x))$ returns a m-dimensional vector whose values are sorted in descending order, i.e., the largest value is the first element, represented by $sort_0(F(x))$. Additionally, $sat_z(x)$ denotes the saturation function, which is equal to x, if $|x| \leq z$, and to $z * sign(x)$, otherwise.

Algorithm 1: Homogeneously Disturbed Classes

Input: $F(x) \in \mathbb{R}^m$
$\sigma_f \leftarrow (sort_0(F(x)) - sort_1(F(x))) \div (2 * r)$;
for $i \in \{0, \ldots, m-1\}$ **do**
  Select a random d from distribution $\mathcal{N}(0, \sigma_f)$;
  $F(x)_i \leftarrow F(x)_i + sat_{\sigma_f * 3}(d)$;
end
return $G(F(x))$;

B. Disturbed Classes with Order Preservation (DCOP)

In this second approach, our goal is to change the HDC method so that the requirement (2) of the problem formulation is attained. To simplify the notation, and without loss of generality, we assume that the vector $F(x)$ given as input to our algorithm is already sorted in descending order. Still, we define the map $diff : \mathbb{R}^m \times \{0, \ldots, m-1\} \to \mathbb{R}$ to be

$$diff(\mathbf{v}, i) = \begin{cases} \mathbf{v}_i - \mathbf{v}_{i+1}, & i = 0 \\ \min\{\mathbf{v}_i - \mathbf{v}_{i+1}, \mathbf{v}_{i-1} - \mathbf{v}_i\}, & 1 \leq i \leq m-2 \\ \mathbf{v}_{m-2} - \mathbf{v}_{m-1}, & i = m-1 \end{cases}$$

This map computes the distance between some element i of $\mathbf{v}$ and its nearest neighbor, where $\mathbf{v}$ is the output probabilities of the classifier in descending order. That is, the $diff$ map computes the maximum allowable disturbance for each class.

The algorithm below shows that, now, the magnitude of each d_i is delimited by $diff(F(x), i)$, thus keeping the ordering of $F(x)$ intact.

Algorithm 2: Disturbed Classes with Order Preservation

Input: $F(x) \in \mathbb{R}^m$
for $i \in \{0, \ldots, m-1\}$ **do**
  $\sigma_{v,i} \leftarrow diff(F(x), i) \div (2 * r)$;
  Select a random d_i from distribution $\mathcal{N}(0, \sigma_{v,i})$;
  $F(x)_i \leftarrow F(x)_i + sat_{\sigma_{v,i} * 3}(d_i)$;
end
return $G(F(x))$;

Although this new version does satisfy requirement (2) of the problem formulation, it does not attain specification (3). So, in the next subsection, we deal with the last variant of our

disturbance approach, called Limited Disturbed Classes with Order Preservation.

C. Limited Disturbed Classes with Order Preservation (LDCOP)

In this approach, the standard deviation σ_v of $\mathcal{N}(0, \sigma_{v,i})$ is computed in the same way as in DCOP. However, $\delta * |F(x)|$ limits the magnitude of each d_i in this variant, as presented in the pseudocode that follows We assume that $F(x)$ is already sorted in descending order.

Algorithm 3: Limited Disturbed Classed with Order Preservation

Input: $F(x) \in \mathbb{R}^m$ and $\delta \in [0, 1]$;
for $i \in \{1, \ldots, m\}$ **do**
  $\sigma_{v,i} \leftarrow diff(F(x), i) \div (2 * r)$;
  Select a random d_i from distribution $\mathcal{N}(0, \sigma_{v,i})$;
  $F_i(x) \leftarrow F_i(x) + sat_{\delta * F_i(x)}(sat_{\sigma_{v,i} * 3}(d_i))$;
end
return $G(F(x))$;

IV. Results

First, we present in this section the three main elements used to evaluate our proposal: the CNN, the attack algorithm ZOO, and the experimental scenarios. Then, we show and discuss the obtained results.

A. Convolution neural network (CNN)

We employ a CNN of 12 layers (described in Table I) that makes use of convolution functions interleaved with batch normalization and max pooling operations.

