BOSS: BIDIRECTIONAL ONE-SHOT SYNTHESIS OF ADVERSARIAL EXAMPLES

Ismail R. Alkhouri[1] *Alvaro Velasquez*[2] *George K. Atia*[1,3]

[1] Department of Electrical and Computer Engineering, University of Central Florida, Orlando FL, USA

[2] Information Directorate, Air Force Research Laboratory, Rome NY, USA

[3] Department of Computer Science, University of Central Florida, Orlando FL, USA

ABSTRACT

The design of additive perturbations to the inputs of classifiers has become a central focus of adversarial machine learning. An alternative approach is to synthesize adversarial examples using structures akin to generative adversarial networks, albeit with the use of large amounts of training data. By contrast, this paper considers the one-shot synthesis of adversarial examples that requires only a single reference datum. In particular, we explore solutions where the generated data must simultaneously satisfy user-defined constraints on its structural similarity to the reference input datum and the output of the classifier of interest it induces. This gives rise to what we call the Bidirectional One-Shot Synthesis (BOSS) problem. We prove that the BOSS problem is **NP-complete**. The experimental results verify that the targeted and confidence reduction attack methods developed either outperform or on par with state-of-the-art methods.

Index Terms— One-Shot Synthesis, Adversarial Attacks, Trained Classifiers, Generative Models, Targeted and Confidence Reduction Attacks, Decision Boundary Examples

1. INTRODUCTION

The problem of robustness is being assessed in adversarial machine learning via additive perturbations to data and the synthesis of adversarial examples, which are often used to test the robustness of a given model. In this paper, we reconcile the notion of one-shot learning [1] and the synthesis of adversarial examples for the first time in what we call one-shot synthesis. In particular, given a datum $\mathbf{x}_d$ and a pre-trained model $p(\,.\,;\theta)$ parameterized by θ, we propose a synthesis procedure that generates a new datum $\mathbf{x}$ to be used as an input to $p(\,.\,;\theta)$ such that constraints are satisfied on both the input structure and the output inference. In terms of the input, we ensure that $\mathbf{x}$ is similar to the given reference datum $\mathbf{x}_d$ by enforcing a small distance $d(\mathbf{x},\mathbf{x}_d) \leq \delta_s$. In terms of the output, we generate $\mathbf{x}$ such that it approximately induces a user-defined output distribution p_d as the inference result $p(\,.\,;\theta)$ of the pre-trained model by enforcing a small distance $D(p(\mathbf{x};\theta),p_d) \leq \delta_c$. In this sense, the underlying Bidirectional One-Shot Synthesis (BOSS) problem is concerned with generating data satisfying constraints on both the input and output directions of the given classifier $p(\,.\,;\theta)$. By controlling the induced output distribution, our approach generalizes traditional notions of targeted and non-targeted attacks [2]. Confidence reduction attacks can also be implemented in our approach, where the goal is to lower the confidence level of the true label to cause ambiguity [3], specifically against systems for which a confidence threshold is introduced and the classification is only regarded if the prediction confidence score is above that threshold [4].

We propose a solution to the BOSS problem by leveraging generative models whose parameters are updated based on the distance between the given datum $\mathbf{x}_d$ (distribution p_d) and the synthesized input datum $\mathbf{x}$ (output inference $p(\mathbf{x}\,;\theta)$). See Figure 1 for the problem description and BOSS samples.

It is worth noting that our generative approach is a one-shot synthesis solution in that it only requires a single datum $\mathbf{x}_d$, which mitigates the excessive data requirements of popular methods based on Generative Adversarial Networks (GANs) [10]. In fact, our proposed framework is more similar to the additive attack methods where a large body of works are presented such as the CW attack [11], the L-BFGS attack [12], Deepfool [13], Fast Adaptive Boundary (FAB) attack [14], saliency map attack [3], and NewtonFool [15].

