Are DNNs fooled by extremely unrecognizable images?*

Soichiro Kumano*  Hiroshi Kera†  Toshihiko Yamasaki‡

Abstract

Fooling images are a potential threat to deep neural networks (DNNs). These images are not recognizable to humans as natural objects, such as dogs and cats, but are misclassified by DNNs as natural-object classes with high confidence scores. Despite their original design concept, existing fooling images retain some features that are characteristic of the target objects if looked into closely. Hence, DNNs can react to these features. In this paper, we address the question of whether there can be fooling images with no characteristic pattern of natural objects locally or globally. As a minimal case, we introduce single-color images with a few pixels altered, called sparse fooling images (SFIs). We first prove that SFIs always exist under mild conditions for linear and nonlinear models and reveal that complex models are more likely to be vulnerable to SFI attacks. With two SFI generation methods, we demonstrate that in deeper layers, SFIs end up with similar features to those of natural images, and consequently, fool DNNs successfully. Among other layers, we discovered that the max pooling layer causes the vulnerability against SFIs. The defense against SFIs and transferability are also discussed. This study highlights the new vulnerability of DNNs by introducing a novel class of images that distributes extremely far from natural images.

1 Introduction

It is well known that deep neural networks (DNNs) can be fooled by images far from those used for training [1–5]. A typical example is fooling images [4]. In contrast to notorious adversarial examples [6], which are subtly perturbed natural images, fooling images are designed to distribute extremely far from natural images and thus unrecognizable to humans. Nevertheless, fooling images are classified by DNNs into certain classes with high confidence scores (Figure 1). This reveals a crucial difference between human and DNN perception, highlighting the vulnerability of learning by DNNs on natural images. However, existing fooling images retain some class-specific patterns (carapace of armadillo highlighted by cyan square and baseball stitch). Our SFI, proposed as a new class of fooling images, does not have any features perceivable by humans locally or globally. (b) “Centipede” images. Adversarial examples are similar to natural images and fooling images are not so. In particular, SFIs are extremely different from natural images.

* kumano@cvm.t.u-tokyo.ac.jp
† kera@chiba-u.jp
‡ yamasaki@cvm.t.u-tokyo.ac.jp

1 This work is partially financially supported by JSPS KAKENHI Grant Number JP19K22863.

Figure 1: (a) An example of images making DNNs behave strangely. Although fooling images are not recognizable to humans as natural images, the classifiers recognize them as natural objects. However, existing fooling images still retain some class-specific patterns (carapace of armadillo highlighted by cyan square and baseball stitch). Our SFI, proposed as a new class of fooling images, does not have any features perceivable by humans locally or globally. (b) “Centipede” images. Adversarial examples are similar to natural images and fooling images are not so. In particular, SFIs are extremely different from natural images.
image) can significantly increase the confidence score on a certain class and change the classifier prediction. Nevertheless, we discovered that DNNs classify SFIs into classes of natural objects with high confidence scores (Figure 1a).

Through theoretical and empirical analysis, we reveal why and to what extent DNNs are vulnerable to SFIs. We first prove that SFIs always exist under mild conditions for linear functions, one-hidden-layer neural networks, and kernel methods. The theoretical conditions we derived suggest that as the number of altered pixels increases, SFIs are more likely to exist; namely, complex models are more vulnerable to SFIs.

For the empirical analysis using DNNs, we developed two methods of generating SFIs 1. The first method is based on differential evolution (DE) 7,8 and applied to a class-targeted setting. The second method is a transformation from a natural image to an SFI and is applied to an image-targeted setting. Extensive experiments revealed that SFIs, which have no recognizable features at all, are classified into classes with high confidence scores by the complex DNNs (e.g., VGG16 9 and ResNet50 10). The reason for this can be found in the similarity of deep features between SFIs and natural images. We revealed that features of SFIs, which show a completely different distribution from that of natural images in the input space, become similar to those of natural images after passing through several layers of DNNs. To examine which layer makes DNNs vulnerable, we compared the four network architectures: multi-layer perceptron (MLP), a convolutional neural network (CNN) with and without max pooling layers (denoted as CNNMP and CNNNone, respectively), and CapsNet 11. We observed that SFIs fool CNNMP most successfully, i.e., max pooling leads to the vulnerability of DNNs. Max pooling discards spatial information and makes decisions based on just whether lower-level features exist, not how they are connected. As a defense against SFIs, we tested out-of-distribution (OOD) detection, which tries to classify the SFIs into an additional class. After training with SFIs in the OOD framework, the trained model successfully detected the SFIs in the test dataset. However, it was still possible to generate another set of SFIs that successfully fools such a model. In addition, the trained model did not detect most SFIs with a base color that was different from those used during training. These results indicate the difficulty of detecting SFIs through OOD detection and suggest the inherent vulnerability of DNNs. Lastly, we show that SFIs generated for one model can transfer to another model. Thus, altered pixels of SFIs form some features that are unrecognizable but commonly learned by different models.

Our contributions are summarized as follows.

- We reveal a new vulnerability of DNNs through SFIs. In contrast to adversarial examples and existing fooling images, SFIs have neither local nor global features, and distribute extremely far from natural images. Analysis

  on such unrecognizable images is important in terms of safe and security of DNNs.

- We first prove the existence of SFIs under mild conditions for three models (Theorems 1–3). The conditions imply that complex models are more vulnerable to SFIs.

- We propose DE- and transformation-based methods to generate SFIs for empirical studies, and demonstrate that the features of natural images and SFIs were similar in deep layers of DNNs. This can partially explain why SFIs are classified into certain classes with high confidence scores.

- By testing four different architectures (MLP, CNNMP, CNNNone, and CapsNet), we reveal the defect of max pooling.

- We find that OOD-based defense against SFIs is not sufficient, i.e., various SFIs can be generated to attack the defended model.

- We show that the SFIs transfer between models. This suggests that DNNs might learn common completely unrecognizable features to humans.

The results listed above all indicate the potential threat of images that are extremely far from natural images. While SFIs are an example of such potentially harmful images, there can be more variety of such making-no-sense images that fool DNNs. We believe that extensive studies on them will be crucial for the safe and secure applications of DNNs.

2 Related Work

Fooling Images and Out-of-Distribution Images. Nguyen et al. proposed fooling images, which do not make sense to humans, but are classified by DNNs with high confidence scores into certain classes (Figure 1a) 4. Some parts and the overall pattern of the fooling images have the characteristics of natural objects. The authors hypothesized that fooling images produce high-confidence predictions despite being far from the natural images in the class because they are typically far from the decision boundary and deep in a classification region. Hein et al. proved that DNNs with a rectified linear unit (ReLU) activation produce high-confidence predictions for samples sufficiently far from the training data 2. In their study, by relying on the fact that ReLU DNNs split the input space into linear regions 12 in most cases, it was shown that for any input sample x, the scaling ax would fall into a consistent linear region when a exceeds a certain value (say, a0). Within such a linear region, the ReLU DNN consistently classifies ax into the same class for any a ≥ a0, and the probability (softmax value) assigned to this class increases monotonically along a. In addition, Simonyan et al. generated white noise-like images by computing the gradient of the class score with respect to the input image to confirm what the CNNs capture 13. These images are not natural images, but have some specific features similar to the target class. In this paper, we deal with

---

Notes:

1. Our code will be available on https://github.com/s-kumano/
sparse-fooling-image
completely unrecognizable images, i.e., images that do not have class-specific features locally or globally. The existence of our proposed images, SFIs, is proved theoretically and experimentally. Note that Hein et al. theoretically proved that any image with extreme scaling can fool DNNs, but they assume unbounded pixel values. In contrast, with SFIs, we theoretically and experimentally demonstrated an example of extreme OOD with bounded pixel values. SFIs suggest the fundamental vulnerability and risk of DNNs that completely featureless images, even single-color images with a few pixels altered, can break the prediction of DNNs.

Adversarial Examples. A similar but distinct concept from fooling images is adversarial examples, which are natural images with small malicious perturbations [6, 14, 27]. The small perturbations imply that adversarial examples are close to the original natural images. Since we address the question of whether it is possible to fool DNNs by images that are extremely far from natural ones, we claim that adversarial examples should be separated from SFIs, and thus they are not of our interest. In particular, one pixel attack [27] might seem similar to our study in that it changes the predictions of DNNs with a change in the small number of pixels; however, its research objective is different from ours. The study of one pixel attack focuses on local perturbations of natural images, whereas our study aims to investigate the potential threat of images that are extremely far from natural ones, we claim the question of whether it is possible to fool DNNs by images that are close to the original natural images. Since we address the existence for several linear and nonlinear models.

3 SFI Generation

An SFI comprises a single color image with a small number of altered pixels. In particular, we refer to an SFI with $k$ altered pixels as a $k$-SFI. For example, the SFIs in Figure 4 are 5-SFIs. A formal definition of $k$-SFIs will be provided later (cf. Definition 1).

We propose two methods for generating SFIs. The first method is a DE-based generation in the class-targeted setting, which works in a semi-black-box setting where only confidence scores are assumed to be accessible. The second method is a transformation-based generation in the image-targeted setting, which works in a white-box setting where all parameters and scores are assumed to be accessible. The second method is later used to highlight that even two significantly distant images can be interpreted similarly by DNNs.

