

Received June 26, 2018, accepted August 8, 2018, date of publication August 20, 2018, date of current version September 7, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2866197

# Multi-Targeted Adversarial Example in Evasion Attack on Deep Neural Network

HYUN KWON<sup>1</sup>, YONGCHUL KIM<sup>2</sup>, KI-WOONG PARK<sup>3</sup>, (Member, IEEE),  
HYUNSOO YOON<sup>1</sup>, AND DAESEON CHOI<sup>4</sup>, (Member, IEEE)

<sup>1</sup>School of Computing, Korea Advanced Institute of Science and Technology, Daejeon 34141, South Korea

<sup>2</sup>Department of Electrical Engineering, Korea Military Academy, Seoul 01819, South Korea

<sup>3</sup>Department of Computer and Information Security, Sejong University, Seoul 05006, South Korea

<sup>4</sup>Department of Medical Information, Kongju National University, Gongju 32588, South Korea

Corresponding author: Daeseon Choi (sunchoi@kongju.ac.kr)

This work was supported in part by the National Research Foundation of Korea funded by the Korea Government (MSIT) under Grants 2016R1A4A1011761 and 2017R1A2B4006026, and in part by the Institute for Information and Communications Technology Promotion funded by the Korea Government (MSIT) under Grant 2016-0-00173 (Security Technologies for Financial Fraud Prevention on Fintech).

**ABSTRACT** Deep neural networks (DNNs) are widely used for image recognition, speech recognition, pattern analysis, and intrusion detection. Recently, the adversarial example attack, in which the input data are only slightly modified, although not an issue for human interpretation, is a serious threat to a DNN as an attack as it causes the machine to misinterpret the data. The adversarial example attack has been receiving considerable attention owing to its potential threat to machine learning. It is divided into two categories: targeted adversarial example and untargeted adversarial example. The untargeted adversarial example happens when machines misclassify an object into an incorrect class. In contrast, the targeted adversarial example attack causes machines to misinterpret the image as the attacker's desired class. Thus, the latter is a more elaborate and powerful attack than the former. The existing targeted adversarial example is a single targeted attack that allows only one class to be recognized. However, in some cases, a multi-targeted adversarial example can be useful for an attacker to make multiple models recognize a single original image as different classes. For example, an attacker can use a single road sign generated by a multi-targeted adversarial example scheme to make model A recognize it as a stop sign and model B recognize it as a left turn, whereas a human might recognize it as a right turn. Therefore, in this paper, we propose a multi-targeted adversarial example that attacks multiple models within each target class with a single modified image. To produce such examples, we carried out a transformation to maximize the probability of different target classes by multiple models. We used the MNIST datasets and TensorFlow library for our experiment. The experimental results showed that the proposed scheme for generating a multi-targeted adversarial example achieved a 100% attack success rate.

**INDEX TERMS** Deep neural network (DNN), evasion attack, adversarial example, machine learning.

## I. INTRODUCTION

As the primary basis of emerging computing technology, machine learning technologies have played a key role as a classification scheme. In particular, deep neural networks (DNNs) [35] are used to achieve better performance in areas such as image recognition [37], speech recognition [14], pattern analysis [4], and intrusion detection [33]. Recently, Szegedy *et al.* [40] introduced an adversarial example method that makes a DNN misidentify an image with minimal distortion, which even humans could not tell the difference. Hence, the adversarial example poses a serious threat to the DNN. For example, if a traffic sign is generated with the adversarial

example method to misidentify a left turn as a right turn, a human could recognize it as a left turn, but an autonomous vehicle using a DNN would recognize it as a right turn. For these reasons, adversarial examples, which have been actively researched recently, are classified into targeted adversarial examples and untargeted adversarial examples according to the attack target. A targeted adversarial example is the generation of an adversarial example that causes the DNN to misidentify the original class as the attacker's intended class. In contrast, an untargeted adversarial example is an attack that causes a DNN to misidentify the original class as an arbitrary class. The untargeted adversarial example has the

advantage in that the amount of distortion is small and the learning process for generation is fast, whereas the targeted adversarial example is a more sophisticated attack. In most cases of targeted adversarial examples that have been studied in the literature, research has been performed on a single targeted attack that allows only one class to be recognized. In situations such as in the military, an attacker may need to attack multiple models to make each model recognize the same class as different classes. For example, when an attacker installed anti-tank mines on the left path after a turn and placed RPG-7 weapons on the right path, it would be very useful if the attacker could lead an enemy tank unit in the left direction and troop transport vehicles in the right direction by using a multi-targeted adversarial example. In other words, it is necessary to create a multi-targeted adversarial example so that models A and B misclassify the multi-targeted adversarial example as each class intended by the attacker. Therefore, in this paper, we propose a multi-targeted adversarial example method that can attack multiple models simultaneously with each target class by using one modulated image. The contributions of this study are as follows:

- To the best of our knowledge, this is the first study on generating a multi-targeted adversarial example that is misclassified as different target classes by multiple models. We describe the principles of the proposed method and systematically organize the proposed scheme to generate the multi-targeted adversarial example.
- The proposed method was considered in terms of scalability for multiple models. We analyzed the amount of distortion and learning process load as the number of attack models increased. The results of this analysis showed a trade-off between distortion and model scalability.
- The proposed method was evaluated by multiple analyses, performance evaluations, and practical experiments based on the datasets. To demonstrate the effectiveness of the proposed method, we experimented with the MNIST dataset. We also analyzed the variation of distortion for the combination of multiple target classes. This analysis can be useful information for attackers planning multi-targeted attacks in the future.

The remainder of this paper is organized as follows. Section 2 provides some background information and related research works on the adversarial example. The problem definition of the proposed method is presented in Section 3, and the proposed method is introduced in Section 4. Section 5 presents the experiment and evaluation method of the proposed scheme, and the experiment results are presented in Section 6. A discussion of the proposed system is presented in Section 7. Section 8 concludes the paper.

## II. BACKGROUND AND RELATED WORKS

Barreno *et al.* [2] discussed several security issues in machine learning. They classified attacks on machine learning as causative attacks [3], [26] that affect learning, with control on the training data, and as exploratory attacks [40], [43]

that cause misclassification but do not affect the training process. As machine learning technology evolves, an adversarial example [40] of a well-known exploratory attack has attracted attention to the security of DNNs. A study on adversarial examples was first introduced by Szegedy *et al.* [40] in 2014. The main goal of using an adversarial example is to cause the DNN to make a mistake by adding a small amount of noise to an original image; however, humans cannot distinguish the difference between the original and the distorted image.

The basic method of generating adversarial examples is described in Section 2.1. Section 2.2 briefly introduces the defense of the adversarial example. In Section 2.3, adversarial example attacks are divided into four categories and described.

### A. ADVERSARIAL EXAMPLE GENERATION

The basic architecture that generates an adversarial example consists of two elements: the target model and the transformer. The transformer takes the original samples  $x$  and the target class  $y$  as input data. The transformer then creates an output, a transformed example  $x^* = x + w$  with noise value  $w$  added to the original sample  $x$ . The transformed example  $x^*$  is supplied as input data to the target model. The target model then provides the transformer with the class probability results of the transformed example. The transformer updates the noise values  $w$  in the transformed example  $x^* = x + w$  so that the other class probabilities are higher than the original class probabilities while minimizing the distortion distances between  $x^*$  and  $x$ .