The contributions of the paper are the following. First, we present the BOSS problem to synthesize feature vectors that follow some desired input and output specifications. Second, we prove that BOSS is **NP-complete**. Our third contribution is the proposed algorithmic procedure that is based on generative networks and the back-propagation algorithm [16] to produce (from scratch) these examples in a white-box settings. Fourth, we present methods to select the input/output specifications to generate targeted adversarial attacks, confidence reduction attacks, and decision boundary samples. On different attack evaluation metrics, we show that BOSS either on par or outperform state-of-the-art methods. Further, we show samples from small-scale and large-scale datasets

Fig. 1: Problem demonstration (*left*) and examples (*right*). The true labels are placed on the left of each desired features (image) with bold font. First row of images represent the desired features where the desired PMFs are placed on the left of each sample. The second row presents the synthesized examples, by BOSS, with their corresponding predictions w.r.t their trained classifiers. From left to right, samples are picked from the MNIST digits [5], MNIST fashion [6], CIFAR-10 [7], GTSRB [8], and ImageNet [9], respectively.

on famous state-of-the-art classification architectures.

2. PROBLEM FORMULATION & CHARACTERIZATION

Suppose we have some trained model p with parameters θ (e.g., a trained Neural Network) and a probability distribution $p(.\,;\theta) : \mathbb{R}^N \to \Delta^M$ over the output of the model with entries $p_m(\mathbf{x}\,;\theta)$ for $m \in [M] := \{1, 2, \ldots, M\}$, where M is the total number of outputs, and Δ^M is the probability simplex over M dimensions.

Given a clean example (desired input features) $\mathbf{x}_d$, the well-known formulation of the basic iterative extension of the Fast Gradient Sign Method (FGSM) method [17] generates an adversarial example $\mathbf{x}$ by minimizing some differentialable loss function between $p(\mathbf{x};\theta)$ and p_d. The distance between $\mathbf{x}$ and $\mathbf{x}_d$, however, is restricted to the l_p norm. A more general formulation is used in [11] where the loss functions on the input and output of the classifier of interest can be chosen more flexibly. Therefore, we will compare our approach to the attacks in [11] and an advanced version in [18].

Let $d : \mathbb{R}^N \times \mathbb{R}^N \to [0, 1]$ and $D : \Delta^M \times \Delta^M \to [0, 1]$ denote distance functions between two feature vectors and distributions, respectively, where a value 0 indicates identical arguments.

Definition 1 (BOSS Problem). *Given a learning model $p(.\,;\theta) : \mathbb{R}^N \to \Delta^M$ parameterized by θ, a tensor $\mathbf{x}_d \in \mathbb{R}^N$, and a probability distribution $p_d \in \Delta^M$, find an input tensor $\mathbf{x} \in \mathbb{R}^N$ such that $d(\mathbf{x}, \mathbf{x}_d) \leq \delta_s$ and $D(p(.\,;\theta), p_d) \leq \delta_c$, where upper bounds δ_s and δ_c and loss functions d and D are given.*

First, we prove that the BOSS problem is **NP-complete**. This establishes that, in general, there is no polynomial-time solution to the BOSS problem unless **P = NP**. In Section 3, we develop a generative approach to obtain an approximate solution to the BOSS problem.

Definition 2 (CLIQUE Problem). *Given an undirected graph* $G = (U, E)$ *and an integer k, find a fully connected subgraph induced by $U' \subseteq U$ such that $|U'| = k$.*

Theorem 1. *The Bidirectional One-Shot Synthesis (BOSS) problem in Definition 1 is **NP-complete**.*