3.1 SFI generation through differential evolution

Following the framework proposed by Su et al. [27], we use DE to optimally select the position and intensity of pixels in the SFIs. The generation of a grayscale $k$-SFI is conducted as follows:

1. Candidate solutions are first generated for $s$. Each candidate solution is a $k \times 3$ matrix. The row of the matrix is $(x, y, c)$, which represents the position $(x, y)$ and the intensity $c$ of the pixel to be altered. The position of a pixel is sampled from a uniform distribution of all pixel positions, and the intensity of a pixel is sampled from the normal distribution $N\left(\frac{m+n}{2}, \frac{m-n}{2}\right)$ of the initial step, where $[m, n]$ is the predefined range of pixel intensity. To maintain the pixel intensity $c$ in the range $[m, n]$, $c$ is mapped to $n$ if $c$ is higher than $n$, and to $m$ if $c$ is lower than $m$.

2. Repeat the following steps for $t = 1, 2, \ldots, T$.

   (a) A tentative group of candidate solutions $\{v_{i,t+1}\}$, $i = 1, 2, \ldots, s$ is generated from the current one $\{x_{i,t}\}$, $i = 1, 2, \ldots, s$ as $v_{i,t+1} = x_{r_1,t} + F \cdot (x_{r_2,t} - x_{r_3,t})$. Then, three non-identical indices of the candidate solutions, $r_1$, $r_2$, and $r_3$ are randomly selected. In addition, $F$ is the scaling factor (here, $F = 0.5$).

   (b) The original group of candidate solutions is updated by replacing $x_{i,t}$ with $v_{i,t+1}$ when $v_{i,t+1}$ has a higher confidence score on the target class; that is, $x_{i,t+1} = v_{i,t+1}$ if $f(x_{i,t}) < f(v_{i,t+1})$ and $x_{i,t+1} = x_{i,t}$, otherwise, where $f(\cdot)$ denotes the confidence score on the target class of the SFI generated from a candidate solution and the predesignated base intensity $g$.

3. After $T$ iterations, an SFI is generated from the candidate solution with the highest confidence score on the target class in the current group of candidate solutions.

3.2 SFI generation through deep feature inversion

To generate image-targeted SFIs, we impose two requirements on an SFI: (i) the features of the SFI through the model are similar to those of a natural image, and (ii) almost all the pixels of the SFI have the same value, which can be formulated as

$$\alpha^* = \arg \min_{\alpha} \frac{||\Phi(z_b 1 + \alpha) - \Phi(x_0)||_2^2}{||\Phi(x_0)||_2^2} + \lambda_{\text{sparse}}||\alpha||_1, \quad (1)$$

where $|| \cdot ||_p$ is the $\ell_p$ norm of a vector, $\Phi(\cdot)$ is the output of the DNN or its intermediate layer, $\lambda_{\text{sparse}}$ is the weight for the regularization, and $z_b 1$ is a single color image (i.e., $z_b$ is

In the RGB color image, a five-dimensional vector $(x, y, R, G, B)$ is used.
zero to generate black-based SFIs. The first term represents using backpropagation. 

By repeating this step, we obtain an SFI. We search for sparse images where $\Phi(x)$ are similar using backpropagation.

Here, the first term explicitly requires a similarity between the output of the CNNs. Specifically, we discuss conditions for the existence of SFIs. We can design an optimal $k$-SFI, and a sufficient and necessary condition is given as follows:

**Theorem 1.** Let $W = [w_1, w_2]^{\top} \in \mathbb{R}^{2 \times D}$ be the weight matrix of the linear function $y(z) = Wz$ for the binary classification of classes $C_1$ and $C_2$. Let $\delta_w = (\delta_{w,1}, \ldots, \delta_{w,D})^{\top} = w_1 - w_2$. For any $W$, there exists a $k$-SFI $z \in [r, R]^D$ with a confidence score of at least $M \in [0, 1]$ on $C_1$ if and only if the following holds: If $\delta_w^\top 1 > 0$, then

$$\ln \left( \frac{M}{1 - M} \right) \leq r\delta_w^\top 1 + (r - R) \sum_{i \in I_-(k)} \delta_{w,i},$$

where $I_-(k) \subset \{1, \ldots, D\}$ is the index set of (at most) $k$ smallest negative entries in $\delta_w$. Otherwise,

$$\ln \left( \frac{M}{1 - M} \right) \leq r\delta_w^\top 1 + (r - r) \sum_{i \in I_+(k)} \delta_{w,i},$$

where $I_+(k) \subset \{1, \ldots, D\}$ is the index set of (at most) $k$ largest positive entries in $\delta_w$.

**Proof.** Let us denote an SFI by $z = z_0 1 + \alpha \in [r, R]^D$. The confidence score of the SFI on $C_1$ is as follows:

$$p_1(z) = \frac{\exp(w_1^\top z)}{\exp(w_1^\top z) + \exp(w_2^\top z)}.$$ (4)

Thus, there exists an SFI with $p_1(z) \geq M$ when the following holds:

$$\frac{\exp(w_1^\top z)}{\exp(w_1^\top z) + \exp(w_2^\top z)} \geq M.$$ (5)

(5) can be simplified to

$$\ln \left( \frac{M}{1 - M} \right) \leq \sum_{d=1}^{D} \delta_{w,d}(z_0 + \alpha_d) =: f(z_0, \alpha).$$ (6)

Here, we consider maximizing $f(z_0, \alpha)$. If $\delta_{w,d} > 0$, $f(z_0, \alpha)$ can be maximized by $z_0 + \alpha_d = R$ because pixel intensities of SFIs should range in $[r, R]$. Similarly, if $\delta_{w,d} < 0$, $z_0 + \alpha_d = r$ can maximize $f(z_0, \alpha)$. Thus, if $\alpha$ had no constrained condition, we would choose

$$z_0 + \alpha_d = \begin{cases} R, & (\delta_{w,d} > 0) \\ r, & (\delta_{w,d} \leq 0) \end{cases},$$ (7)

to maximize $f(z_0, \alpha)$. However, $\alpha$ is restricted to have only $k$ nonzero entries. We can select the nonzero entries of $\alpha$ as follows: if $\delta_w^\top 1 > 0$, we define $I_-(k)$ as the index set of (at most) $k$ smallest negative entries in $\delta_w$. Then, for $i \in I_-(k)$,

4.1 Linear function
In conclusion, there exists a $k$-SFI with a confidence score of at least $M$ on $C_1$ if and only if
\[
\ln \left( \frac{M}{1 - M} \right) \leq \sum_{d_h=1}^{D_h} \delta_{v,d_h} \sigma \odot \left( \sum_{i \in I(k)} u_{d_h,i} \right) \tag{10}
\]
where $I(k)$ is determined by Algorithm 4.

**Theorem 2.** Let $\mathbf{U} = [\mathbf{u}_1 \cdots \mathbf{u}_{D_h}]^\top \in \mathbb{R}^{D_h \times D}$ be the weight of a hidden layer and $\mathbf{V} = [\mathbf{v}_1 \mathbf{v}_2]^\top \in \mathbb{R}^{2 \times D_h}$ be the weight of an output layer of a one-hidden-layer neural network $\mathbf{y}(\mathbf{x}) = \mathbf{V} (\sigma \odot (\mathbf{U} \mathbf{x}))$ for the binary classification between classes $C_1$ and $C_2$, where $D_h$ is dimension of the hidden layer and $\sigma \odot (\cdot)$ applies a ReLU function to each entry of a vector. Let $\delta_v = (\delta_{v,1} \cdots \delta_{v,D_h})^\top = \mathbf{v}_1 - \mathbf{v}_2$. For any $\mathbf{U}$, $\mathbf{V}$, and $M \in [0,1]$, there exists a $k$-SFI $\mathbf{z} \in [0,R]^D$ with a confidence score of at least $M$ on $C_1$ if
\[
\ln \left( \frac{M}{1 - M} \right) \leq \sum_{d_h=1}^{D_h} \delta_{v,d_h} \sigma \odot \left( \sum_{i \in I(k)} u_{d_h,i} \right) \tag{10}
\]

**Algorithm 1** Algorithm for determination of $I$ on a one-hidden-layer neural network

**Input:** $k \in \mathbb{N}$, $\delta_v \in \mathbb{R}^{D_h}$, $\mathbf{U} \in \mathbb{R}^{D_h \times D}$

**Output:** $I$

- $I \leftarrow \emptyset$
- $D \leftarrow \{1, \ldots, D\}$
- $h \leftarrow 0$
- **while** $|I| < n$ **do**
  - $i^* \leftarrow \arg \max_{i \in D} \sum_{d_h=1}^{D_h} \delta_{v,d_h} \sigma \odot \left( \sum_{j \in \{I \cup \{i^*\}\}} u_{d_h,j} \right)$
  - **if** $h_{\text{new}} > h$ **then**
    - $I \leftarrow I \cup \{i^*\}$
    - $D \leftarrow D \setminus \{i^*\}$
    - $h \leftarrow h_{\text{new}}$
  - **else**
  - **break**
- **end while**

**return** $I$

[12] can be simplified to
\[
\ln \left( \frac{M}{1 - M} \right) \leq \sum_{d_h=1}^{D_h} \delta_{v,d_h} \sigma \odot \left( \sum_{i \in I(k)} u_{d_h,i} \right) \tag{13}
\]
\[
= h(\alpha). \tag{14}
\]