### B. DEFENSE OF THE ADVERSARIAL EXAMPLE

Defense of the adversarial examples has been studied with adversarial training [13], [40], filtering method [10], [36], defensive distillation [32], and the magnet method [23]. First, Szegedy *et al.* [40] and Goodfellow *et al.* [13] introduced adversarial training methods that resist adversarial example attacks. The DNN model has resistance to the adversarial example attack by learning through adversarial examples. To improve the defense against adversarial examples, a recent study by Tramèr *et al.* [41] proposed an ensemble adversarial training method by using a number of local models. This method is simple and increases the defense against adversarial examples. However, the adversarial training method has a high probability of reducing the classification accuracy of the original sample. Second, Shen *et al.* [36] proposed a filtering method that eliminates adversarial perturbation using generative adversarial nets [12]. This method preserves the classification accuracy for the original sample, but requires a filtering module and module process learning to eliminate adversarial perturbation. Instead of using a generative adversarial net, Fawzi *et al.* [10] proposed a robust classifier that resists adversarial examples by removing adversarial perturbation using theoretical methods. Fawzi *et al.* [11] also proposed an advanced method that is resistant to adversarial examples through classifier decision analysis using geometric transformation. Third, Papernot *et al.* [32] proposed a

defensive distillation that has resistance to the adversarial example attack. To block the attack gradient, this defensive distillation has two neural networks in which the output class probability of the classifier is used as the input for the second stage of classifier training. Similarly, Mosca and Magoulas [25] proposed a smooth gradient method to block the attack gradients. As a state-of-the-art defense against the black box attack, the magnet method was proposed by Meng and Chen [23]. It consists of a detector and a reformer to resist the adversarial example attack. There are two steps in this method: First, a detector filters the adversarial example with more distortion as a prelearning step consisting of the difference between the original sample and the adversarial example. Then, a reformer finds the original sample that can be transformed into a small perturbation of the adversarial example. However, according to a recent study by Carlini and Wagner [7], [8], it was shown that the filtering method, defensive distillation, and the magnet method could be deceived 100% of the time [36].

### C. DETAILS OF THE ADVERSARIAL EXAMPLE ATTACK

Related works on the adversarial example attack can be divided into four categories: target model information, distance measure, recognition on adversarial example, and generating method, as described in the following subsections.

#### 1) TYPE OF TARGET MODEL INFORMATION

Attacks that generate adversarial examples can also be divided into two different types, depending on how much information about the target is required for the attack: the white box attack [8], [24], [40] and the black box attack. The white box attack is used when the attacker has detailed information about the target model, i.e., model architecture, parameters, and class probabilities of the output. Hence, the success rate of the white box attack reaches almost 100%. Some articles that have been recently published [23], [32] showed that defending a DNN from a white box attack [6], [7] is extremely difficult.

On the other hand, a black box attack is used when the attacker can query the target model without the target model information. The well-known black box attacks are of two types: the substitute network attack [30] and the transferability attack [13], [40]. The substitute model attack proposed in [30] is an example of a well-known black box attack. In this scheme, an attacker can create a substitute network similar to the target model by repeating the query process. Once a substitute network is created, the attacker can perform a white box attack. Papernot *et al.* [30] created a substitute network against Amazon and Google services and showed that their success rate on the MNIST dataset was 81.2%.

Second, the transferability attack is another example of a well-known black box attack. The works in [13] and [40] introduced the transferability that an adversarial example had modified for a single local model, and it is effective for the other models that classify the same type of data. Since then, various studies [19], [27], [42] on transferability have

been done. To improve the transferability, Strauss *et al.* [39] proposed an ensemble adversarial example method by using multiple local models to attack the other models.

#### 2) TYPE OF DISTANCE MEASURE

There are three ways to measure the distortion between the original sample and the adversarial example [8], [23]. The first distance measure  $L_0$  represents the sum of the number of all changed pixels:

$$\sum_{i=0}^n |x_i - x_i^*| \quad (1)$$

where  $x_i$  is an original  $i^{th}$  pixel and  $x_i^*$  is an adversarial example  $i^{th}$  pixel. The second distance measure  $L_2$  represents a standard Euclidean norm as follows:

$$\sum_{i=0}^n \sqrt{(x_i - x_i^*)^2} \quad (2)$$

The third distance measure  $L_\infty$  is the maximum distance value between  $x_i$  and  $x_i^*$ . Therefore, as the three distance measures become small, the similarity of the sample image to the original sample increases from a human's perspective.

#### 3) TYPE OF TARGET RECOGNITION ON THE ADVERSARIAL EXAMPLE

Depending on which class the target model recognizes from the adversarial examples, the category of adversarial examples [8], [28], [42] can be divided into two subcategories: a targeted adversarial example and an untargeted adversarial example. First, a targeted adversarial example is an adversarial example that causes the target model to recognize the adversarial image as a particular intended class and can be expressed mathematically as follows.

Given a target model and original sample  $x \in X$ , the problem can be reduced to an optimization problem that generates a targeted adversarial example  $x^*$ :

$$x^* : \underset{x^*}{\operatorname{argmin}}(x, x^*) \quad \text{s. t. } f(x^*) = y^* \quad (3)$$

where  $L(\cdot)$  is a distance measure between original sample  $x$  and transformed example  $x^*$ , with  $y^*$  being a particular intended class.  $\underset{x}{\operatorname{argmin}} F(x)$  is the  $x$  value at which the function  $F(x)$  becomes minimal. s. t. is an abbreviation of such that.  $f(\cdot)$  is an operation function that provides the class results for the input values of the target model.

On the other hand, an untargeted adversarial example is an adversarial example that causes the target model to recognize the adversarial image as a class other than the original class and can be expressed mathematically as follows.

Given a target model and original sample  $x \in X$ , the problem can be reduced to an optimization problem that generates an untargeted adversarial example  $x^*$ :

$$x^* : \underset{x^*}{\operatorname{argmin}}(x, x^*) \quad \text{s. t. } f(x^*) \neq y \quad (4)$$

where  $y \in Y$  is the original class.

The untargeted adversarial example has the advantage of less distortion in the original images and a shorter learning time compared to the targeted adversarial example. However, the targeted adversarial example is a more elaborate and powerful attack to control the perception of the attacker's chosen class.

Until recently, conventional research [8], [28] has only studied the single-model misclassification of targeted adversarial examples. However, there has been no published scheme for building targeted adversarial examples that multiple models misclassify as each provided class. In this paper, we propose a multi-targeted adversarial example that can trick multiple models into misclassifying each target class.