In Table I, $\mathrm{conv(m, n)} - \mathrm{c}$ is used to refer to a convolutional layer with kernel of size $m \times n$ and c channels. ReLU activation function was applied for all the layers, except for the last one.

TABLE I
Neural Network Architecture

Layer	Type	Dimensions
0	Input	(32x32x3)
1	Conv(3x3)-32	(32x32x32)
2	Conv(3x3)-32	(32x32x32)
3	Batch Normalization	(32x32x32)
4	Max Pooling(2x2)	(16x16x32)
5	Conv(3x3)-64	(16x16x64)
6	Conv(3x3)-64	(16x16x64)
7	Batch Normalization	(16x16x64)
8	Max Pooling(2x2)	(8x8x64)
9	Fully Connected	(1024)
10	Batch Normalization	(1024)
11	Fully Connected	(1024)
12	Fully Connected	(10)

The classifier model presented above (Table I), was trained on the MNIST data set, which is a digit classification benchmark that contains images of single digits from 0 to 9 and the corresponding labels (10 possible classes). There are $70,000$ images in this data set, which we divided into three different disjoint subsets: training set, validation set, and test set.

The training set comprised $50,000$ images, whereas the validation and test sets, $10,000$ samples each. ADAM [2] was used as the optimization method to adjust the CNN parameters, minimizing a categorical cross-entropy loss on the training set. For such an optimization process we applied a learning rate of $5\mathrm{e}{-5}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 1\mathrm{e}{-08}$.

The CNN was trained for approximately 50 epochs, resulting in an accuracy of 99.59% and 97.71% on the training set and test set, respectively.

B. Zero Order Optimization Method, ZOO

Given a neural network N and an input image x_0 that belongs to the class l_0, the Zeroth Order Optimization Method (ZOO) computes the adversarial x^* associated to x_0 by solving the following optimization problem:

$$\min_x g(x) = \min_x(||x - x_0||_2^2 + c * f(x)) \quad (2)$$
$$s.t.\ x \in [0, 1]^n$$
$$f(x) = \max\{log[F_{l_0}(x)] - \max_{i \neq l_0}\{log[F_i(x)\}], -\kappa\}, \quad (3)$$

where $f(x)$ is a function that depends on the difference between the two highest confidence scores in $F(x)$, c and κ are positive constants, and $||x - x_0||_2^2$ is a normalization factor used to obtain adversarial images x^* as close as possible to x_0.

According to Equation (2), the adversarial x^* is an image similar to x_0 that makes the neural network change its classification output from l_0 to another class. To solve the optimization problem above, the ZOO attack approximates the gradient of $F(x)$ with respect to the coordinates x_i by the quotient:

$$\frac{\partial f(x)}{\partial x_i} \approx \frac{f(x + h * e_i) - f(x - h * e_i)}{2 * h}, \quad (4)$$

where h is a small and fixed constant and e_i is the standard basis vector with the i-th component set to one and all other components equal to zero.

Finally, the increment added at each iteration of the optimization algorithm to obtain the image x_i^{k+1} from x_i^k is modulated by a factor η, e.g., if the Newton's method is used to solve Equation (2), then $x_i^{k+1} = x_i^k + \eta * \frac{\partial f}{\partial x_i} \Big/ \frac{\partial f^2}{\partial x_i^2}$, instead of simply $x_i^{k+1} = x_i^k + \frac{\partial f}{\partial x_i} \Big/ \frac{\partial f^2}{\partial x_i^2}$. In this study, we employed the ADAM optimizer [2] to solve the optimization problem presented in Equation 2.

C. Experiments

Below we describe the test scenarios used to evaluate the performance of our defense approaches:

1) Set the maximum number of iterations of the ZOO algorithm to 10;
2) Set $h = 0.0001$ and the optimizer parameters $\beta_1 = 0.9, \beta_2 = 0.99$ and $\epsilon = 1\mathrm{e}{-8}$, as proposed by the authors of the ZOO method [15];
3) Randomly select 500 images from the training set[1];

[1]Preliminary experiments show equivalent results with images selected from the test set.