We consider making $h(\alpha)$ as large as possible. We only need to choose the index set of altered pixels $I(k) \subset \{1, \ldots, D\}$ by assuming $\alpha_d \in \{0,R\}$ for any $d \in \{1, \ldots, D\}$; that is, we consider making
\[
h'(I(k)) := h(\alpha \in \{0,R\}^D) \tag{15}
\]
\[
= R \sum_{d_h=1}^{D_h} \delta_{v,d_h} \sigma \odot \left( \sum_{i \in I(k)} u_{d_h,i} \right) \tag{16}
\]
as large as possible rather than $h(\alpha)$. Here, $I(k)$ is determined by Algorithm 1. Algorithm 1 tries to maximize $h'(I(k))$ as large as possible by determining an element of $I$ sequentially. Although in this case, $I(k)$ does not maximize $h'(I(k))$, $h'(I(k))$ monotonically increases with respect to $k$. In conclusion, there exists a $k$-SFI with a confidence score of at least $M$ on $C_1$ if
\[
\ln \left( \frac{M}{1 - M} \right) \leq R \sum_{d_h=1}^{D_h} \delta_{v,d_h} \sigma \odot \left( \sum_{i \in I(k)} u_{d_h,i} \right). \tag{17}
\]

### 4.3 Kernel method

As a nonlinear model, we also consider a kernel method. A kernel method transforms the data in raw representation into feature vector representations, and classifies them in
high-dimension space. Kernel method computes a weighted sum of similarities

$$\sum_{n=1}^{N} s_{C,n} K(x_{C,n}, x'),$$

where $N$ is the number of data in a class, $s_{C}$ is the weight vector, $x_{C,n}$ is the $n$-th data in class $C \in \{C_1, C_2\}$, $x'$ is an unlabeled input, and $K$ is the kernel function. Also, by rearranging [18], the following equation is obtained and weight vector in feature space $t_{C}$ is defined:

$$\sum_{n=1}^{N} s_{C,n} K(x_{C,n}, x') = \left( \sum_{n=1}^{N} s_{C,n} \varphi(x_{C,n}) \right) ^{T} \varphi(x'),$$

$$=: t_{C} ^{T} \varphi(x'),$$

where $K(x_1, x_2) = \varphi(x_1) ^{T} \varphi(x_2)$ and $\varphi(.)$ is the feature vector. Here, let $K$ be homogeneous polynomial kernel (HPK) of degree two $K(x_1, x_2) = (x_1 ^{T} x_2)^2$. Let SFI be $z = z_0 1 + \alpha \in [0, R]^D$. The conditions on the weight vector in feature space $T = [t_1, t_2] ^{T}$ such that confidence score $p_1(z) \geq M$ are as follows:

**Theorem 3.** Let $S = [s_1, s_2] ^{T} \in \mathbb{R}^{2 \times N}$ be the weight matrix of a kernel method. $T = [t_1, t_2] ^{T}$ is defined as (20).

We assume HPK of degree two and

$$\phi(x) = (x_1 ^{2}, \ldots, x_1 x_2, \sqrt{2} x_1 x_3, \ldots, \sqrt{2} x_D, x_D).$$

(21)

Let $\delta_t = (\delta_{t,1}, \ldots, \delta_{t,D}) ^{T} = t_1 - t_2$. For any $T$ and $M \in [0, 1]$, there exists an $k$-SFI $z \in [0, R]^D$ with a confidence score of at least $M$ on $C_1$ if

$$\ln \left( \frac{M}{1 - M} \right) \leq R^2 \sum_{i \in I(k)} \left( \delta_{t, i} + \sqrt{2} \sum_{j \in I(I(k) \cup \{1, \ldots, i\})} \delta_{t, D+i+j-2} \right),$$

(22)

where $I(k) \subset \{1, \ldots, D\}$ is determined by Algorithm 2.

**Proof.** Let us denote an SFI by $z = z_0 1 + \alpha \in [0, R]^D$. The confidence score of the SFI on $C_1$ is as follows:

$$p_1(z) = \frac{\exp(t_1 \alpha)}{\exp(t_1 \alpha) + \exp(t_2 \alpha)} \geq M.$$  

(23)

Thus, there exists an SFI with $z_0 = 0$ and $p_1(z) \geq M$ when the following holds:

$$\frac{\exp(t_1 \alpha)}{\exp(t_1 \alpha) + \exp(t_2 \alpha)} \geq M.$$  

(24)

can be simplified to

$$\ln \left( \frac{M}{1 - M} \right) \leq \sum_{i=1}^{D} \alpha_i \left( \delta_{t, i} \alpha_i + \sqrt{2} \sum_{j=i+1}^{D} \delta_{t, D+i+j-2} \alpha_j \right)$$

(25)

$$=: g(\alpha).$$

(26)

We consider making $g(\alpha)$ as large as possible. We only need to choose the index set of altered pixels $I(k) \subset \{1, \ldots, D\}$

**Algorithm 2** Algorithm for determination of $I$ on a kernel method

**Input:** $k \in \mathbb{N}$, $\delta_t \in \mathbb{R}^{D}$

**Output:** $I$

$I \leftarrow \emptyset$

$D \leftarrow \{1, \ldots, D\}$

$h \leftarrow 0$

while $|I| < n$

$i^* \leftarrow \arg \max_{i \in D} \sum_{j \in I \cup \{i\}} (\delta_{t, j} + \sqrt{2} \sum_{k \in I \cup \{i\} \setminus \{1, \ldots, j\}} \delta_{t, D+j+k-2})$

$h_{\text{new}} \leftarrow \sum_{j \in I \cup \{i^*\}} (\delta_{t, j} + \sqrt{2} \sum_{k \in I \cup \{i^*\} \setminus \{1, \ldots, j\}} \delta_{t, D+j+k-2})$

if $h_{\text{new}} > h$

$I \leftarrow I \cup \{i^*\}$

$D \leftarrow D \setminus \{i^*\}$

$h \leftarrow h_{\text{new}}$

else

break

endif

end while

return $I$

by assuming $\alpha_d \in \{0, R\}$ for any $d \in \{1, \ldots, D\}$; that is, we consider making

$$g'(I(k)) := g(\alpha \in [0, R]^D)$$

(27)

$$= R^2 \sum_{i \in I(k)} \left( \delta_{t, i} + \sqrt{2} \sum_{j \in I(I(k) \cup \{1, \ldots, i\})} \delta_{t, D+i+j-2} \right)$$

(28)

as large as possible rather than $g(\alpha)$. Here, $I(k)$ is determined by Algorithm 2. Algorithm 2 tries to maximize $h'(I(k))$ as large as possible by determining an element of $I$ sequentially. Although in this case, $I(k)$ does not maximize $g'(I(k))$, $g'(I(k))$ monotonically increases with respect to $k$. In conclusion, there exists an $k$-SFI with a confidence score of at least $M$ on $C_1$ if

$$\ln \left( \frac{M}{1 - M} \right) \leq R^2 \sum_{i \in I(k)} \left( \delta_{t, i} + \sqrt{2} \sum_{j \in I(I(k) \cup \{1, \ldots, i\})} \delta_{t, D+i+j-2} \right).$$

(29)

□

### 4.4 The effect of the number of altered pixels

We discuss the effect of the number of altered pixels, $k$, in k-SFI attacks. The left-hand side of inequalities (2) and (3) increase at $O(\ln(M))$ for $M \in [0, 1)$, whereas the right-hand side increases at $O(k)$ for $k \ll D$. Therefore, (2) and (3) are easily satisfied for a high confidence score $M$ by increasing $k$ slightly. In the one-hidden-layer neural network and a kernel method, we can obtain $O(Dh_k)$ and $O(k^2)$, respectively, as the effect of the number of altered pixels.

**Proof.** In linear function (cf. Theorem 1), if $\delta_w \mathbf{1} > 0$ and
\(|\{ i \mid \delta_{w,i} < 0 \} \gg k\), from \(^2\), we obtain
\begin{align}
R\delta^T_w \mathbf{1} + (r - R) \sum_{i \in I \setminus \{k\}} \delta_{w,i} & \geq R\delta^T_w \mathbf{1} + k(r - R) \Delta_w \\
& = O(k),
\end{align}
where \(\Delta_w = \max_{i \in I \setminus \{k\}} \delta_{w,i}\). Also, if \(\delta^T_w \mathbf{1} \leq 0\) and \(|\{ i \mid \delta_{w,i} > 0 \} \gg k\), from \(^3\), we obtain
\begin{align}
r\delta^T_w \mathbf{1} + (R - r) \sum_{i \in I \setminus \{k\}} \delta_{w,i} & \geq r\delta^T_w \mathbf{1} + k(R - r) \Delta_w \\
& = O(k),
\end{align}
where \(\Delta_w = \min_{i \in I \setminus \{k\}} \delta_{w,i}\). Therefore, the vulnerability of a linear function against \(-SFIs increases at \(O(k)\).

\textbf{Proof.} In one-hidden-layer neural network (cf. Theorem 2), if all entries of \(\delta_w\) and \(u_{d,\alpha,i}\) are positive, where \(d_h \in \{1, \ldots, D_h\}\) and \(i \in I\), from \(^10\), we obtain
\begin{align}
R \sum_{d_h=1}^{D_h} \delta_{c,d_h} \sigma \left( \sum_{i \in I} u_{d,\alpha,i} \right) & \geq RD_h k \Delta_v \mu \\
& = O(D_h k),
\end{align}
where \(\Delta_v = \min_{d_h \in \{1, \ldots, D_h\}} \delta_{c,d_h}\) and \(\mu = \min_{d_h \in \{1, \ldots, D_h\}, i \in I} u_{d,\alpha,i}\). Therefore, the vulnerability of a one-hidden-layer neural network against \(-SFIs increases at \(O(D_h k)\).