Proof. It is easy to verify that the BOSS problem is in **NP** since, given a tensor $\mathbf{x}$, one can check whether the input and output constraints $d(\mathbf{x}, \mathbf{x}_d) \leq \delta_s$ and $D(p(\mathbf{x}\,;\theta), p_d) \leq \delta_c$ are satisfied in polynomial time. It remains to be shown whether the BOSS problem is **NP-hard**. We will establish this result via a reduction from the CLIQUE problem in Definition 2. Given a CLIQUE instance $\langle G = (U, E), k \rangle$ with $|U| = n$ and $|E| = m$, we construct its corresponding BOSS instance $\langle p(.\,;\theta), \mathbf{x}_d, p_d, \delta_s, \delta_c \rangle$ as follows. Let $\mathbf{x}_d = \mathbf{0}$ denote the all-zeroes vector and let p_d be defined as the desired output distribution below

$$p_d = \left(\frac{e^{\epsilon k}}{e^{\epsilon k(k-1)/2} + e^{\epsilon k}} \,,\, \frac{e^{\epsilon k(k-1)/2}}{e^{\epsilon k(k-1)/2} + e^{\epsilon k}} \right)^T, \quad (1)$$

where $\epsilon \leq 1 - \sqrt{1 - (1/(k+1))}$. Finally, let $\delta_s = k/n$ and $\delta_c = 0$. We choose the mean square error loss (MSE) function to compute $d(\mathbf{x}, \mathbf{x}_d) \leq \delta_s$. The choice of loss function for computing $D(p(\mathbf{x};\theta), p_d) \leq \delta_c$ is superfluous since we have chosen $\delta_c = 0$. For the given trained model $p(.\,;\theta)$, we define its connectivity and parameters θ as follows. The input layer consists of n entries given by the solution $\mathbf{x} \in [0, 1]^n$. There is one hidden layer consisting of $n+m$ ReLU functions $\sigma_1, \ldots, \sigma_{n+m}$ such that the first n ReLU functions have a bias term of $\epsilon - 1$ and the next m ReLU functions have a bias term of $\epsilon - 2$. Finally, there is an output layer with two softmax output activation functions $p_1(\mathbf{x}\,;\theta)$ and $p_2(\mathbf{x}\,;\theta)$. Let θ_{ij}^h denote the weight of the connection between the ith input x_i and the jth ReLU activation σ_j in the hidden layer. For each $u_i \in U$ in the given CLIQUE instance, we have $\theta_{ii}^h = 1$. The outputs of these n ReLU activation functions are fully connected to the softmax output activation function $p_1(\mathbf{x}\,;\theta)$, each with a corresponding weight of 1. For each edge $e_k = (u_i, u_j) \in E$, we have $\theta_{i,n+k}^h = \theta_{j,n+k}^h = 1$. This defines the input connectivity of ReLU functions σ_{n+1} to σ_{n+m}. The outputs of

Fig. 2: Example reduction from a graph G (*left*) to a classifier $p(.\,;\theta)$ (*right*).

these are then fully connected to the second softmax function $p_2(\mathbf{x}\,;\theta)$, each with weight 1. See Figure 2 for an example. We now prove that there is a clique of size k in G if and only if there is a feasible solution $\mathbf{x}$ to the reduced BOSS instance.

($\implies$) Assume there is a clique of size k in G. We can derive a feasible solution $\mathbf{x}$ to the reduced BOSS instance as follows. For every vertex $u_i \in U$ in the clique, let $x_i = 1$ and let all other values of $\mathbf{x}$ be 0. The corresponding MSE loss is $d(\mathbf{x}, \mathbf{x}_{\mathrm{d}}) = k/n$, thereby satisfying the input constraint defined by δ_{s}. The solution $\mathbf{x}$ induces an output of $\sigma_i(x_i + (\epsilon - 1)) = \epsilon$ for each entry of $\mathbf{x}$ corresponding to a vertex u_i in the clique and an output of 0 for all other entries. Thus, we have k inputs of value ϵ into the first softmax output function. Now, let us consider the edges induced by this clique. For each edge $e_k = (u_i, u_j) \in E$ in the clique, we have $\sigma_{n+k}(x_i + x_j + (\epsilon - 2)) = \epsilon$ and an output of 0 for all other edges. Since there are $k(k-1)/2$ edges in a clique of size k, this yields $k(k-1)/2$ inputs of value ϵ into the second softmax output function. Thus, we have $p(\mathbf{x}\,;\theta) = p_{\mathrm{d}}$ and the constraint $D(p(\mathbf{x};\theta), p_{\mathrm{d}}) \le 0 = \delta_{\mathrm{c}}$ is satisfied. As a caveat, it is worth noting that, under this construction, the all-zeroes vector $\mathbf{0}$ yields equal outputs $p_1(\mathbf{0};\theta) = p_2(\mathbf{0};\theta) = 1/2$. Per the preceding arguments, it is also the case that a feasible solution $\mathbf{x}$ derived for a clique of size $k = 3$ will output $p_1(\mathbf{x};\theta) = p_2(\mathbf{x};\theta) = 1/2$. This is because $k = k(k-1)/2$ for $k = 3$. Thus, for the remainder of the proof, we assume that cliques of interest are of size $k > 3$.