TABLE II
DEFENSE METHODS FOR CNN CLASSIFIER

v	Method
0	Original CNN
1	HDC
2	DCOP
3	LDCOP10
4	LDCOP20
5	LDCOP50

4) For each $v \in \{0, \ldots, 5\}$, obtain the corresponding neural network N_v according to Table II, where LDCOPX denotes the LDCOP defense strategy with $\delta = X/100$.;
5) For each $\eta \in \{0.01, 0.05, 0.1\}$, compute the set of adversarial images $S_\eta(N_v)$ using the ZOO algorithm;
6) Classify the sets of examples $S_\eta(N_v)$ with the network N_v.

D. Results

We evaluate our defense method along with two main directions: the reduction in the success rate of the attack, and the increment in the magnitude of the perturbation inserted by the attacker.

1) Our defense decreases the success rate of the adversarial attacks generated by ZOO: Figure 1 shows the success rate of the adversarial images generated by ZOO for the original network as well as for the networks that are enhanced with our defense methods, i.e., with the disturbance $\mathbf{d}$ added to $F(x)$. We observe a significant improvement in the accuracy of those networks with defenses on the classification of the candidate adversarial images. Our method reduced the attack success rate of ZOO from 99.8% to 65% in the most restricted scenario (variant LDCOP10). The different performances of the defense strategies are associated with the restrictions imposed on the additive disturbance $\mathbf{d}$: the more properties of $F(x)$ the disturbance $\mathbf{d}$ must preserve, the less effective is the defense. In other words, as the LDCOP approach preserved properties (1), (2), and (3) from problem formulation its performance was expected to be lower or equal in comparison to the other methods, while HDC preserves the property (1).

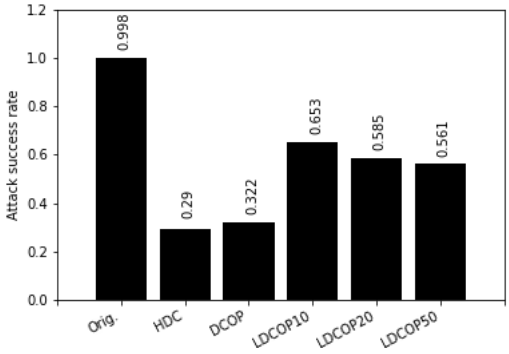

Fig. 1. Attack success rate for the ZOO algorithm against different variants of CNNs, with and without defenses. The leftmost bar shows the result for the original CNN without any defense.

2) Our defense forces the attackers to increase the perturbation added to the images: Let $I = \{x_0, x_1, \ldots, x_i\}$ be a set of images $x_k \in \mathbb{R}^n$ and $S(N_v) = \{x_0^{adv}, x_1^{adv}, \ldots, x_i^{adv}\}$ be the corresponding set of adversarial images generated by ZOO for the network N_v, $v \in \{0, \ldots, 5\}$. Define α_k to be $||x_k^{adv} - x_k||_2$, i.e., the 2-norm of the perturbation added to x_k in order to generate an adversarial x_k^{adv}.

Figure 2 presents the distribution of the α_k for all six network variants. We can see that the defenses forced the attack algorithm to increase the magnitude of the applied perturbation. The higher the magnitude of α_k, the less similar is x_k^{adv} to x_k, making x_k^{adv} a weaker and possibly a more questionable adversarial image. Therefore, our method not only decreases the success rate of the attacking images, but also forces the attackers to generate adversaries of worse quality.

Figure 2 also shows the distribution of those α_k associated with the images x_k that indeed fooled the classifier (black bars). We observe that for smaller perturbations ($\alpha_k \leq 3$), the percentage of adversarial examples declined from 57.1% for the original network to 7.0% for the network defended with the LDCOP10 approach, corresponding to a reduction of 87.7% in that interval.