\textbf{Proof.} In the kernel method with HPK of degree 2 (cf. Theorem 3), if all entries of \(\delta_t\) are positive, from \(^22\), we obtain
\begin{align}
R^2 \sum_{i \in I \setminus \{k\}} \left( \delta_{t,i} + \sqrt{2} \sum_{j \in I \setminus \{1, \ldots, i\}} \delta_{t,j,D+i+j-2} \right) & \geq \frac{\sqrt{3}R^2k^2 \Delta_t}{2} \\
& = O(k^2),
\end{align}
where \(\Delta_t = \min_{i \in I \setminus \{k\}, j \in I \setminus \{1, \ldots, i\}} \delta_{t,D+i+j-2}\). Therefore, the vulnerability of a kernel method against \(-SFIs increases at \(O(k^2)\).

These results indicate that complex models are more likely to be vulnerable to \(-SFIs. Although the analysis here is restricted to simple linear and nonlinear models, our results on these models theoretically support the empirical observations described in the Section 2 in that \(-SFIs can be found for various classes and datasets.

\section{5 Results}

\subsection{5.1 Detailed settings}
Here, we describe the experimental settings. Table 1 shows the top-1 accuracy of each architecture. The architectures on MNIST and FashionMNIST are trained using Adam with learning rate=0.001. Those on CIFAR10 are trained using stochastic gradient descent (SGD) with learning rate=0.01, momentum=0.9, and weight decay=5 \times 10^{-4}. That on ImageNet is the Pytorch pretrained model. Figure 3 shows detailed architectures of MLP and CNN. Dropout layers of MLP are used only during training.

\begin{table}[h]
\centering
\caption{Top-1 accuracy (\%) on natural test dataset.}
\begin{tabular}{lccc}
\hline
\textbf{dataset} & \textbf{architecture} & \textbf{accuracy (top-1)} \\
\hline
MNIST & MLP & 97 \\
CNN & 98 \\
CapsNet & 98 \\
FashionMNIST & MLP & 86 \\
CNN & 88 \\
CapsNet & 89 \\
CIFAR10 & VGG16 & 80 \\
NiN & 84 \\
ResNet50 & 80 \\
ImageNet & ResNet50 & 76 \\
\hline
\end{tabular}
\end{table}

Figure 3: MLP and CNN architectures. FC and the number in parentheses refer to a fully connected layer and output dimensions, respectively. DO and the number in parentheses refer to a dropout layer and the probability of dropping neurons, respectively. CONV and the number in parentheses refer to a convolutional layer (kernel size=3 \times 3, stride=1) with zero-padding of 1 pixel and output channels, respectively. MP refers to a max pooling layer.

\subsection{5.2 SFIs generated by DE}

\textbf{CIFAR10.} We show that our DE-based method (cf. Section 3.1) can generate \(-SFIs (k = 1, 5, 25, 125)\) that can fool the VGG16, NiN, and ResNet50 models trained on CIFAR10. The number of candidate solutions \(s\) and iterations of \(DE T\) were set to 500 and 3,000, respectively. To evaluate the effect of the base color in \(-SFIs, we performed SFI generation with six base colors (black, gray, white, red, green, and blue). Figure 7 shows a few examples of \(-SFIs generated by DE. In the top left, the plane image classified by the DNN with a high confidence score (99\%) cannot be recognized by humans as a plane. As shown in Tables 2, the \(-SFIs generated by the proposed method lead to high confidence scores through the three models over various target classes and base colors.

The confidence scores of a single-color image are not uniform over classes, contrary to intuition. Interestingly, the few pixels that are altered in the base color to generate
an SFI have a significant impact on the confidence score through the model. For example, Table 3 shows a black image is given a 0% confidence score by the model on the plane class; however, after only five pixel modifications (SFI at the top left of Figure 4), the confidence score of the image on the plane class increases to 99%. The extreme increase in confidence scores by a mere five-pixel change is seen for various base colors and target classes (e.g., gray and plane class, and white and cat class).

Confidence scores tend to be higher for larger $k$. Figure 5 shows a few examples of SFIs targeting the plane class. Although changing many pixels can produce class-specific features, in this experiment, 25-SFIs and 125-SFIs do not have remarkable patterns, and both yield high confidence scores. However, some base colors cannot attack the models well regardless of $k$. For example, the confidence scores of blue-based $k$-SFIs ($k = 1, 5, 25, 125$) generated for VGG16 and targeting the car class are 0% (Tables 2–5). We consider the reason why even 125-SFIs cannot fool the models well is because of the incomplete optimization of DE. Our DE-based method takes only a simple calculation, thus it cannot explore the large solution space well. Therefore, it might be hard for the DE-based method to optimize a lot of $k$ pixels to be altered. The difference between the base colors that can attack the model and those that cannot is discussed in Section 5.3.

**ImageNet.** We also generate 5-SFIs by DE ($s=500$, $T=500$) for ResNet50 trained on ImageNet. The number of iterations of DE, $T$, is set to a lower value than that of the CIFAR10 experiment and the confidence scores were obtained by one SFI because of high computational cost. Figure 6 shows almost all SFIs cannot attack the ResNet50 model. In particular, SFIs targeting the class 151–287, dog and cat classes, tend to fail in fooling the model. This might be because of the softmax function. For example, when SFIs target the golden retriever class and try to increase the confidence score of this class, SFIs cannot help increasing the confidence scores of other dog classes such as Brittany spaniel, collie, and dalmatian. Then, outputs corresponding to all dog classes increase, which prohibits the softmax function from giving a high confidence score only to the golden retriever class.

With more iterations of DE (large $T$), one might be able to yield stronger SFIs that succeed even on ImageNet. We found the gray-based 10-SFI classified as a centipede with a 91% confidence score through ResNet50 by DE ($s = 1,000$, $T = 30,000$) (Figure 1). Also, we note that we used only 5 pixels for SFIs due to the computational cost. Since 5 pixels is approximately 0.01% ($5/(224 \times 224)$) on the whole image, SFIs generation on ImageNet is much more difficult than CIFAR10 (5 pixels is approximately 0.5% on the whole image). This prevents SFIs from attacking on ImageNet well. We consider that more altered pixels make DNNs more vulnerable on even ImageNet.

**Figure 4:** 5-SFIs generated by DE. Class name and confidence scores at the bottom of the SFIs are obtained through VGG16 \[9\] trained on CIFAR10.

**Figure 5:** SFIs generated by DE and targeting the plane class. The confidence scores at the bottom of the SFIs are obtained through each model trained on CIFAR10.
Table 2: Average confidence scores (%) of 1-SFIs through VGG16. Rows denote base colors, and columns denote target classes. That is, the first row and column are the average confidence scores of 50 SFIs targeted to the plane class and are based on a black image. The values in parentheses are the confidence scores of single-color images, such as all black images. The bold text represents values greater than or equal to 90.

|       | plane | car  | bird | cat  | deer | dog  | frog | horse | ship  | truck |
|-------|-------|------|------|------|------|------|------|-------|-------|-------|
| black | 87 (0)| 7 (0)| **99** (62)| 99 (0)| 37 (0)| 98 (35)| 79 (0)| 59 (0)| **94** (0)|       |
| gray  | 99 (11)| 0 (0)| 92 (2)| 63 (3)| 7 (0)| 0 (0)| 17 (0)| 0 (0)| **99** (81)| 5 (0) |
| white | **99** (9)| 1 (0)| 99 (5)| 87 (0)| 11 (0)| 0 (0)| 17 (0)| 9 (0)| **99** (84)| 44 (0)|
| red   | 18 (2)| 6 (3)| 2 (0)| 70 (50)| 1 (0)| 0 (0)| 12 (3)| 0 (0)| 48 (20)| 32 (17)|
| blue  | **99** (95)| 0 (0)| 22 (2)| 0 (0)| 0 (0)| 0 (0)| 0 (0)| 24 (1)| 0 (0) |       |

Table 3: Average confidence scores (%) of 5-SFIs through VGG16. The description of the table are the same as Table 2.

|       | plane | car  | bird | cat  | deer | dog  | frog | horse | ship  | truck |
|-------|-------|------|------|------|------|------|------|-------|-------|-------|
| black | **99** (0)| 99 (0)| 99 (0)| 99 (62)| 99 (0)| 99 (0)| 99 (35)| 99 (0)| 99 (0)| 99 (0) |
| gray  | 100 (11)| 52 (0)| 99 (2)| 99 (3)| 99 (0)| 95 (0)| 99 (0)| 99 (0)| 99 (81)| 97 (0) |
| white | 100 (9)| 33 (0)| 100 (5)| 99 (0)| 99 (0)| 99 (0)| 98 (0)| 99 (0)| 99 (84)| 99 (0) |
| red   | **99** (97)| 53 (0)| 56 (0)| **98** (0)| 4 (0)| 0 (0)| 1 (0)| 17 (0)| **97** (0)| 91 (0) |
| green | 99 (2)| 29 (3)| 74 (0)| **96** (50)| 52 (0)| 3 (0)| **90** (3)| 5 (0)| **92** (20)| 87 (17) |
| blue  | **99** (95)| 0 (0)| **98** (2)| 37 (0)| 4 (0)| 0 (0)| 0 (0)| 0 (0)| **98** (1)| 0 (0) |