($\impliedby$) We prove the contrapositive. That is, if there is no clique of size k in G, then the reduced BOSS instance is infeasible. We proceed by showing that there must be exactly k non-zero entries in $\mathbf{x}$ in order to satisfy constraints $d(\mathbf{x}, \mathbf{x}_{\mathrm{d}}) \le k/n$ and $D(p(\mathbf{x}\,;\theta), p_{\mathrm{d}}) \le 0$ and that, if there is no clique of size k, then there is no choice of k non-zero entries in $\mathbf{x}$ that will satisfy $D(p(\mathbf{x}\,;\theta), p_{\mathrm{d}}) \le 0$. Note that there must be at least k entries in $\mathbf{x}$ with value strictly greater than $(1 - \epsilon)$ in order to yield an input of ϵk into the first softmax output function and satisfy the first entry in p_{d}. Let us con-

sider the minimum MSE loss for a solution $\mathbf{x}$ with more than k non-zero entries. For $k + 1$ entries of value strictly greater than $(1 - \epsilon)$, we have $d(\mathbf{x}, \mathbf{x}_{\mathrm{d}}) = (k + 1)(1 - \epsilon)^2/n$. With some algebraic manipulation, we have that, for any value of $\epsilon \le 1 - \sqrt{1 - (1/(k+1))}$, $(k + 1)(1 - \epsilon)^2/n > k/n$, thereby violating the constraint $d(\mathbf{x}, \mathbf{x}_{\mathrm{d}}) \le \delta_{\mathrm{s}}$. Thus, there must be exactly k non-zero entries in $\mathbf{x}$. Now, let us consider the second softmax output function, which requires an input of $\epsilon k(k - 1)/2$. Since there is no clique of size k in G, any choice of k vertices in G will induce a set of edges whose cardinality is strictly less than $k(k - 1)/2$. Therefore, the output of the second softmax function will be strictly less than the second entry in p_{d}. This violates the constraint $D(p(\mathbf{x}\,;\theta), p_{\mathrm{d}}) \le 0$. $\qquad\square$

Note that, for a given CLIQUE instance in the proof of Theorem 1, the corresponding reduced BOSS instance is such that, if there exists a polynomial-time solution to the BOSS problem, then we could use this solution to solve the CLIQUE problem in polynomial time. This would imply that **P = NP**. We therefore conjecture that a polynomial-time solution to the BOSS problem is not likely to exist.

3. GENERATIVE APPROACH

To obtain a solution to the BOSS problem in Definition 1, we take a generative approach in which $\mathbf{x}$ is obtained as the output of a generative network, $g(.\,;\phi) : \mathbb{R}^Q \to \mathbb{R}^N$, with parameters ϕ, i.e., $g(\mathbf{z}\,;\phi) = \mathbf{x}$, where $\mathbf{z} \in \mathbb{R}^Q$ is a random input to the generative network. We utilize the adjustable parameters of network g for the objectives of BOSS. Therefore, we define the combined network $h(.\,;\psi) : \mathbb{R}^Q \to [M]$, whose layers are the concatenation of the layers of g and p, where $\psi = \{\phi, \theta\}$. In other words, $h(\mathbf{z}\,;\psi) = p(\mathbf{x}\,;\theta) = p(g(\mathbf{z}\,;\phi)\,;\theta)$. We augment a repeated version of vector $\mathbf{z}$ to create a small training dataset. Given the two objectives of BOSS, and the utilization of the adjustable parameters of network h, ϕ, we introduce the surrogate losses $\mathcal{L}_h(p(g(\mathbf{z}\,;\phi)\,;\theta), p_{\mathrm{d}})$ and $\mathcal{L}_g(g(\mathbf{z}\,;\phi), \mathbf{x}_{\mathrm{d}})$, and use the back-propagation algorithm [16] to optimize ϕ based on the minimization