#### 4) METHODS OF THE ADVERSARIAL EXAMPLE ATTACK

There are four typical attacks that generate adversarial examples. The first method is the fast-gradient sign method (FGSM) [13], which can find  $x^*$  through  $L_\infty$ :

$$x^* = x + \epsilon \cdot \text{sign}(\nabla \text{loss}_{F,t}(x)) \quad (5)$$

where  $F$  is an object function, and  $t$  is a target class. In every iteration of the FGSM, the gradient is updated by  $\epsilon$  from the original  $x$ , and the  $x^*$  is found through optimization. This method is simple and has good performance. The second method is iterative FGSM (I-FGSM) [18], which is an updated version of the FGSM. Instead of changing the amount of  $\epsilon$  in every step, the method updates a smaller amount of  $\alpha$  and eventually clips it by the same  $\epsilon$ :

$$x_i^* = x_{i-1}^* - \text{clip}_\epsilon(\alpha \cdot \text{sign}(\nabla \text{loss}_{F,t}(x_{i-1}^*))) \quad (6)$$

This I-FGSM method provides a better performance than the FGSM method. The third is the DeepFool method [24], which is an untargeted attack that uses an  $L_2$  distance measure. This method generates an adversarial example that is more efficient than that generated by FGSM and as close as possible to the original image. To generate an adversarial example with the DeepFool method, it constructs a neural network and looks for  $x^*$  using the linear approximation method. However, since the neural network is not completely linear, the method has to find the adversarial example through many iterations, and as such, it is a more complicated process than FGSM. The fourth method is the Carlini attack [8], which is the latest attack method providing the best performance compared to those of FGSM and I-FGSM. This method can achieve a 100% success rate even against distillation structures [32] that have recently been introduced in the literature. The key point of this method is to use a different objective function as follows:

$$D(x, x^*) + c \cdot f(x^*), \quad (7)$$

instead of using the conventional objective function  $D(x, x^*)$ , and it proposes a way to find an appropriate binary  $c$  value. In addition, it suggests a method to control the attack success rate even if the distortion increases by reflecting the confidence value as follows:



$$f(x^*) = \max(Z(x^*)_t) - \max\{Z(x^*)_i : i \neq t\}, -k), \quad (8)$$

where  $Z(\cdot)$  represents the pre-softmax classification result vector, with  $t$  being a target class. In this study, we constructed the model by applying the Carlini attack, which is the most powerful method among the four methods, and used  $L_2$  as the distance measure.

### III. PROBLEM DEFINITION

Table 1 shows the class scores of an adversarial example: "2"  $\rightarrow$  "3" and original sample "2". If the DNN model takes a sample as an input value, it provides the output value of the classification result score of each class. Thus, the DNN model classifies the sample as a class with the highest class score.

**TABLE 1. Class score of an adversarial example: "2"  $\rightarrow$  "3".**

Description	Original ("2")	Adversarial example("3")
Image		
Class score	[4 8 <b>29</b> -5 -8 -16 3 -1 -3 -9]	[-1 2 <b>18.701</b> <b>18.703</b> -11 -4 -16 3 2 -3]

For example, the class "2" (29) of the original sample is higher than any other class score, as shown in Table 1. Therefore, the DNN model correctly classifies the original sample as class "2". However, the DNN model incorrectly classifies the adversarial example as class "3". Class "3" (18.703) of the adversarial example is slightly higher than the score of the original class (18.701), as shown in Table 1. This result shows that the process of generating the adversarial example continues until the target class score is slightly higher than the original class score owing to the minimization of the distance between the adversarial example and the original sample. This is because the distance between these two must be minimized. If this is explained by the expression of the decision boundary, it is as shown in Fig. 1(a):

Fig. 1(a) shows the decision boundary of the single model A, which misclassifies adversarial example  $x^*$  targeting model A. The line is the decision boundary of the model function  $f_A(\cdot)$ . For example, in Fig. 1(a), model A classifies the original sample  $x$  as original class "2" because the original sample  $x$  is in the area of class "2":

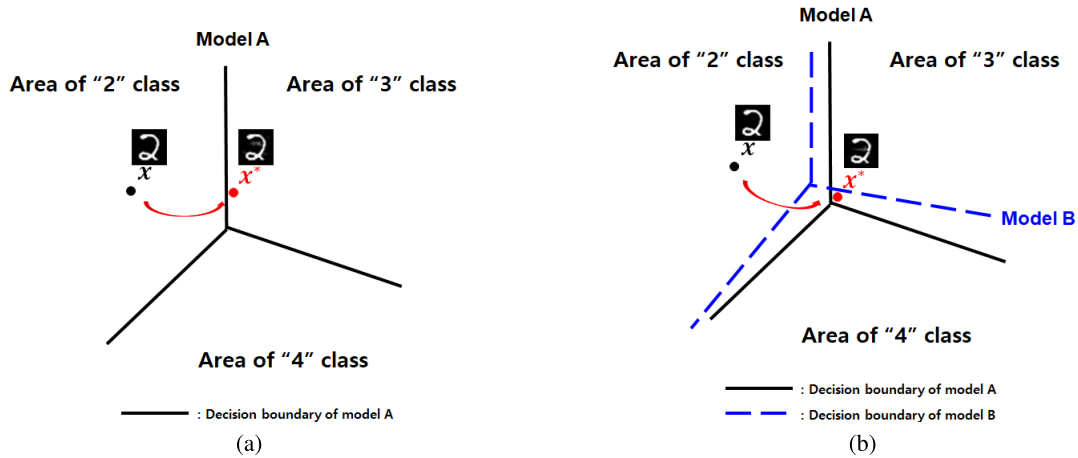
$$f_A(x) = y, \quad (9)$$

where  $y$  is the original class "2". However, since the adversarial example  $x^*$  is in the area of class "3", model A misclassifies the targeted adversarial example  $x^*$  as target class "3"

$$f_A(x^*) = t_A, \quad (10)$$

where  $t_A$  is the target class "3" chosen by the attacker. Fig. 1(a) shows that the targeted adversarial example  $x^*$  is generated around the line of model A and is misclassified as target class "3" while minimizing the distance of the original sample.





**FIGURE 1.** Examples of targeted adversarial examples: a single targeted adversarial example of model A and multi-targeted adversarial example of models A and B. (a) Single targeted. (b) Multi-targeted.

In some situations, such as in the military, adversarial examples may need to be misclassified as different target classes for different models. For example, an adversarial example  $x^*$  in Fig. 1(b) should be incorrectly classified as target class "3" by model A and as target class "4" by model B.

$$f_A(x^*) = t_A, \quad f_B(x^*) = t_B, \quad (11)$$

where  $t_A$  is target class "3" and  $t_B$  is target class "4". Thus, to be misclassified into each target class intended by models A and B, an adversarial example must be generated in the combined target areas of A (" $t_A$ ") and B (" $t_B$ ").

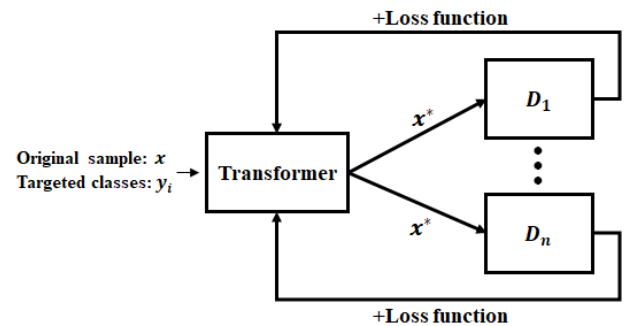
Considering this concept, this paper proposes a new scheme for generating a multi-target adversarial example. This method generates multiple targeted adversarial examples that are misclassified as each target class intended by multiple models while minimizing distortion from the original sample.

#### IV. PROPOSED METHOD

To generate a multi-targeted adversarial example, we propose an architecture that consists of a transformer and multiple models  $D_i$  ( $1 \leq i \leq n$ ) (Fig. 2). The role of the transformer is to generate a multi-targeted adversarial example. The multiple models  $D_i$  ( $1 \leq i \leq n$ ) are pretrained classifiers that classify input values. This proposed architecture is a modified architecture of [20].