Table 4: Average confidence scores (%) of 25-SFIs through VGG16. The description of the table are the same as Table 2.

|       | plane | car  | bird | cat  | deer | dog  | frog | horse | ship  | truck |
|-------|-------|------|------|------|------|------|------|-------|-------|-------|
| black | **99** (0)| 99 (0)| 99 (0)| 99 (62)| 99 (0)| 99 (0)| 99 (35)| 99 (0)| 99 (0)| 99 (0) |
| gray  | 100 (11)| 66 (0)| 99 (2)| 99 (3)| 99 (0)| 97 (0)| 99 (0)| 97 (0)| 99 (81)| 98 (0) |
| white | 100 (9)| 90 (0)| 99 (5)| 99 (0)| 99 (0)| 99 (0)| 98 (0)| 99 (0)| 99 (84)| 99 (0) |
| red   | **99** (97)| 77 (0)| 52 (0)| **99** (0)| 2 (0)| 1 (0)| 4 (0)| 32 (0)| 89 (0)| 85 (0) |
| green | 96 (2)| 26 (3)| 31 (0)| **97** (50)| 21 (0)| 0 (0)| **96** (3)| 4 (0)| 84 (20)| 74 (17) |
| blue  | **99** (95)| 0 (0)| **99** (2)| 66 (0)| 5 (0)| 0 (0)| 1 (0)| 0 (0)| **99** (1)| 0 (0) |

Table 5: Average confidence scores (%) of 125-SFIs through VGG16. The description of the table are the same as Table 2.

|       | plane | car  | bird | cat  | deer | dog  | frog | horse | ship  | truck |
|-------|-------|------|------|------|------|------|------|-------|-------|-------|
| black | **99** (0)| 99 (0)| 99 (0)| 99 (62)| 99 (0)| 99 (0)| 99 (35)| 99 (0)| 99 (0)| 99 (0) |
| gray  | 99 (11)| 81 (0)| 99 (2)| 99 (3)| 99 (0)| 99 (0)| 99 (0)| 99 (0)| 99 (81)| 99 (0) |
| white | **99** (9)| 98 (0)| **99** (5)| 99 (0)| 99 (0)| 99 (0)| 99 (0)| 99 (0)| 99 (84)| 99 (0) |
| red   | **99** (97)| 98 (0)| **97** (0)| 99 (0)| 77 (0)| 25 (0)| **92** (0)| 79 (0)| **94** (0)| 99 (0) |
| green | 98 (2)| 30 (3)| 65 (0)| **98** (50)| 73 (0)| 0 (0)| **99** (3)| 12 (0)| 85 (20)| 87 (17) |
| blue  | **99** (95)| 0 (0)| **99** (2)| **90** (0)| 7 (0)| 0 (0)| 3 (0)| 0 (0)| **99** (1)| 0 (0) |
### Table 6: Average confidence scores (%) of 1-SFIs through NiN [33]. The description of the table are the same as Table 2.

|          | plane | car  | bird | cat  | deer | dog  | frog | horse | ship | truck |
|----------|-------|------|------|------|------|------|------|-------|------|-------|
| black    | 75 (7)| 40 (1)| 62 (25)| 83 (33)| 38 (22)| 4 (2)| 3 (2)| 5 (0)| 8 (3)| 3 (0) |
| gray     | 93 (35)| 11 (1)| 56 (12)| 11 (3)| 10 (5)| 2 (0)| 3 (2)| 0 (0)| 42 (36)| 3 (0) |
| white    | 96 (27)| 2 (0)| 85 (35)| 23 (6)| 25 (13)| 1 (0)| 4 (2)| 1 (0)| 26 (12)| 0 (0) |
| red      | 67 (5)| 68 (26)| 6 (1)| 44 (28)| 20 (11)| 14 (9)| 2 (1)| 17 (7)| 2 (1)| 27 (6) |
| green    | 63 (11)| 2 (1)| 82 (23)| 7 (3)| 53 (30)| 0 (0)| 42 (36)| 3 (0)| 18 (9)| 0 (0) |
| blue     | 95 (20)| 4 (2)| 9 (2)| 13 (5)| 23 (14)| 1 (0)| 6 (2)| 0 (0)| 63 (50)| 1 (1) |

### Table 7: Average confidence scores (%) of 5-SFIs through NiN. The description of the table are the same as Table 2.

|          | plane | car  | bird | cat  | deer | dog  | frog | horse | ship | truck |
|----------|-------|------|------|------|------|------|------|-------|------|-------|
| black    | 99 (7)| 98 (1)| 99 (25)| 99 (33)| 51 (22)| 98 (2)| 95 (2)| 95 (0)| 90 (3)| 81 (0) |
| gray     | 99 (35)| 85 (1)| 99 (12)| 69 (3)| 41 (5)| 98 (0)| 20 (2)| 17 (0)| 97 (36)| 44 (0) |
| white    | 100 (27)| 36 (0)| 99 (35)| 95 (6)| 72 (13)| 99 (0)| 18 (2)| 23 (0)| 98 (12)| 4 (0) |
| red      | 99 (5)| 97 (26)| 74 (1)| 88 (28)| 68 (11)| 52 (9)| 15 (1)| 97 (7)| 17 (1)| 93 (6) |
| green    | 99 (11)| 16 (1)| 99 (23)| 24 (3)| 88 (30)| 0 (0)| 84 (19)| 2 (0)| 53 (9)| 4 (0) |
| blue     | 99 (20)| 27 (2)| 64 (2)| 83 (5)| 60 (14)| 12 (0)| 40 (2)| 5 (0)| 98 (50)| 27 (1) |

### Table 8: Average confidence scores (%) of 25-SFIs through NiN. The description of the table are the same as Table 2.

|          | plane | car  | bird | cat  | deer | dog  | frog | horse | ship | truck |
|----------|-------|------|------|------|------|------|------|-------|------|-------|
| black    | 100 (7)| 99 (1)| 99 (25)| 99 (33)| 95 (22)| 98 (2)| 95 (2)| 95 (0)| 99 (3)| 99 (0) |
| gray     | 100 (35)| 97 (1)| 99 (12)| 95 (3)| 86 (5)| 98 (0)| 87 (2)| 59 (0)| 98 (36)| 77 (0) |
| white    | 100 (27)| 97 (0)| 99 (35)| 99 (6)| 90 (13)| 99 (0)| 51 (2)| 98 (0)| 99 (12)| 58 (0) |
| red      | 99 (5)| 99 (26)| 68 (1)| 93 (28)| 81 (11)| 63 (9)| 22 (1)| 70 (7)| 98 (12)| 4 (0) |
| green    | 99 (11)| 27 (1)| 99 (23)| 39 (3)| 88 (30)| 0 (0)| 99 (19)| 2 (0)| 29 (9)| 2 (0) |
| blue     | 99 (20)| 55 (2)| 83 (2)| 96 (5)| 77 (14)| 24 (0)| 39 (2)| 18 (0)| 97 (50)| 42 (1) |

### Table 9: Average confidence scores (%) of 125-SFIs through NiN. The description of the table are the same as Table 2.

|          | plane | car  | bird | cat  | deer | dog  | frog | horse | ship | truck |
|----------|-------|------|------|------|------|------|------|-------|------|-------|
| black    | 99 (7)| 99 (1)| 99 (25)| 99 (33)| 99 (22)| 95 (2)| 99 (2)| 93 (0)| 99 (3)| 99 (0) |
| gray     | 99 (35)| 99 (1)| 99 (12)| 99 (3)| 99 (5)| 98 (0)| 99 (2)| 82 (0)| 99 (36)| 96 (0) |
| white    | 100 (27)| 99 (0)| 99 (35)| 99 (6)| 99 (13)| 99 (0)| 99 (2)| 96 (0)| 99 (12)| 99 (0) |
| red      | 99 (5)| 99 (26)| 96 (1)| 99 (28)| 89 (11)| 77 (9)| 86 (1)| 94 (7)| 94 (1)| 99 (6) |
| green    | 98 (11)| 47 (1)| 99 (23)| 74 (3)| 99 (30)| 12 (0)| 99 (19)| 8 (0)| 99 (24)| 9 (0) |
| blue     | 99 (20)| 99 (2)| 98 (2)| 99 (5)| 88 (14)| 55 (0)| 70 (2)| 41 (0)| 99 (50)| 99 (1) |
Table 10: Average confidence scores (%) of 1-SFIs through ResNet50. The description of the table are the same as Table 2.