$$\min_{\phi} \left[\mathcal{L}_g\Big(g(\mathbf{z}\,;\phi), \mathbf{x}_{\mathrm{d}}\Big) + \lambda \mathcal{L}_h\Big(p(g(\mathbf{z}\,;\phi)\,;\theta), p_{\mathrm{d}}\Big) \right], \quad (2)$$

where λ is a loss weight. It is important to note that (2) is used to update parameters ϕ while the trained classifier parameters θ remain unchanged. Due to the use of network h, the surrogate loss functions $\mathcal{L}_g$ and $\mathcal{L}_h$ can be selected as the MSE and the categorical cross-entropy loss, respectively.

In the following, we present an algorithmic approach to solve BOSS by iteratively optimizing (2). At every iteration, the adjustable parameters ϕ of the generator model g are updated to satisfy the two objectives of small PMF distance from p_{d} and high similarity of the generated example to $\mathbf{x}_{\mathrm{d}}$. We

Algorithm 1 BOSS Algorithm

Input: $\mathbf{z}$, $p(\,.\,;\theta)$, g, $\mathbf{x}_d$, p_d, δ_c, δ_s
Output: $\mathbf{x}$
1: **Initialize** $\mathbf{x}$, ϕ, λ
2: **while** $D(p(\mathbf{x}\,;\theta), p_d) \geq \delta_c$ **or** $d(\mathbf{x}, \mathbf{x}_d) \geq \delta_s$
3:     **obtain** ϕ as the minimizer of (2) with λ
4:     $\mathbf{x} = g(\mathbf{z}\,;\phi)$
5:     **update** λ using (3)
6: **return** $\mathbf{x}$

---

define an exit criteria if either a maximum number of iterations/steps is reached, or if a feasible solution per Definition 1 is found given $\mathbf{x}_d$, p_d, δ_s, and δ_c.

The parameter λ in (2) weighs the relative importance of each loss function to both avoid over-fitting and handle situations in which the solver converges for one loss function prior to the other [19]. We propose a dynamic update that depends on the distance D between the desired specification p_d and the actual output $p(g(\mathbf{z}\,;\phi)\,;\theta)$ at every iteration. Specifically, we update λ as

$$\lambda \leftarrow \sigma\left(\lambda - \lambda^0 \frac{\delta_c}{D}\,\mathrm{sign}\left(\frac{\delta_c}{D} - 1\right)\right). \qquad (3)$$

As such, it is required to have the distance function D returning values in the range of $[0, 1]$. Here, we utilize the Jensen-Shannon (JS) divergence distance [20], which returns 0 for two equivalent PMFs and is upper bounded by 1. The updates are also a function of the initial selection of λ denoted λ^0. In this paper we focus on the task of image classification. The authors in [21] proposed the LPIPS distance metric as a measure of similarity that mimicks human perceptibility. However, since this metric is classifier-dependent, here we use a universal metric. Specifically, we set $d = 1 - I$, where I is the Structural Similarity Index (SSIM) [22], which is equal to 1 for two identical images and captures luminance, contrast, and structure in the measurements.

The signum function $\mathrm{sign}(.)$ is used to determine whether to increase or decrease λ, based on the ratio of the actual and desired specifications which regulates the amount of change. The ReLU function $\sigma(\,.\,)$ prevents λ from becoming negative. This occurs when the desired p_d is easily attained in early steps of the algorithm. The procedure is presented in Algorithm 1.