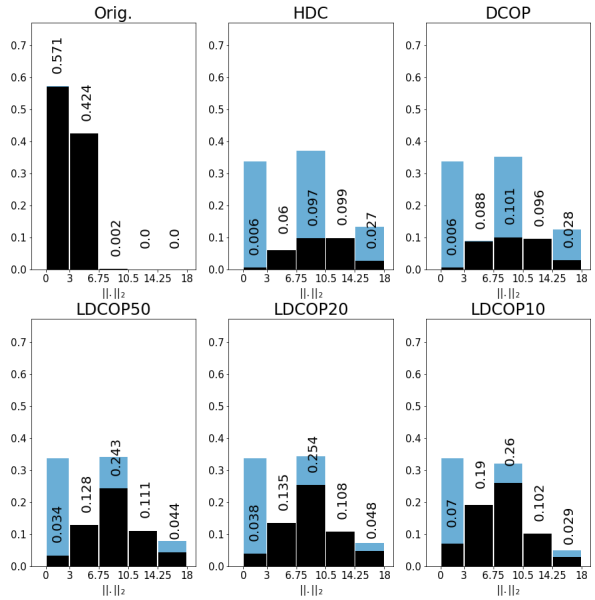

Fig. 2. Distribution of the magnitude of perturbations added by ZOO to generate adversarial images from a given set of images. The x-axis is divided into five intervals of perturbation magnitudes. The y-axis shows the relative frequency associated with each of these intervals. For example, 57.1% of the adversarial images generated for the original network have a perturbation magnitude between 0 and 3, whereas only 0.6% of all images lie in this interval for the network defended with HDC. The black bars correspond to effective adversarial examples, whereas the blue ones represent the total candidates of adversaries. The first interval is smaller than the others as the Euclidean norm is greater or equal to 0.

Figure 3 depicts the quality degradation of the adversarial images generated by the ZOO algorithm. For each column, it displays the candidate adversarial image generated by ZOO and the label predicted by the respective network. Note how these images are visually more perturbed in columns 2, 3, and 4 (the ones where we employ our defenses) when compared to

column 1. This degradation is a consequence of the increased perturbation forced by the defense methods.

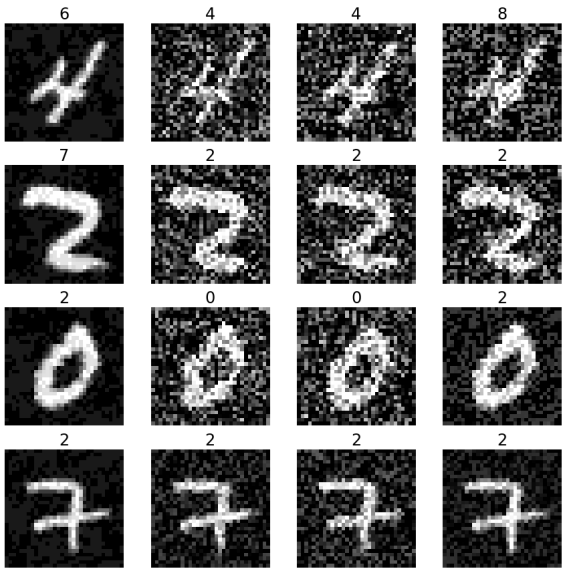

Fig. 3. Examples of images generated by ZOO to attack different CNNs trained on MNIST. Starting at the leftmost column, we have a sample of images generated for the following networks: 0 (Original CNN), 1 (HDC), 2 (DCOP) and 3 (LDCOP with $\delta = 0.1$). The class with highest score according to the corresponding classifier is shown above each image. Note how the quality of the adversarial images worsen when a defense method is employed.