|       | plane | car  | bird | cat  | deer | dog  | frog | horse | ship  | truck |
|-------|-------|------|------|------|------|------|------|-------|-------|-------|
| black | 23 (8)| 0 (0)| 16 (5)| 56 (13)| 83 (65)| 0 (0) | 1 (0) | 2 (0) | 12 (5) | 0 (0) |
| gray  | 99 (11)| 0 (0)| 16 (1)| 5 (0) | 16 (3) | 4 (0) | 2 (0) | 0 (0) | 96 (81)| 0 (0) |
| white | 90 (71)| 0 (0)| 22 (13)| 23 (11)| 0 (0) | 0 (0) | 0 (0) | 4 (2) | 0 (0) | 0 (0) |
| red   | 11 (8)| 18 (14)| 5 (3)| 67 (62)| 0 (0) | 1 (0) | 0 (0) | 0 (0) | 10 (8) | 0 (0) |
| green | 99 (99)| 0 (0)| 0 (0) | 0 (0) | 0 (0) | 0 (0) | 0 (0) | 0 (0) | 12 (5) | 0 (0) |
| blue  | 90 (65)| 0 (0)| 34 (15)| 3 (1)| 11 (6) | 0 (0) | 2 (0) | 0 (0) | 17 (9) | 0 (0) |

Table 11: Average confidence scores (%) of 5-SFIs through ResNet50. The description of the table are the same as Table 2.

|       | plane | car  | bird | cat  | deer | dog  | frog | horse | ship  | truck |
|-------|-------|------|------|------|------|------|------|-------|-------|-------|
| black | 99 (8)| 0 (0)| 88 (5)| 97 (13)| 96 (65)| 0 (0)| 11 (0)| 67 (0)| 34 (5) | 8 (0) |
| gray  | 99 (11)| 2 (0)| 99 (1)| 81 (0)| 97 (0)| 86 (0)| 0 (0)| 99 (81)| 34 (0) | 0 (0) |
| white | 99 (71)| 1 (0)| 99 (13)| 90 (11)| 10 (0)| 6 (0)| 41 (0)| 0 (0)| 70 (2) | 7 (0) |
| red   | 18 (8)| 70 (14)| 9 (3)| 72 (62)| 0 (0)| 1 (0)| 0 (0)| 0 (0)| 0 (0) | 14 (8) |
| green | 99 (99)| 0 (0)| 0 (0) | 0 (0) | 0 (0)| 0 (0)| 4 (0)| 0 (0) | 0 (0) | 0 (0) |
| blue  | 99 (65)| 0 (0)| 94 (15)| 15 (1)| 18 (6)| 0 (0)| 9 (0)| 0 (0)| 53 (9) | 0 (0) |

Table 12: Average confidence scores (%) of 25-SFIs through ResNet50. The description of the table are the same as Table 2.

|       | plane | car  | bird | cat  | deer | dog  | frog | horse | ship  | truck |
|-------|-------|------|------|------|------|------|------|-------|-------|-------|
| black | 99 (8)| 33 (0)| 99 (5)| 99 (13)| 99 (65)| 4 (0)| 84 (0)| 99 (0)| 77 (5) | 93 (0) |
| gray  | 99 (11)| 38 (0)| 99 (1)| 98 (0)| 99 (3)| 99 (0)| 99 (0)| 6 (0)| 99 (81)| 36 (0) |
| white | 99 (71)| 6 (0)| 99 (13)| 99 (11)| 96 (0)| 49 (0)| 99 (0)| 9 (0)| 89 (2)| 19 (0) |
| red   | 27 (8)| 84 (14)| 11 (3)| 77 (62)| 0 (0)| 1 (0)| 0 (0)| 0 (0)| 1 (0) | 18 (8) |
| green | 99 (99)| 0 (0)| 0 (0) | 0 (0)| 0 (0)| 0 (0)| 18 (0)| 0 (0)| 0 (0) | 0 (0) |
| blue  | 99 (65)| 0 (0)| 96 (15)| 49 (1)| 33 (6)| 1 (0)| 26 (0)| 1 (0)| 63 (9) | 0 (0) |

Table 13: Average confidence scores (%) of 125-SFIs through ResNet50. The description of the table are the same as Table 2.

|       | plane | car  | bird | cat  | deer | dog  | frog | horse | ship  | truck |
|-------|-------|------|------|------|------|------|------|-------|-------|-------|
| black | 99 (8)| 99 (0)| 99 (5)| 99 (13)| 99 (65)| 99 (0)| 99 (0)| 98 (0)| 95 (5) | 99 (0) |
| gray  | 99 (11)| 98 (0)| 99 (1)| 99 (0)| 99 (3)| 99 (0)| 99 (0)| 89 (0)| 99 (81)| 87 (0) |
| white | 99 (71)| 59 (0)| 99 (13)| 99 (11)| 99 (0)| 99 (0)| 100 (0)| 98 (0)| 94 (2) | 32 (0) |
| red   | 99 (8)| 92 (14)| 24 (3)| 82 (62)| 2 (0)| 4 (0)| 2 (0)| 0 (0)| 1 (0) | 65 (8) |
| green | 99 (99)| 44 (0)| 0 (0)| 0 (0)| 1 (0)| 0 (0)| 97 (0)| 0 (0)| 0 (0) | 0 (0) |
| blue  | 100 (65)| 9 (0)| 99 (15)| 96 (1)| 81 (6)| 25 (0)| 96 (0)| 43 (0)| 80 (9) | 12 (0) |
5.3 Features of SFIs in the deep layer

To understand why changing only a few pixels in a single-color image can significantly alter the confidence scores through a model, we compared the deep features of natural images and those of five black-based SFIs generated by DE. Specifically, we compared the output at the second-to-last layer of the VGG16 model trained on CIFAR10 for natural images and SFIs. Deep features are projected into a two-dimensional space through a principal component analysis (PCA) (Figure 7). It can be seen that within the same class (e.g., airplane class), both natural images (blue dots) and SFIs (blue crosses) are distributed nearby. This result shows that natural images and SFIs, which appear to be completely different in the input space, become extremely similar after passing through the layers of the model. This result also indicates the difference between a base color that can attack (black triangle) or cannot attack (blue triangle) (cf. Section 5.2) does not depend on the position of the deep features of the base color. The difference will be studied in a future study; however, an improved DE [34,35] may mitigate the effect of the base color because the difference might be attributed to a defective optimization.

We emphasize that this result is not trivial; that is, in the feature space, SFIs could be distributed far from natural images and classified as the target class with high confidence scores. In fact, some OOD methods take advantage of the differences in the distribution of deep features of the in-distribution objects and OOD objects of the same class [36,37]. The PCA projection results show that natural images and SFIs are distributed near the feature space. Therefore, they cannot be discriminated from each other by such OOD methods.

5.4 SFIs generated by deep feature inversion

Natural images and SFIs that are given the same label by a model are distributed near the feature space. This observation implies that, given a natural image, we can generate an SFI that is extremely similar to the given image in the feature space, but not in the input space. Herein, we demonstrate that our second SFI generation method (cf. Section 3.2) realizes this. Generating SFIs, we set $\lambda_{\text{sparse}}$ to $1.0 \times 10^{-5}$ and $z_b$ to 0 (i.e., black-based SFIs) in (1).

We used the deep features in the second to last layer of the VGG16 model trained on CIFAR10. A few examples of SFIs generated by deep feature inversion are shown in Figure 8. The corresponding SFI can be generated for a given natural image. Interestingly, some colors of the altered pixels of the SFIs generated by deep feature inversion are not included in the original image. For example, the SFI generated from the yellow car image in the second column from the left in Figure 8 has red and blue pixels that do not appear to be
5.5 Comparison between DE-based and deep feature inversion methods

The distribution of the modified pixel positions in the SFIs generated by DE and deep feature inversion are compared in Figure 10. Some classes (car, bird, deer, dog, horse, and ship) seem to have inclusion relations, and the other classes (plane, cat, frog, and truck) do not seem to match at all through VGG16. Figure 9 shows the original natural car images and corresponding SFIs. Deep feature inversion can generate blue-based SFIs targeting the car class. This is because deep feature inversion in a image-targeted setting can treat many altered pixels more efficiently than DE in a class-targeted setting. It is left for future work whether SFIs with only five altered pixels can fool the DNN or not.

Figure 9: Original natural images and corresponding SFIs generated by deep feature inversion. The inner product is obtained from deep features of an SFI and those of the original natural image. The confidence scores are obtained by SFIs through the VGG16 model.

included in the original image.

Blue-based SFIs generated by DE can attack few classes of CIFAR10 through VGG16 model (Table 5). To understand the difference between our two methods, we generate blue-based SFIs by deep feature inversion. In particular, we target the car class that SFIs generated by DE cannot fool at all through VGG16. Figure 9 shows the original natural car images and corresponding SFIs. Deep feature inversion can generate blue-based SFIs targeting the car class. This is because deep feature inversion in a image-targeted setting can treat many altered pixels more efficiently than DE in a class-targeted setting. It is left for future work whether SFIs with only five altered pixels can fool the DNN or not.

5.6 Comparison of four architectures

We now discuss why SFIs can attack DNNs successfully by comparing four different architectures: MLP, CNNMP, CNNNone, and CapsNet. MLP consists of three fully connected layers with ReLU activation (Figure 3a)), CNNMP consists of three convolutional layers and max pooling layers followed by two fully connected layers (Figure 3b)), CNNNone is identical to CNNMP except for no max pooling layers, and CapsNet is that used in [11]. The four models were trained on MNIST and FashionMNIST, and their robustness was evaluated by the success rate of attack by the 5-SFIs generated using DE (the number of candidate solutions s is set to 500, and the iteration time T is set to 1,000). Among the ten SFIs generated for each model and target class, the one with the highest confidence score is shown in Tables 14 and 19. Although all the models were fooled by the SFIs, CNNMP was particularly vulnerable to attack.