The transformer takes the original sample  $x \in X$  and the target class  $y_i \in Y$  ( $1 \leq i \leq n$ ) as input values and generates the transformed example  $x^*$  as the output value.  $y_i$  is the target class chosen by the attacker for each model  $D_i$  ( $1 \leq i \leq n$ ). The multiple models  $D_i$  ( $1 \leq i \leq n$ ) take the transformed example  $x^*$  as an input value and provide the transformer with a feedback, which is a classification result (i.e., loss).

The goal of the proposed scheme is to generate a multi-targeted adversarial example  $x^*$  that is misclassified by each of the multiple models  $D_i$  ( $1 \leq i \leq n$ ) as each target class  $y_i$



**FIGURE 2.** The proposed architecture.

( $1 \leq i \leq n$ ), while minimizing the calculated pixel distance of the original sample  $x$ . In the mathematical expression, the operation functions of  $D_i$  ( $1 \leq i \leq n$ ) are denoted by  $f_i(x)$  ( $1 \leq i \leq n$ ), respectively. Given the pretrained models  $D_i$  ( $1 \leq i \leq n$ ) and the original sample  $x \in X$  and target classes  $y_i$  ( $1 \leq i \leq n$ ), this is an optimization problem that generates a multi-targeted adversarial example  $x^*$ :

$$x^* : \operatorname{argmin}_{x^*} L(x, x^*) \text{ s.t. } f_i(x^*) = y_i \quad (1 \leq i \leq n), \quad (12)$$

where  $L(\cdot)$  is the distance between the original sample  $x$  and the multi-targeted adversarial example  $x^*$ .

To achieve this goal, the procedure has two steps: create pre-training models  $D_i$  ( $1 \leq i \leq n$ ) and generate a multi-targeted adversarial example  $x^*$ . First, the models  $D_i$  ( $1 \leq i \leq n$ ) go through a learning process to correctly classify the original sample  $x$  into its original class  $y^{\text{org}}$ :

$$f_i(x) = y^{\text{org}} \in Y, \quad (13)$$

where  $y^{\text{org}}$  is the original class. In our experiment, models  $D_i$  ( $1 \leq i \leq n$ ) were trained to classify the original sample with higher than 99% accuracy.

Second, the transformer accepts the original sample  $x$  and target classes  $y_i$  ( $1 \leq i \leq n$ ) as input values and generates

transformed example  $x^*$ . In this study, we modified the transformer architecture in [8] and  $x^*$  was defined as

$$x^* = \frac{\tanh(x + w)}{2},$$

where  $w$  is used as a modifier to optimally minimize the gradient and  $\tanh$  is used to soften the gradient. The classification losses of  $x^*$  by models  $D_i$  ( $1 \leq i \leq n$ ) are returned to the transformer. The transformer then updates the transformed example  $x^*$  through the iterative process above to calculate the total loss  $loss_T$  and optimally minimize the total loss  $loss_T$ .  $loss_T$  is defined as

$$loss_T = loss_{\text{distortion}} + \sum_{i=1}^n c_i \times loss_i, \quad (14)$$

where  $loss_{\text{distortion}}$  is the distortion of the transformed example,  $loss_i$  is the classification loss of  $D_i$ , and  $c_i$  is the loss weight of model  $D_i$ . The initial value of the loss weight is 1.  $loss_{\text{distortion}}$  is the distance between the original sample  $x$  and the transformed example  $x^*$ .

$$loss_{\text{distortion}} = \sqrt{(x^* - \frac{\tanh(x)}{2})^2}.$$

To satisfy  $f_i(x^*) = y_i$  ( $1 \leq i \leq n$ ), we need to minimize  $\sum_{i=1}^n loss_i$ :

$$\sum_{i=1}^n loss_i = \sum_{i=1}^n g_i(x^*),$$

where  $g_i(k) = \max \{Z_i(k)_j : j \neq y_i\} - Z_i(k)_{y_i}$  and  $Z_i$  [8], [31] is the probability of the classes being predicted by the model  $D_i$ .  $f_i(x^*)$  has a higher probability of predicting the target class  $y_i$  than the other classes by optimally minimizing  $loss_i$ .

The detailed procedure for generating a multi-targeted adversarial example is described in Algorithm 1. In Algorithm 1, the iteration  $r$  depends on the number of target models, and, therefore, the experimental results in Section 5.2 will show the correlation analysis between the iteration  $r$  and the number of target models.

## V. EXPERIMENT AND EVALUATION

Through experiments, we show a proposed scheme that generates a multi-targeted adversarial example that is misclassified by each of the multiple models as each target class, while minimizing the calculated pixel distance of the original sample. We used TensorFlow [1], an open source library widely used for machine learning. A server with a Xeon E5-2609 1.7-GHz processor was used in the experiment.

### A. EXPERIMENTAL METHOD

The MNIST dataset [22] was used as the dataset in the experiment. It is a standard dataset with handwritten images from 0 to 9. The experimental method consisted of pretraining  $D_i$  ( $1 \leq i \leq n$ ) and generating the multi-targeted adversarial example.

**Algorithm 1** Generation of a multi-targeted adversarial example in a transformer.

**Input:** original sample  $x$ , targeted class  $y_i$  ( $1 \leq i \leq n$ ), iterations  $r$ , loss weight  $c_i$

**Multi-Targeted Adversarial Example Generation:**

```

 $w \leftarrow 0$ 
 $t \leftarrow y_i$ 
 $x^* \leftarrow x$ 
for  $r$  step do
   $x^* \leftarrow \frac{\tanh(x^* + w)}{2}$ 
   $loss1 \leftarrow \sqrt{(x^* - \frac{\tanh(x)}{2})^2}$ 
   $loss2 \leftarrow c_i \sum_{j=1}^n \max \{Z_i(x^*)_j : j \neq t\} - Z_i(x^*)_t$ 
   $loss3 \leftarrow loss1 + loss2$ 
  Update  $w$  by minimizing the gradient of  $loss3$ 
end for
return  $x^*$ 

```

First, multiple models  $D_i$  ( $1 \leq i \leq n$ ) are common convolution neural networks [21]. Their configuration and training parameters are shown in Tables 8 and 9 of the Appendix. To have sufficient means to train multiple models  $D_i$  ( $1 \leq i \leq n$ ), we used 60,000 samples of training data. Five models were used in this experiment. As shown in Table 10 in the Appendix, several  $D_i$  ( $1 \leq i \leq 5$ ) models were created by learning different training data while maintaining the accuracy of the original sample at over 99%. In the experimental results using 10,000 test data,  $D_i$  ( $1 \leq i \leq 5$ ) correctly classified the original sample into the original class at over 99% accuracy.




























































































Second, the Adam algorithm [16] was used as an optimizer to generate a multi-targeted adversarial example, minimizing the total loss with a learning rate of  $1 \times 10^{-2}$  and an initial constant of  $1 \times 10^{-3}$ . For a given number of iterations, the transformer updated the transformed example and provided it to models  $D_i$  ( $1 \leq i \leq n$ ), and received feedback from these models. At the end of the set of iterations, the transformed example  $x^*$  was evaluated on the basis of the targeted attack success, the required number of iterations, human recognition, and the amount of distortion. Distortion was measured as the pixel distance from the original sample, using mean square error.

### B. EXPERIMENTAL RESULTS










The experimental results showed images of multi-targeted adversarial examples, the distortion of each target class, the class score of a multi-targeted adversarial example, the scalability analysis, and multiple target class information.