4. EXPERIMENTAL RESULTS

We show results for Targeted attacks which we call BOSS-T. The desired distribution is selected such that $p_d(l) = 1$ if l is the target entry and 0 otherwise. Second, Confidence reduction examples, which we dub BOSS-C. Let the true label of $\mathbf{x}_d$ be f^* and the desired confidence be c_d, then $p_d(l) = c_d$

Table 1: BOSS-C and NewtonFool with $c_d = 0.6$, $\delta_c = 0.2$, and $\delta_s = 0.85$ for the MNIST dataset.

Environment	CA(%)	σ_{con}(%)	σ_s(%)	σ_{JS}
Model	98.1	99.1	100	0.03
Model+BOSS-C	98.1	66.73	87.89	0.19
Model+NewtonFool [15]	75.5	67.85	97.02	0.48

Fig. 3: Samples from each class of the CIFAR-10 dataset (columns). The first row shows the original examples. Rows 2-4 represent the synthesized images for BOSS-C, BOSS-B, and BOSS-T, respectively. The rounded percentage values of the confidence level, c, of the BOSS-C samples are placed at the bottom of each image along with the predicted label. For BOSS-B, the first pair at the bottom of every image represents the highest two predicted labels along with their rounded classification scores (second pair). Predicted labels are placed at the bottom of each BOSS-T example. The percentage of the rounded similarity measure (I) is placed on top of each generated example.

if $l = f^*$, and $(1 - c_d)/(M - 1)$ otherwise. In addition to the samples from BOSS-T and BOSS-C, we also show instances of boundary adversarial examples. In this case, a value of 0.5 is assigned to the two class labels on both sides of the boundary.

We use D as the JS distance to compare PMFs (desired and actual) and the SSIM index I as a measure of similarity between examples. We define σ_{JS} and σ_s as the average of D and I, respectively, over the set of observations $\mathcal{X}$. In addition to the aforementioned metrics, for BOSS-T, we utilize the *attack success rate* $\alpha =: N_s/|\mathcal{X}|$, where N_s is the number of times a generated adversarial sample is classified as the predefined target label. Further, we use σ_2 and σ_∞ to denote the average l_2 and l_∞, respectively. For BOSS-C, we compute σ_{con}, defined as the average confidence level of prediction of the true label over the set of interest $\mathcal{X}$.

The random vector $\mathbf{z}$ of dimension $Q = 100$ is generated from a uniform distribution over the interval $[0, 1]$, and 80 repeated samples are used for training. The initial loss weights are chosen as $\lambda^0 = \lambda_v^0 = 0.001$. The parameters are updated using the ADAM optimizer [23] with initial step size 0.025. The details of the pre-trained classifiers and the generative networks, and our code are available online[1].

---

[1]https://github.com/ialkhouri/BOSS

Table 2: BOSS-T attack overall comparison with state-of-the-art attack methods using the CIFAR-10 dataset. For targets, all labels other than the predicted ones are considered.

Attack	$\alpha(\%)$	σ_2	σ_∞	σ_s	Average Adversarial Confidence	Average Run Time (sec)
CW-l_2 ($\kappa = 0$) [11]	99.55	0.4195	0.0519	0.9966	0.4371	97.0982
CW-l_2 ($\kappa = 10$) [11]	95.2381	0.7755	0.0907	0.9889	0.9964	95.5681
CW-l_∞ ($\kappa = 0$) [11]	99.55	1.0761	0.1188	0.9780	0.7126	0.1744
CW-l_∞ ($\kappa = 10$) [11]	96.82	1.91	0.1676	0.9999	0.9966	0.8912
EAD (EN decision) [18]	100	0.4419	0.0885	0.9961	0.3951	141.971
EAD (l_1 decision) [18]	100	0.5618	0.173	0.9938	0.3754	142.394
BOSS-T (MSE)	100	1.1589	0.1046	0.9818	0.9879	16.5554
BOSS-T (Huber)	100	1.1454	0.1075	0.9799	0.9806	21.6079
BOSS-T (log cosh)	100	1.1055	0.1042	0.9828	0.9785	21.1732