We consider the nature of the max pooling to be the cause of the vulnerability of CNNMP. CNNMP first captures low-level features with a convolutional layer and a max pooling layer, and then applies successive layers. Finally, CNNMP fires the neurons that represent high-level features by connecting low-level features. However, the high-level features captured by CNNMP do not contain much spatial information of the low-level features because of max pooling. To classify objects with incomplete spatial information, CNNMP makes decisions based on whether lower-level features exist rather than how they are connected. The inappropriate processing of spatial information in CNNMP can also be responsible for the vulnerability of CNNMP to SFIs. Different pixels in the SFI can induce some neurons to fire through the convolution layers. The way the neurons fire by SFIs are patterns that would not be possible in natural images, but max pooling causes the CNNMP to map these inappropriate feature patterns to regions that are similar to those in natural images.

MLP, CNNNone, and CapsNet, by contrast, connect low-level features to higher-level features without discarding the spatial information of the lower-level features. Their ability to represent the spatial connectivity between features more richly than CNNMP may require SFIs to fire neurons more appropriately, thus making it harder for SFIs to successfully attack. Although these robust architectures show a certain resistance against SFIs, it is difficult for MLP and CNNNone to obtain a high image classification accuracy; in addition, the accuracy of CapsNet for large datasets such as ImageNet has yet to be studied owing to its high computational cost. To realize learning with a high classification accuracy and robustness against SFIs at the same time, a new DNN architecture that can handle large datasets without a loss of spatial information is required.

5.7 SFI detection

We also considered the defense of DNNs against SFIs. In particular, we used OOD detection for defense purposes. A typical method of OOD detection uses the outputs of the deep layers of the model [36, 37]. However, these methods cannot be used because, as shown in Section 5.3, natural images and SFIs share similar distributions; thus, it is difficult to distinguish them by using the dissociation of the distributions between natural images and SFIs. Here, starting from a VGG16 model M0 trained using the CIFAR10 dataset, we repeatedly generated SFIs and retrained the model by incor-
Figure 10: Distribution of changed pixels (intensity > 0.3, where the range is [0, 1]) in SFIs generated by each method, represented as heat maps. Black, red, and white values indicate increases in that order.

Table 14: Comparison of four architectures trained on MNIST. Values in the table are the highest confidence scores (%) obtained by assessing ten SFIs generated by DE through the models. The three values represent differences in the base color and are in order of black, gray, and white. The accuracy and confidence scores next to the models are for the natural test dataset. Bold text represents values equal to or greater than average confidence score of natural dataset.

| model (accuracy, confidence) | 0          | 1          | 2          | 3          | 4          |
|-----------------------------|------------|------------|------------|------------|------------|
| MLP (98, 99)                | 69/99/100  | 100/99/99  | 55/99/0    | 13/99/1    | 91/99/0    |
| CNN_MP (99, 99)             | 100/100/100| 100/100/100| 100/100/100| 100/100/100| 100/100/100|
| CNNNone (98, 99)            | 100/0/0    | 100/0/0    | 100/100/100| 100/0/0    | 100/0/0    |
| CapsNet (99, 20)            | 12/16/13   | 18/12/12   | 17/19/11   | 17/18/11   | 17/18/15   |
| MLP (98, 99)                | 100/97/0   | 99/90/0    | 99/99/0    | 50/93/68   | 97/97/0    |
| CNN_MP (99, 99)             | 100/100/100| 100/100/100| 100/100/100| 99/99/100  | 100/100/85 |
| CNNNone (98, 99)            | 100/0/0    | 100/0/0    | 100/0/0    | 100/1/99   | 100/0/0    |
| CapsNet (99, 20)            | 21/18/9    | 12/16/12   | 18/18/12   | 14/20/12   | 14/17/14   |

The model was trained to classify the SFIs into an outlier class. In the case of the CIFAR10 dataset, the outlier class becomes the 11th class. In the experiment, the following procedures were repeated for \( i = 0, \ldots, 4 \).

1. 6,000 SFIs based on black (denoted by \( S_i \)) are generated by DE for model \( M_i \); 600 SFIs are generated for each class, and the number of changed pixels \( k \) is sampled randomly from \([1, 100]\). It should be noted that SFIs are generated by targeting one of the 10 classes, and the retrained model is expected to classify them into the 11th class (outlier class). The number of candidate solutions \( s \) was set to 500, and the number of iterations \( T \) was set to 3,000. When the confidence scores of a candidate solution in the group are higher than or equal to 90% during DE, the SFI is determined by the candidate solution at that time (early stopping).

2. Model \( M_{i+1} \) is trained using CIFAR10 and 5,000 SFIs of \( S_j, (j = 0, \ldots, i) \). Unused 1,000 SFIs of \( S_j \) will be used for test.

This approach results in an imbalance in the number of data points between the SFI dataset and each class of CIFAR10 (much larger number of SFIs). This imbalance makes it more difficult for SFIs to fool the model. The results of the training are listed in Table 16. Each model perfectly classified the prepared test dataset of the SFIs as the outlier class without losing the accuracy on the natural test dataset. This seems to be intuitively correct because SFIs have a clearly different distribution from natural images in input space. However, new SFIs can still be generated to attack these models. Table 17 shows the ratio of the SFIs that successfully attacked each target class. Even for model \( M_4 \), which has the highest detection accuracy, the SFIs targeting the car class succeed in attacking at a high rate. We also attacked the model \( M_4 \) with white-based SFIs to determine the extent to which OOD detection depends on the base color. The generated SFIs were found to fool model \( M_4 \) at a rate of 99% (Table 17). This indicates that models trained with SFIs are still vulnerable to SFIs in different base colors that are not used during training. Note that SFIs can be removed using a simple filtering, but SFIs are only a minimal case of images that are extremely far from natural images. Filtering all such cases is difficult and SFIs could become a potential risk in most DNN-based systems.

5.8 Transferability

Here, we observe the transferability of SFIs on CIFAR10. We input 5-SFIs generated in the source model to the destination model and counting SFIs which yield high confidence scores (\( \geq 90\% \)) in both models. These results are shown in Table 18. Most SFIs are not able to attack other models, but
Table 15: Comparison in four trained models on Fashion-MNIST. The description of the table are the same as Table 14.

| model (accuracy, confidence) | 0     | 1     | 2     | 3     | 4     |
|-----------------------------|-------|-------|-------|-------|-------|
| MLP (89, 97)                | 64/100/0 | 28/100/0 | 47/99/0 | 73/99/0 | 87/98/0 |
| CNNMP (90, 99)              | 100/100/100 | 100/100/100 | 100/100/100 | 100/100/100 | 100/100/100 |
| CNNnone (88, 94)            | 99/85/99 | 78/98/99 | 87/44/4 | 94/52/0 | 93/67/24 |
| CapsNet (91, 20)            | 21/19/19 | 13/19/15 | 15/19/20 | 20/20/16 | 19/18/20 |

Table 16: Classification accuracy (%) of models trained on CIFAR10 and SFIs. The model classifies SFIs as the outlier class. Combined accuracy means the ratio of natural images and SFIs correctly classified. Natural accuracy refers to the ratio of correctly classified natural images. Defense accuracy refers to the ratio of correctly classified SFIs. Reattack accuracy refers to the ratio of successful attacks by the SFIs generated for model $M_i$.

| model | combined | natural | defense | reattack |
|-------|----------|---------|---------|----------|
| $M_0$ | None     | 80      | None    | 99       |
| $M_1$ | 79       | 77      | 100     | 62       |
| $M_2$ | 82       | 79      | 100     | 89       |
| $M_3$ | 83       | 78      | 100     | 35       |
| $M_4$ | 85       | 79      | 100     | 19       |

Table 17: Ratio of SFIs that successfully attacked each target class.

| model | plane | car | bird | cat | deer |
|-------|-------|-----|------|-----|------|
| $M_0$ | 100   | 99  | 100  | 100 | 100  |
| $M_1$ | 63    | 55  | 59   | 90  | 51   |
| $M_2$ | 98    | 90  | 97   | 82  | 47   |
| $M_3$ | 78    | 42  | 38   | 29  | 0    |
| $M_4$ | 2     | 75  | 1    | 19  | 27   |

| model | dog | frog | horse | ship | truck |
|-------|-----|------|-------|------|-------|
| $M_0$ | 100 | 100  | 100   | 100  | 100   |
| $M_1$ | 18  | 32   | 62    | 98   | 93    |
| $M_2$ | 94  | 93   | 98    | 96   | 95    |
| $M_3$ | 13  | 0    | 93    | 54   | 0     |
| $M_4$ | 0   | 0    | 25    | 1    | 39    |

Table 18: Ratio of SFIs that successfully attacked each target class.

| model | white | plane | car | bird | cat | deer |
|-------|-------|-------|-----|------|-----|------|
| $M_0$ | 100   | 99    | 100 | 100  | 100 | 100  |
| $M_1$ | 63    | 55    | 59  | 90   | 51  | 51   |
| $M_2$ | 98    | 90    | 97  | 82   | 47  | 47   |
| $M_3$ | 78    | 42    | 38  | 29   | 0   | 0    |
| $M_4$ | 2     | 75    | 1   | 19   | 27  | 27   |

5.9 Relation to Adversarial Attacks

In this paper, we do not conduct attack efficiency comparison to standard adversarial attacks \[14\][26] and the one pixel attack \[27\]. The concept of our paper is to show images that are extremely far from natural images can also fool CNNs. We are not claiming that SFIs are more efficient than the other attack methods.