Table 2 shows an example of multi-targeted adversarial examples that were misclassified as different target classes by models  $D_1$  and  $D_2$ . The table 2 consists of target classes that were misclassified by  $D_2$  for each fixed target class that was misclassified by  $D_1$ . This example was required for 1000 iterations, and the average distortion in the example was 3.04. In Table 2, the multi-targeted adversarial

**TABLE 2.** Example of multi-targeted adversarial examples that were misclassified as different target classes by models  $D_1$  and  $D_2$ .

$D_1$	Targeted classes misclassified by $D_2$									
	"0"	"1"	"2"	"3"	"4"	"5"	"6"	"7"	"8"	"9"
"9"										
"8"										
"7"										
"6"										
"5"										
"4"										
"3"										
"2"										
"1"										
"0"										

**TABLE 3.** The distortion of multi-targeted adversarial samples for the original sample "9" for each target class of model  $D_2$  when target class of model  $D_1$  is "0" in Table 2.

$D_1$	Targeted classes misclassified by $D_2$									
	"0"	"1"	"2"	"3"	"4"	"5"	"6"	"7"	"8"	
"0"										
Rate	2.78	2.74	2.68	2.54	2.74	3.76	3.15	3.41	2.35	

examples were similar to the original samples in human perception.

For original sample "9" and target class "0" of model  $D_1$  in Table 2, the distortion of the multi-targeted adversarial examples for each target class in model  $D_2$  is analyzed in Table 3. This table 3 shows that the distortion of these samples was different for each target class in model  $D_2$ . For example, in model  $D_2$ , target class "5" resulted in maximum distortion and target class "8" resulted in minimal distortion. The total average distortion in Table 3 is approximately 2.91. In addition, Fig. 4 of the Appendix shows the distortion of the multi-targeted adversarial examples for each target class in models  $D_1$  and  $D_2$  for Table 2.

Table 4 shows an example of multi-targeted adversarial examples for each target class in models  $D_1$  and  $D_2$  for original sample "9" in Table 2. Similar to Table 2, Table 4 shows that in terms of human perception, multi-targeted adversarial examples were similar to the original sample "9". Fig. 5 of the Appendix shows the average distortion of each target class of models  $D_1$  and  $D_2$  for original sample "9". This information can be used to select target classes in certain scenarios.

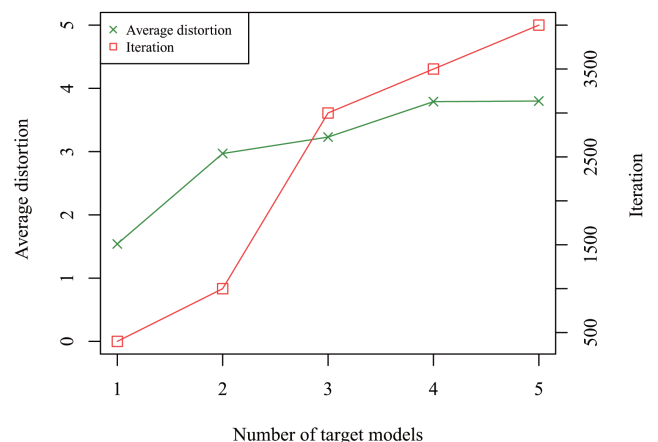



















































































**FIGURE 3.** Average distortion and iteration at 100% targeted attack success rate of 900 random multi-targeted adversarial examples for each set of models.

Table 5 shows the class score of a multi-targeted adversarial example for target class "0" ( $D_1$ ) and target class "8" ( $D_2$ ) for original sample "9" in Table 2. For  $D_1$ , the score of the target class "0" (5.09) is slightly higher than that of

**TABLE 4.** Example of multi-targeted adversarial examples for each target class in models  $D_1$  and  $D_2$  for the original sample "9" of Table 2.

$D_1$	Targeted classes misclassified by $D_2$								
	"0"	"1"	"2"	"3"	"4"	"5"	"6"	"7"	"8"
"0"									
"1"									
"2"									
"3"									
"4"									
"5"									
"6"									
"7"									
"8"									

**TABLE 5.** Class score of original sample "9" and a multi-targeted adversarial example: model  $D_1$  ("9" → "0") and model  $D_2$  ("9" → "8") in Table 2.







Description	Original ("9")	Multi-targeted adversarial example	
		$D_1$ ("0")	$D_2$ ("8")
Image			
Class score	[-11 -11 -15 0.03 11 -4.1 -18 6.6 6.8 <b>32</b> ]	[ <b>5.09</b> -2.5 -2.7 -3.9 -1.6 -2.5 -2.6 4.3 5.08 <b>5.03</b> ]	[-2.1 -2.6 -2.6 3.4 1.5 -4.7 -8.2 0.9 <b>5.08</b> <b>5.03</b> ]

the original class (5.03). For  $D_2$ , the score of the target class "8" (5.08) is slightly higher than that of the original class (5.03). To optimally reduce the distortion of a multi-targeted adversarial example, this result shows that the multi-targeted adversarial example had been modified until the score of each target class of models A and B was higher than that of the original class.

To analyze the scalability of the multi-targeted adversarial examples, we show in Fig. 3 the average distortion and iteration of a 100% targeted attack success rate of 900 random multi-targeted adversarial examples for each set of models. In the figure, the iteration pattern shows that multi-targeted adversarial examples required more learning processes to attack multiple models as the number of models increased. On the other hand, the pattern of average distortion shows that as the number of models increased, the distortion increased, but the rate of change decreased.

In addition, Table 6 shows an example of multi-targeted adversarial examples for each number of models in Fig. 3. In Table 6, since there were at least three models, there was little difference in the multi-targeted adversarial example with regard to human perception.

**TABLE 6.** Example of a multi-targeted adversarial example for each number of models shown in Fig. 3.

Description	Number of target models				
Original	"1"	"2"	"3"	"4"	"5"
					