Original	'Eagle': 0.99	'Convertible': 0.97	'Combination Lock': 0.98
BOSS-T	$\sigma_s = 0.82$ $\sigma_{JS} = 0.12$ 'Goldfinch': 0.98	$\sigma_s = 0.96$ $\sigma_{JS} = 0.28$ 'Beer Bottle': 0.84	$\sigma_s = 0.93$ $\sigma_{JS} = 0.32$ 'Library': 0.81
BOSS-B	$\sigma_s = 0.78$ $\sigma_{JS} = 0.18$ 'Red Wine': 0.54 'Espresso': 0.39	$\sigma_s = 0.95$ $\sigma_{JS} = 0.41$ 'Desk Computer': 0.26 'Dial Telephone': 0.46	$\sigma_s = 0.95$ $\sigma_{JS} = 0.25$ 'Goldfish': 0.32 'White Shark': 0.58

Fig. 4: Clean samples from some classes of the ImageNet dataset on VGG16. The first row shows the original examples. Rows 2 and 3 show the synthesized images for BOSS-T and BOSS-B, respectively. The labels are given at the bottom of each example along with the classification confidence. The SSIM and JS measures are reported on the right of each BOSS image.

For BOSS-C, Table 1 presents the results for σ_{con}, σ_s, and σ_{JS}. For NewtonFool, we use 50 iterations and set the small perturbations parameter as $\eta = 0.01$. For an average confidence of $\sigma_{con} \approx 67\%$, BOSS-C (with $c_d = 0.6$) and NewtonFool are successful in reducing the average confidence of the model from the original value $\sigma_{con} = 99.1\%$. This is accomplished with very high level of similarity measure of $\sigma_s \approx 88\%$ and $\sigma_s \approx 97\%$ for BOSS-C and NewtonFool, respectively. While NewtonFool attack produces examples with higher σ_s, it fails to maintain the classification accuracy CA which drops from 98.1% to 75.5%, and yields a large distance σ_{JS} from the desired PMF.

The results for BOSS-T are presented in Table 2 and compared to the state-of-the-art CW [11] and elastic nets attacks (EAD) [24]. We choose these baselines since their formulations admit any differentiable loss function, unlike the well-known Projected Gradient Descent method [25] where the distance between $\mathbf{x}$ and $\mathbf{x}_d$ is limited to the l_p norm. For each testing example, all labels other than the predicted one are used as targets. Results of the average adversarial confidence and average run time are reported for each case in

the last two columns. The parameters for CW and EAD on CIFAR10 are selected from the reported parameters in their respective papers. It is important to note that both of these methods apply their attacks based on the pre-softmax output (sometimes called logits), and hence, cannot specify an exact p_d. However, in the CW formulation, the parameter κ was introduced to represent the desired logit value to achieve higher adversarial confidence. While setting $\kappa = 0$ in the CW attack returns the best result in terms of imperceptibility, it does not yield the best adversarial confidence. Therefore, we report results for $\kappa = 10$, which yields a better tradeoff between both measures. Furthermore, we implement BOSS-T with different surrogate loss functions in $\mathbf{x}$ and $\mathbf{x}_d$.

While some variant of EAD and CW achieve a relatively lower imperceptibility (as seen from σ_2 and σ_∞), in terms of adversarial confidence, all variants of BOSS-T return the best results. CW, with $\kappa = 10$, reports similar adversarial confidence, but the attack success ratio does not achieve 100%, and requires 5 times the run time for l_2 and nearly 50% increase in imperceptibility (presented in σ_2 and σ_∞) are observed.

Figures 1, 3, and 4 show BOSS-C, BOSS-T, and BOSS-B samples. As observed, BOSS is successful in generating adversarial examples given the desired input specification, represented the original images in the first row, and output specification as represented in the corresponding application.