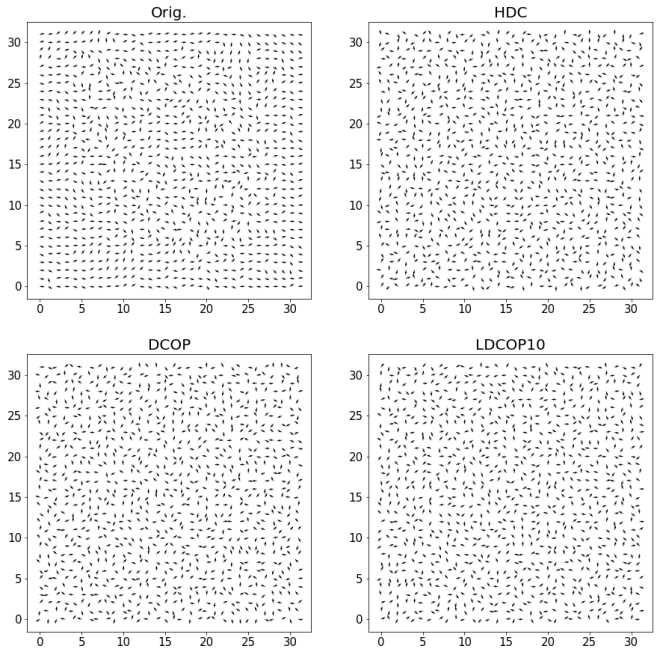

Fig. 4. Visualization of the approximate gradient (Equation 4), built by ZOO at its first algorithm iteration for a particular image and different defense strategies for the CNN. Each plot is formed by a matrix of arrows which represent an approximation of the loss function gradient with respect to each input pixel. The arrow length was fixed and its angle represents the gradient value for the pixel position. The defense is active except for the top leftmost plot.

Figure 4 shows the gradient approximation computed using the ZOO algorithm for each of the proposed defense methods. We can see that the gradients computed for N_v, with $v \in \{1, 2, 3\}$ (when the defense is active) have no spatial regularities through the image pixels when compared with the gradient computed for the neural network N_0 (*Orig.*). This is a result of the disturbance $\mathbf{d}$ applied to the output probabilities of the neural network $F(x)$, inducing uncertainty in the gradient approximation at every iteration of the attack. The attacker is left confused about which direction it should perturb the image: the uncertainties perceived by the attacker are spatial (through the image) and temporal (at each iteration).

V. Conclusions

In this work, we have proposed a new defense strategy against black-box adversarial attacks. It consists of adding a controlled random disturbance to the softmax layer of a neural network classifier. This intervention acts exclusively during the network prediction phase. We have designed three different approaches to compute such disturbance. The difference among them lies in the set of requirements the disturbance vector must satisfy, such as preserving the ordering of the most probable classes, and/or be proportional to the neurons' outputs.

In order to evaluate our approach, we have trained a CNN on the MNIST data set and attacked it with images generated by the ZOO algorithm. Our findings show that the proposed defense methods make the trained CNN less vulnerable to adversarial images crafted by ZOO. The defenses considerably reduce the success rate of the attacks, while still keeping the usability of the network predictions, i.e., the general form of the probability distribution over classes. For instance, the LDCOP10 method, which keeps the class ordering of the softmax output and limits the noise magnitude, still attains a significant reduction in the attack success rate generated by ZOO from 99.8% to 65.3%.

Our defense methods also force the attacker to increase the magnitude of the perturbation necessary to cause a misclassification. Therefore, if the network misclassifies an image generated by the attacker, it is probable that this image either: cannot be classified as adversarial, due to its high perturbation; or can be detected by other defense methods [21]–[25].

This combination of methods, left as future work, would result in a much stronger defense since they are orthogonal: while most methods depend on augmenting the training data set or retraining the network, ours act exclusively on the network prediction and does not explicitly depend on a particular data set.

Additionally, we plan to investigate the performance of our defenses in different settings, such as: a) against other black-box attack algorithms that depends on the output of the softmax layer; b) when the classifier is trained on different data sets (e.g.: CIFAR-10, CIFAR-100); c) when different probability distributions (such as the uniform distribution) are employed to compute the required controlled disturbance.

Finally, because of its simplicity and ease of application, we advocate the usage of the proposed strategy as a standard method to evaluate the robustness of new black-box attack methods.

REFERENCES

[1] C. Szegedy, S. Ioffe, V. Vanhoucke, A. Alemi, "Inception-v4, Inception-ResNet and the impact of residual connections on learning". AAAI Conference on Artificial Intelligence, 2016.

[2] D. P. Kingma, B. Jimmy. "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980 (2014).

[3] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I.J. Goodfellow, R. Fergus, "Intriguing properties of neural networks". CoRR, abs/1312.6199.