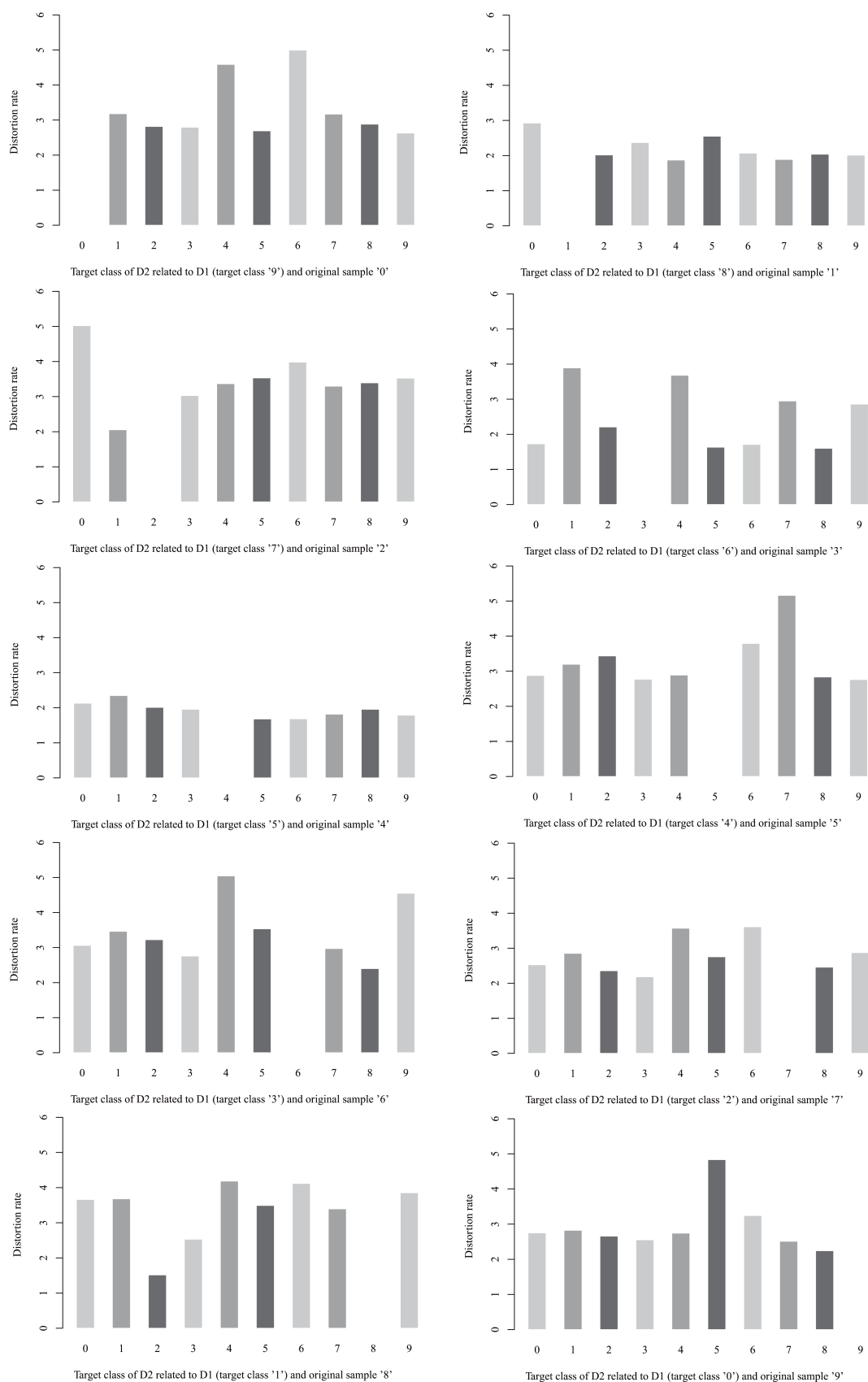
For the multi-targeted adversarial example aimed at the five models in Table 6, Table 7 shows the class score of the multi-targeted adversarial example for each target class in models  $D_1$ ,  $D_2$ ,  $D_3$ ,  $D_4$ , and  $D_5$ . It is possible to generate a multi-targeted adversarial example that is misclassified by each model as each of the several target classes selected by the attacker, as shown in Table 7.

## VI. DISCUSSION

### a: Attack Considerations

From Figs. 4 and 5 of the Appendix, we can see that the distortion of the multi-targeted adversarial example for each target class was different. This information can be useful if





**FIGURE 4.** Average distortion of the target class of model  $D_2$  for each target class of model  $D_1$  and an original sample (1000 iterations).

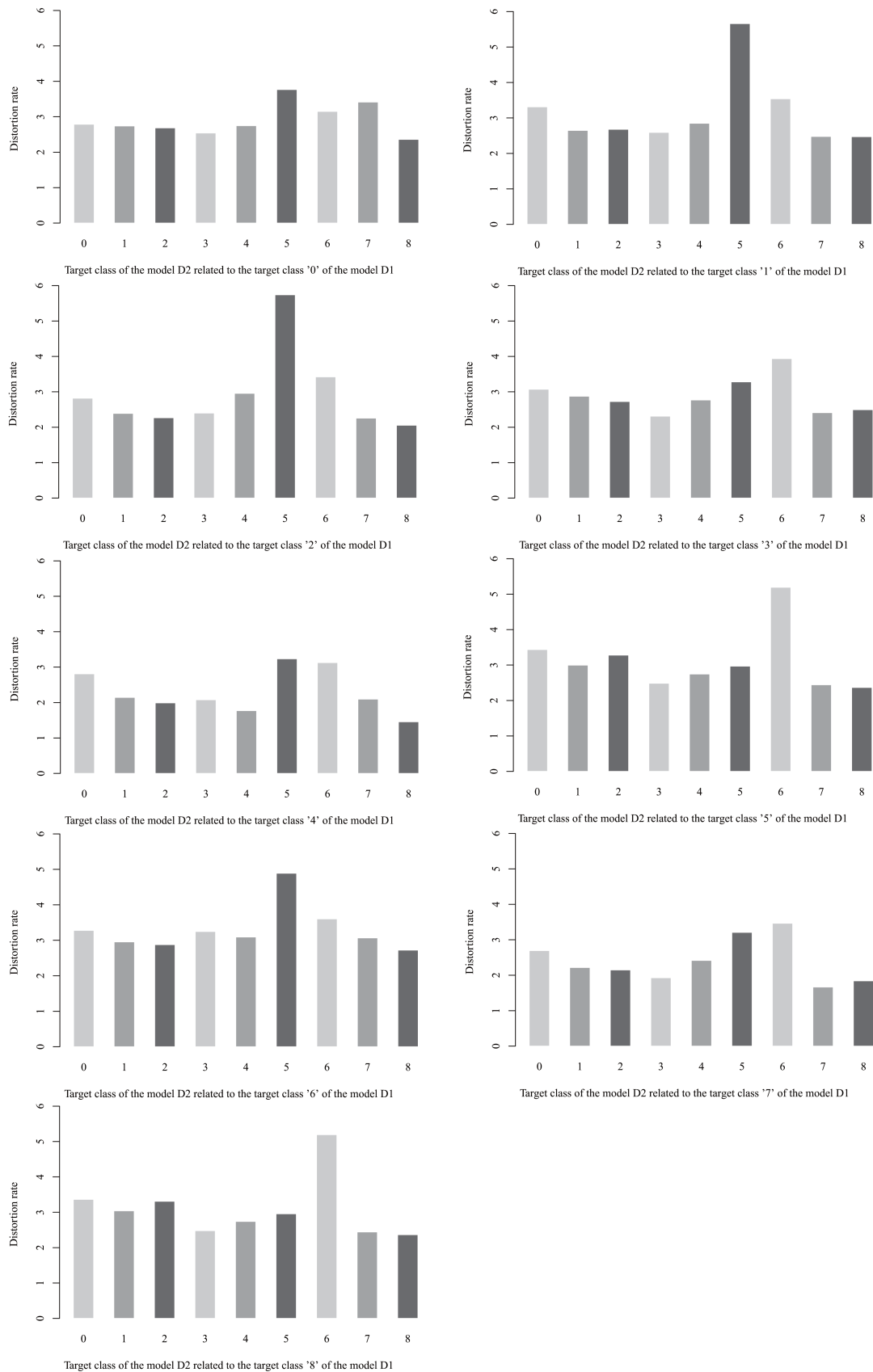



FIGURE 5. Average distortion of each target class in models  $D_1$  and  $D_2$  for the original sample "9" (1000 iterations).

**TABLE 7.** Class score of a multi-targeted adversarial example for each target class in models  $D_1$ ,  $D_2$ ,  $D_3$ ,  $D_4$ , and  $D_5$  in Table 6.