One may think that our SFIs are conceptually similar to the one pixel attack \[27\]. However, as we discussed in \[2\], some SFIs produce high confidence scores in the two models. Also, SFIs generated in a robust model tend to succeed in attacking another more vulnerable model. In particular, 45% of SFIs generated in the ResNet50 model fooled the VGG16 model with high confidence scores. Ilyas et al. have hypothesized that the success of adversarial attacks is attributed to non-robust features in a training dataset \[38\]. Non-robust features are imperceptible to humans and improve the generalized performance of DNNs. They have insisted that adversarial attacks add perturbation like non-robust features to natural images and fool DNNs. Therefore, according to this theory, the transferability of adversarial attacks can be explained by non-robust features, which are commonly learned by several models. Altered pixels of SFIs might be such non-robust features, so SFIs have transferability.

6 Conclusion

We addressed and affirmatively answered the fundamental question of whether it is possible to fool DNNs by images that are extremely unrecognizable to humans. We proposed SFIs as a minimal case of such images and revealed the vulnerability of DNNs to SFIs. SFIs are not recognizable to humans as natural objects, but are classified by a DNN as natural images, and fool DNNs. Therefore, according to this theory, the transferability of adversarial attacks can be explained by non-robust features, which are commonly learned by several models. Altered pixels of SFIs might be such non-robust features, so SFIs have transferability.
Table 18: Result of transferability. The values in this table are the number of SFIs and the ratio, which are calculated by inputting 5-SFIs generated in source model to destination model and counting SFIs which yield high confidence scores (≥ 90%) in both models.

| source | VGG16 | NiN   | ResNet50 |
|--------|-------|-------|----------|
|        | 1947 (1.00) | 202 (0.10) | 188 (0.09) |
| NiN    | 338 (0.28) | 1170 (1.00) | 110 (0.09) |
| ResNet50 | 309 (0.45) | 213 (0.31) | 672 (1.00) |

similar to each other in the deep layers of the DNN, although at the input domain, they are clearly different in appearance. This explains why SFIs can fool complex DNNs. We compared four architectures (MLP, CNN with max pooling layers, CNN without them, and CapsNet) regarding their robustness against SFIs and showed that a CNN with max pooling layers, CNN without them, and CapsNet) are more vulnerable to SFIs than other architectures. We also observed the vulnerability of max pooling, which loses spatial information of low-level features. To develop a defense against SFIs, we used OOD detection, which is able to detect the SFIs used for the test dataset with 100% accuracy. However, it was possible to generate further SFIs that can attack the retrained model. Lastly, we showed SFIs have the transferability as well as adversarial examples. This implicates that the altered pixels of SFIs are unrecognizable but commonly learned by different models.

In a future study, since SFIs are minimal cases of images that are extremely different from natural images, we further analyze the vulnerability of DNNs by finding other such images. Furthermore, as some studies have shown that adversarial examples can be used for data augmentation to improve clean accuracy [39, 40], we consider that SFIs can be used as one of the augmentation methods and improve the clean accuracy of the test set.

References

[1] D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané, “Concrete problems in ai safety,” arXiv preprint arXiv:1606.06563, 2016.

[2] M. Hein, M. Andriushchenko, and J. Bitterwolf, “Why relu networks yield high-confidence predictions far away from the training data and how to mitigate the problem,” in IEEE Conference on Computer Vision and Pattern Recognition, pp. 41–50, 2019.

[3] D. Hendrycks and K. Gimpel, “A baseline for detecting misclassified and out-of-distribution examples in neural networks,” International Conference on Learning Representations, 2017.

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

[5] V. Sehwag, A. N. Bhagoji, L. Song, C. Sitawarin, D. Cullina, M. Chiang, and P. Mittal, “Better the devil you know: An analysis of evasion attacks using out-of-distribution adversarial examples,” arXiv preprint arXiv:1905.01726, 2019.

[6] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, “Intriguing properties of neural networks,” in International Conference on Learning Representations, 2014.

[7] S. Das and P. N. Suganthan, “Differential evolution: A survey of the state-of-the-art,” IEEE Transactions on Evolutionary Computation, vol. 15, no. 1, pp. 4–31, 2010.

[8] R. Storn and K. Price, “Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces,” Journal of global optimization, vol. 11, no. 4, pp. 341–359, 1997.

[9] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in International Conference on Learning Representations, 2014.

[10] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778, 2016.

[11] S. Sabour, N. Frosst, and G. E. Hinton, “Dynamic routing between capsules,” in Advances in Neural Information Processing Systems, pp. 3856–3866, 2017.

[12] R. Arora, A. Basu, P. Mianjy, and A. Mukherjee, “Understanding deep neural networks with rectified linear units,” in International Conference on Learning Representations, 2018.

[13] K. Simonyan, A. Vedaldi, and A. Zisserman, “Deep inside convolutional networks: Visualising image classification models and saliency maps,” in International Conference on Learning Representations WS, 2014.

[14] N. Carlini and D. Wagner, “Towards evaluating the robustness of neural networks,” in IEEE Symposium on Security and Privacy, pp. 39–57, 2017.

[15] P.-Y. Chen, Y. Sharma, H. Zhang, J. Yi, and C.-J. Hsieh, “EAD: elastic-net attacks to deep neural networks via adversarial examples,” in the AAAI Conference on Artificial Intelligence, 2018.

[16] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li, “Boosting adversarial attacks with momentum,” in IEEE Conference on Computer Vision and Pattern Recognition, pp. 9185–9193, 2018.
[17] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, “Deepfool: a simple and accurate method to fool deep neural networks,” in IEEE Conference on Computer Vision and Pattern Recognition, pp. 2574–2582, 2016.

[18] K. Eykholt, I.Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song, “Robust physical-world attacks on deep learning visual classification,” in IEEE Conference on Computer Vision and Pattern Recognition, pp. 1625–1634, 2018.

[19] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter, “Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition,” in the 2016 acm sigsac conference on computer and communications security, pp. 1528–1540, 2016.

[20] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter, “A general framework for adversarial examples with objectives,” ACM Transactions on Privacy and Security, vol. 22, no. 3, pp. 1–30, 2019.

[21] S. Thys, W. Van Ranst, and T. Goedeme, “Fooling automated surveillance cameras: Adversarial patches to attack person detection,” in IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2019.

[22] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” in International Conference on Learning Representations, 2015.

[23] F. Croce and M. Hein, “Sparse and imperceivable adversarial attacks,” in IEEE International Conference on Computer Vision, pp. 4724–4732, 2019.

[24] A. Modas, S.-M. Moosavi-Dezfooli, and P. Frossard, “Sparsefool: a few pixels make a big difference,” in IEEE Conference on Computer Vision and Pattern Recognition, pp. 9087–9096, 2019.

[25] N. Narodytska and S. P. Kasiviswanathan, “Simple black-box adversarial perturbations for deep networks,” arXiv preprint arXiv:1612.06299, 2016.

[26] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, “The limitations of deep learning in adversarial settings,” in IEEE Symposium on Security and Privacy, pp. 372–387, 2016.

[27] J. Su, D. V. Vargas, and K. Sakurai, “One pixel attack for fooling deep neural networks,” IEEE Transactions on Evolutionary Computation, vol. 23, no. 5, pp. 828–841, 2019.

[28] A. Mahendran and A. Vedaldi, “Understanding deep image representations by inverting them,” in IEEE Conference on Computer Vision and Pattern Recognition, pp. 5188–5196, 2015.

[29] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in IEEE Conference on Computer Vision and Pattern Recognition, 2015.

[30] A. Krizhevsky, “Learning multiple layers of features from tiny images,” tech. rep., University of Toronto, 2009.

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

[32] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al., “Pytorch: An imperative style, high-performance deep learning library,” in Advances in Neural Information Processing Systems, pp. 8026–8037, 2019.

[33] M. Lin, Q. Chen, and S. Yan, “Network in network,” in International Conference on Learning Representations, 2014.

[34] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer, “Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems,” IEEE Transactions on Evolutionary Computation, vol. 10, no. 6, pp. 646–657, 2006.

[35] D. V. Vargas, J. Murata, H. Takano, and A. C. B. Delbem, “General subpopulation framework and taming the conflict inside populations,” IEEE Transactions on Evolutionary Computation, vol. 23, no. 1, pp. 1–36, 2015.

[36] D. Hendrycks and K. Gimpel, “A baseline for detecting misclassified and out-of-distribution examples in neural networks,” in International Conference on Learning Representations, 2017.

[37] K. Lee, K. Lee, H. Lee, and J. Shin, “A simple unified framework for detecting out-of-distribution samples and adversarial attacks,” in Advances in Neural Information Processing Systems, pp. 7167–7177, 2018.

[38] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry, “Adversarial examples are not bugs, they are features,” in Advances in Neural Information Processing Systems, pp. 125–136, 2019.

[39] J. Kim, J. Jeong, and J. Shin, “M2m: Imbalanced classification via major-to-minor translation,” in IEEE Conference on Computer Vision and Pattern Recognition, pp. 13896–13905, 2020.

[40] C. Xie, M. Tan, B. Gong, J. Wang, A. L. Yuille, and Q. V. Le, “Adversarial examples improve image recognition,” in IEEE Conference on Computer Vision and Pattern Recognition, pp. 819–828, 2020.