[4] I.J. Goodfellow, J. Shlens, C. Szegedy, "Explaining and harnessing adversarial examples". CoRR, abs/1412.6572.

[5] A. Kurakin, I.J. Goodfellow, S. Bengio, "Adversarial examples in the physical world". ArXiv, abs/1607.02533.

[6] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, A. Swami, "The Limitations of Deep Learning in Adversarial Settings". 372-387. 10.1109/EuroSP.2016.36.

[7] S. Moosavi-Dezfooli, A. Fawzi, P. Frossard, "DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks". 2574-2582. 10.1109/CVPR.2016.282.

[8] N. Carlini, D. Wagner, "Towards Evaluating the Robustness of Neural Networks". 39-57. 10.1109/SP.2017.49.

[9] S. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, P. Frossard, "Universal Adversarial Perturbations". 86-94. 10.1109/CVPR.2017.17.

[10] S. Sabour, Y. Cao, F. Faghri, D.J. Fleet, "Adversarial Manipulation of Deep Representations". CoRR, abs/1511.05122.

[11] A. Athalye, N. Carlini, D.A. Wagner, "Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples". ICML 2018.

[12] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, A. Vladu, "Towards Deep Learning Models Resistant to Adversarial Attacks". ArXiv, abs/1706.06083.

[13] N. Papernot, P.D. McDaniel, I.J. Goodfellow, "Transferability in Machine Learning: from Phenomena to Black-Box Attacks using Adversarial Samples". ArXiv, abs/1605.07277.

[14] N. Narodytska, S. Kasiviswanathan, "Simple Black-Box Adversarial Attacks on Deep Neural Networks," 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Honolulu, HI, 2017, pp. 1310-1318. doi: 10.1109/CVPRW.2017.172

[15] P. Chen, H. Zhang, Y. Sharma, J. Yi, and C. Hsieh, "ZOO: Zeroth Order Optimization Based Black-box Attacks to Deep Neural Networks without Training Substitute Models". In Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security (AISec '17). Association for Computing Machinery, New York, NY, USA, 15–26. DOI:https://doi.org/10.1145/3128572.3140448.

[16] J. Su, , Vargas, D.V., & Sakurai, K. (2017). One Pixel Attack for Fooling Deep Neural Networks. IEEE Transactions on Evolutionary Computation, 23, 828-841.

[17] Z. Zhao, D. Dua, S. Singh, "Generating Natural Adversarial Examples". ArXiv, abs/1710.11342.

[18] N. Papernot, P.D. McDaniel, X. Wu, S. Jha, A. Swami, "Distillation as a Defense to Adversarial Perturbations Against Deep Neural Networks". 2016 IEEE Symposium on Security and Privacy (SP), 582-597.

[19] A. Kurakin, I.J. Goodfellow, S. Bengio, "Adversarial Machine Learning at Scale". ArXiv, abs/1611.01236.

[20] F. Tramèr, A. Kurakin, N. Papernot, D. Boneh, P.D. McDaniel, "Ensemble Adversarial Training: Attacks and Defenses". ArXiv, abs/1705.07204.

[21] J. Lu, T. Issaranon, D. Forsyth, "SafetyNet: Detecting and Rejecting Adversarial Examples Robustly". 446-454. 10.1109/ICCV.2017.56.

[22] J.H. Metzen, T. Genewein, V. Fischer, B. Bischoff, "On Detecting Adversarial Perturbations". ArXiv, abs/1702.04267.

[23] R. Feinman, R.R. Curtin, S. Shintre, A.B. Gardner, "Detecting Adversarial Samples from Artifacts". ArXiv, abs/1703.00410.

[24] K. Grosse, P. Manoharan, N. Papernot, M. Backes, P.D. McDaniel, "On the (Statistical) Detection of Adversarial Examples". ArXiv, abs/1702.06280.

[25] Y. Song, T. Kim, S. Nowozin, S. Ermon, N. Kushman, "PixelDefend: Leveraging Generative Models to Understand and Defend against Adversarial Examples". CoRR abs/1710.10766 (2017).