Description	Class score of a multi-targeted adversarial example
	
$D_1$ (targeted "9")	[1.287 -0.61 0.16 1.27 -1.56 -1.38 -1.48 1.10 1.19 <b>1.288</b> ]
$D_2$ (targeted "0")	[ <b>5.62</b> -5.46 3.70 4.59 -8.34 1.84 0.65 1.87 -0.67 -4.48]
$D_3$ (targeted "1")	[-2.58 <b>3.99</b> 3.15 3.96 -0.89 -1.67 2.52 -3.19 3.21 -7.87]
$D_4$ (targeted "2")	[-2.76 -0.27 <b>6.72</b> 4.45 -7.92 -4.55 -2.22 2.23 2.47 -5.06]
$D_5$ (targeted "3")	[-0.72 -1.00 5.06 <b>5.62</b> -6.87 1.68 -0.67 0.39 0.41 -4.79]

an attacker needs a multi-targeted adversarial example with minimal distortion.

For example, an attacker might try to attack the original sample "9" in Fig. 5. The attacker wants to generate a multi-targeted adversarial example that was misclassified as target class "8" by model  $D_1$  and would incorrectly classify it as a wrong class that is not the original class by model  $D_2$ . Since model  $D_2$  does not require targeting, the attacker chooses class "8" of model B.

#### b: Application

If we apply a multi-targeted adversarial example to an application, we can use it for road signs. For example, with a multi-targeted adversarial example, an altered left-hand turn sign can cause a Ford vehicle to turn to the right instead, deceive a GM vehicle into proceeding straight ahead, and deceive a Toyota vehicle into making a U-turn.

In addition, the multi-targeted adversarial example can be used for covert channels [38]. The sender generates a multi-targeted adversarial example that is misclassified by each target class for multiple models, but the multi-targeted adversarial example is classified correctly by censorship. Each target class for multiple models has hidden information sent over a covert channel.

#### c: Attack success rate

The goal of the proposed scheme is to attack known models. A white-box attack can be used because the attacker has detailed information about multiple models. By applying a modified version of the method of Carlini and Wagner [8], the proposed scheme is an optimized method to satisfy both minimum distortion and multi-model attack success. Therefore, the proposed scheme can generate multi-targeted adversarial examples for a 100% success rate of targeted attacks for each class against several models.

#### d: Method Considerations

Since the amount of loss for each multiple model is used in the process of generating multi-targeted adversarial examples, the loss weights of multiple models must be considered. For example, if the loss weight of model A increases, the probability of a successful attack on model A increases.

However, the probability of a successful attack on other models decreases inversely with the distortion. Therefore, appropriate loss weights should be considered to increase the probability of attack success for multiple models while minimizing the distortion of the multi-targeted adversarial example.

#### e: Distortion

The experimental results in Fig. 3 show an increase in average distortion correlating to the number of models. This is because, in the process of generating a multi-targeted adversarial example, a multi-targeted adversarial example must be misclassified by more models as each target class. Therefore, there is a trade-off between distortion and model scalability. When generating multi-targeted adversarial examples, the attacker must consider the trade-off relationship.

## VII. CONCLUSION

In this paper, we proposed a multi-targeted adversarial example that is misclassified by each of the multiple models as each target class, while minimizing the distance of the original sample. From the experimental results on the MNIST dataset, the multiple models misclassified the multi-targeted adversarial example as each target class with a 100% attack success rate. In addition, we discovered that distortion differed between each target class in multiple models. This information is useful for selecting target classes. We also found a trade-off between distortion and model scalability.

Future studies will extend the experiment to other standard image datasets, such as CIFAR10 [17] and ImageNet [9]. A correlation analysis between distortion and human perception in stealth is also included in a future work. In addition, the concept of a multi-targeted adversarial example can be applied to voice [5], [44], music [15], face [34], and CAPTCHA systems [29]. We will also work on generating a multi-targeted adversarial example not through transformation, but by applying a generative scheme, such as a generative adversarial example [12]. Finally, developing a countermeasure to the proposed scheme will provide another challenge.

## APPENDIX

See Tables 8–10.

**TABLE 8.**  $D_i$  ( $1 \leq i \leq n$ ) model architecture.

Layter type	MNIST shape
Convolution+ReLU	[3, 3, 32]
Convolution+ReLU	[3, 3, 32]
Max pooling	[2, 2]
Convolution+ReLU	[3, 3, 64]
Convolution+ReLU	[3, 3, 64]
Max pooling	[2, 2]
Fully connected+ReLU	[200]
Fully connected+ReLU	[200]
Softmax	[10]

TABLE 9.  $D_i$  ( $1 \leq i \leq n$ ) model parameter.

Parameter	MNIST model
Learning rate	0.1
Momentum	0.9
Dealy rate	-
Dropout	0.5
Batch size	128
Epochs	50

TABLE 10.  $D_i$  ( $1 \leq i \leq 5$ ) pretrained model.

Model	Training data	Rate
$D_1$	0~12,000	99.18%
$D_2$	12,000~24,000	99.17%
$D_3$	24,000~36,000	99.25%
$D_4$	36,000~48,000	99.22%
$D_5$	48,000~60,000	99.17%