5. CONCLUSION

We introduced BOSS, a framework for one-shot synthesis of adversarial samples that satisfy input and output specifications for pre-trained classifiers. We formulated the BOSS problem and proved that the problem is **NP-Complete**. We developed an approximate solution using generative networks and surrogate loss functions. The flexibility of BOSS is demonstrated through various applications, including synthesis of boundary examples, targeted attacks, and reduction of confidence samples. A set of experiments verify that BOSS, in general, performs on par with state-of-the-art methods and generates the highest adversarial confidence examples.

6. REFERENCES

[1] Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni, "Generalizing from a few examples: A survey on few-shot learning," *ACM Computing Surveys (CSUR)*, vol. 53, no. 3, pp. 1–34, 2020.

[2] Gabriel Resende Machado, Eugênio Silva, and Ronaldo Ribeiro Goldschmidt, "Adversarial machine learning in image classification: A survey towards the defender's perspective," *arXiv preprint arXiv:2009.03728*, 2020.

[3] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami, "The limitations of deep learning in adversarial settings," in *IEEE European Symposium on Security and Privacy (EuroS&P)*, 2016, pp. 372–387.

[4] David Stutz, Matthias Hein, and Bernt Schiele, "Confidence-calibrated adversarial training: Generalizing to unseen attacks," in *Proceedings of the 37th International Conference on Machine Learning*, Hal Daumé III and Aarti Singh, Eds. 13–18 Jul 2020, vol. 119 of *Proceedings of Machine Learning Research*, pp. 9155–9166, PMLR.

[5] Yann LeCun, Corinna Cortes, and CJ Burges, "Mnist handwritten digit database," *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, vol. 2, 2010.

[6] Han Xiao, Kashif Rasul, and Roland Vollgraf, "Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms," *CoRR*, vol. abs/1708.07747, 2017.

[7] Alex Krizhevsky et al., "Learning multiple layers of features from tiny images," 2009.

[8] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition," *Neural Networks*, , no. 0, pp. –, 2012.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[10] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C Courville, and Yoshua Bengio, "Generative adversarial nets," in *NIPS*, 2014.

[11] Nicholas Carlini and David Wagner, "Towards evaluating the robustness of neural networks," in *IEEE Symposium on Security and Privacy*, 2017, pp. 39–57.

[12] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus, "Intriguing properties of neural networks," *preprint arXiv:1312.6199*, 2013.

[13] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2574–2582.

[14] Francesco Croce and Matthias Hein, "Minimally distorted adversarial examples with a fast adaptive boundary attack," in *International Conference on Machine Learning*. PMLR, 2020, pp. 2196–2205.

[15] Uyeong Jang, Xi Wu, and Somesh Jha, "Objective metrics and gradient descent algorithms for adversarial examples in machine learning," in *Proceedings of the 33rd Annual Computer Security Applications Conference*, 2017, pp. 262–277.

[16] Martin Riedmiller and Heinrich Braun, "A direct adaptive method for faster backpropagation learning: The rprop algorithm," in *IEEE International Conference on Neural Networks*, 1993, pp. 586–591.

[17] Alexey Kurakin, Ian Goodfellow, and Samy Bengio, "Adversarial machine learning at scale," *arXiv preprint arXiv:1611.01236*, 2016.

[18] Pin-Yu Chen, Yash Sharma, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh, "Ead: elastic-net attacks to deep neural networks via adversarial examples," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018, vol. 32.

[19] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio, *Deep learning*, vol. 1, MIT press Cambridge, 2016.

[20] Jianhua Lin, "Divergence measures based on the shannon entropy," *IEEE Transactions on Information theory*, vol. 37, no. 1, pp. 145–151, 1991.

[21] Cassidy Laidlaw, Sahil Singla, and Soheil Feizi, "Perceptual adversarial robustness: Defense against unseen threat models," in *International Conference on Learning Representations*, 2020.

[22] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[23] Diederik P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[24] Pin-Yu Chen, Yash Sharma, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh, "Ead: Elastic-net attacks to deep neural networks via adversarial examples," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, Apr. 2018.

[25] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.