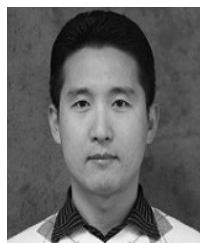
## REFERENCES

- [1] M. Abadi et al., "TensorFlow: A system for large-scale machine learning," in *Proc. OSDI*, vol. 16, 2016, pp. 265–283.
- [2] M. Barreno, B. Nelson, A. D. Joseph, and J. Tygar, "The security of machine learning," *Mach. Learn.*, vol. 81, no. 2, pp. 121–148, 2010.
- [3] B. Biggio, B. Nelson, and P. Laskov, "Poisoning attacks against support vector machines," in *Proc. 29th Int. Conf. Int. Conf. Mach. Learn.*, 2012, pp. 1467–1474.
- [4] C. M. Bishop, *Neural Networks for Pattern Recognition*. London, U.K.: Oxford Univ. Press, 1995.
- [5] N. Carlini et al., "Hidden voice commands," in *Proc. USENIX Secur. Symp.*, 2016, pp. 513–530.
- [6] N. Carlini and D. Wagner, "Adversarial examples are not easily detected: Bypassing ten detection methods," in *Proc. 10th ACM Workshop Artif. Intell. Secur.*, 2017, pp. 3–14.
- [7] N. Carlini and D. Wagner. (2017). "MagNet and 'efficient defenses against adversarial attacks' are not robust to adversarial examples." [Online]. Available: <https://arxiv.org/abs/1711.08478>
- [8] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 39–57.
- [9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2009, pp. 248–255.
- [10] A. Fawzi, O. Fawzi, and P. Frossard, "Analysis of classifiers' robustness to adversarial perturbations," *Mach. Learn.*, vol. 107, no. 3, pp. 481–508, 2015.
- [11] A. Fawzi, S.-M. Moosavi-Dezfooli, and P. Frossard, "The robustness of deep networks: A geometrical perspective," *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 50–62, Nov. 2017.
- [12] I. Goodfellow et al., "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.
- [13] I. J. Goodfellow, J. Shlens, and C. Szegedy. (2014). "Explaining and harnessing adversarial examples." [Online]. Available: <https://arxiv.org/abs/1412.6572>
- [14] G. Hinton et al., "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.
- [15] C. Kereliuk, B. L. Sturm, and J. Larsen, "Deep learning and music adversaries," *IEEE Trans. Multimedia*, vol. 17, no. 11, pp. 2059–2071, Nov. 2015.
- [16] D. P. Kingma and J. Ba. (2014). "Adam: A method for stochastic optimization." [Online]. Available: <https://arxiv.org/abs/1412.6980>
- [17] A. Krizhevsky, V. Nair, and G. Hinton. (2014). *The CIFAR-10 Dataset*. [Online]. Available: <http://www.cs.toronto.edu/kriz/cifar.html>
- [18] A. Kurakin, I. Goodfellow, and S. Bengio. (2016). "Adversarial examples in the physical world." [Online]. Available: <https://arxiv.org/abs/1607.02533>
- [19] A. Kurakin, I. J. Goodfellow, and S. Bengio. (2016). "Adversarial machine learning at scale." [Online]. Available: <https://arxiv.org/abs/1611.01236>
- [20] H. Kwon, Y. Kim, K.-W. Park, H. Yoon, and D. Choi, "Friend-safe evasion attack: An adversarial example that is correctly recognized by a friendly classifier," *Comput. Secur.*, 2018.
- [21] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
- [22] Y. LeCun, C. Cortes, and C. J. Burges. (2010). MNIST Handwritten Digit Database. AT&T Labs. [Online]. Available: <http://yann.lecun.com/exdb/mnist>
- [23] D. Meng and H. Chen, "MagNet: A two-pronged defense against adversarial examples," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2017, pp. 135–147.
- [24] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "DeepFool: A simple and accurate method to fool deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 2574–2582.
- [25] A. Mosca and G. D. Magoulas, "Hardening against adversarial examples with the smooth gradient method," *Soft Comput.*, vol. 22, no. 10, pp. 3203–3213, May 2018.
- [26] M. Mozaffari-Kermani, S. Sur-Kolay, A. Raghunathan, and N. K. Jha, "Systematic poisoning attacks on and defenses for machine learning in healthcare," *IEEE J. Biomed. Health Inform.*, vol. 19, no. 6, pp. 1893–1905, Nov. 2015.
- [27] N. Narodytska and S. Kasiviswanathan, "Simple black-box adversarial attacks on deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 1310–1318.
- [28] G. L. Oliveira, A. Valada, C. Bollen, W. Burgard, and T. Brox, "Deep learning for human part discovery in images," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2016, pp. 1634–1641.
- [29] M. Osadchy, J. Hernandez-Castro, S. Gibson, O. Dunkelman, and D. Pérez-Cabo, "No bot expects the DeepCAPTCHA! Introducing immutable adversarial examples, with applications to CAPTCHA generation," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 11, pp. 2640–2653, Nov. 2017.
- [30] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proc. ACM Asia Conf. Comput. Commun. Secur.*, 2017, pp. 506–519.
- [31] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *Proc. IEEE Eur. Symp. Secur. Privacy (EuroS&P)*, Mar. 2016, pp. 372–387.
- [32] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2016, pp. 582–597.
- [33] S. Potluri and C. Diedrich, "Accelerated deep neural networks for enhanced intrusion detection system," in *Proc. IEEE 21st Int. Conf. Emerg. Technol. Factory Autom. (ETFA)*, Sep. 2016, pp. 1–8.
- [34] A. Rozsa, M. Günther, E. M. Rudd, and T. E. Boulton, "Facial attributes: Accuracy and adversarial robustness," *Pattern Recognit. Lett.*, 2017.
- [35] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.
- [36] S. Shen, G. Jin, K. Gao, and Y. Zhang, "APE-GAN: Adversarial perturbation elimination with GAN," *ICLR Submission*, 2017.
- [37] K. Simonyan and A. Zisserman. (2014). "Very deep convolutional networks for large-scale image recognition." [Online]. Available: <https://arxiv.org/abs/1409.1556>
- [38] S. Zander, G. Armitage, and P. Branch, "A survey of covert channels and countermeasures in computer network protocols," *IEEE Commun. Surveys Tuts.*, vol. 9, no. 3, pp. 44–57, 3rd Quart., 2007.
- [39] T. Strauss, M. Hanselmann, A. Junginger, and H. Ulmer. (2017). "Ensemble methods as a defense to adversarial perturbations against deep neural networks." [Online]. Available: <https://arxiv.org/abs/1709.03423>
- [40] C. Szegedy et al. (2013). "Intriguing properties of neural networks." [Online]. Available: <https://arxiv.org/abs/1312.6199>
- [41] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel. (2017). "Ensemble adversarial training: Attacks and defenses." [Online]. Available: <https://arxiv.org/abs/1705.07204>
- [42] F. Tramèr, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel. (2017). "The space of transferable adversarial examples." [Online]. Available: <https://arxiv.org/abs/1704.03453>
- [43] F. Zhang, P. P. K. Chan, B. Biggio, D. S. Yeung, and F. Roli, "Adversarial feature selection against evasion attacks," *IEEE Trans. Cybern.*, vol. 46, no. 3, pp. 766–777, Mar. 2016.
- [44] G. Zhang, C. Yan, X. Ji, T. Zhang, T. Zhang, and W. Xu, "Dolphinattack: Inaudible voice commands," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2017, pp. 103–117.





**HYUN KWON** received the B.S. degree in mathematics from the Korea Military Academy, South Korea, in 2010, and the M.S. degree from the School of Computing, Korea Advanced Institute of Science and Technology, in 2015, where he is currently pursuing the Ph.D. degree. His research interests include information security, computer security, and intrusion tolerant system.



**YONGCHUL KIM** received the B.E. degree in electrical engineering from the Korea Military Academy, South Korea, in 1998, the M.S. degree in electrical engineering from the University of Surrey, U.K., in 2001, and the Ph.D. degree in electrical and computer engineering from the Department of Electrical and Computer Engineering, North Carolina State University, in 2007. He is currently a Professor with the Department of Electrical Engineering, Korea Military Academy.

His research interests include WiMAX and wireless relay networks.



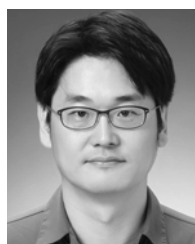
**KI-WOONG PARK** received the B.S. degree in computer science from Yonsei University, South Korea, in 2005, and the M.S. and Ph.D. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology in 2007 and 2012, respectively. He was a Senior Researcher with the National Security Research Institute. He is currently a Professor with the Department of Computer and Information Security, Sejong University. His research interests include security

issues for cloud and mobile computing systems as well as the actual system implementation and subsequent evaluation in a real computing system. He was a recipient of the 2009–2010 Microsoft Graduate Research Fellowship.



**HYUNSOO YOON** received the B.E. degree in electronics engineering from Seoul National University, South Korea, in 1979, the M.S. degree in computer science from the Korea Advanced Institute of Science and Technology (KAIST) in 1981, and the Ph.D. degree in computer and information science from The Ohio State University, Columbus, OH, USA, in 1988. From 1988 to 1989, he was a Technical Staff Member with AT&T Bell Labs. Since 1989, he has been a Faculty Member with the School of Computing, KAIST. His main research inter-

ests include wireless sensor networks, 4G networks, and network security.



**DAESEON CHOI** received the B.S. degree in computer science from Dongguk University, South Korea, in 1995, the M.S. degree in computer science from the Pohang Institute of Science and Technology, South Korea, in 1997, and the Ph.D. degree in computer science from the Korea Advanced Institute of Science and Technology, South Korea, in 2009. He is currently a Professor with the Department of Medical Information, Kongju National University, South Korea. His

research interests include information security and identity management.

...