Simple Black-Box Adversarial Examples Generation with Very Few Queries

Yuya SENZAKI†(a), Satsuya OHATA†(b), Nonmembers, and Kanta MATSUURA†††(c), Senior Member

SUMMARY Research on adversarial examples for machine learning has received much attention in recent years. Most of previous approaches are white-box attacks; this means the attacker needs to obtain before-hand internal parameters of a target classifier to generate adversarial examples for it. This condition is hard to satisfy in practice. There is also research on black-box attacks, in which the attacker can only obtain partial information about target classifiers; however, it seems we can prevent these attacks, since they need to issue many suspicious queries to the target classifier. In this paper, we show that a naive defense strategy based on surveillance of number query will not suffice. More concretely, we propose to generate not pixel-wise but block-wise adversarial perturbations to reduce the number of queries. Our experiments show that such rough perturbations can confuse the target classifier. We succeed in reducing the number of queries to generate adversarial examples in most cases. Our simple method is an untargeted attack and may have low success rates compared to previous results of other black-box attacks, but needs in average fewer queries. Surprisingly, the minimum number of queries (one and three in MNIST and CIFAR-10 dataset, respectively) is enough to generate adversarial examples in some cases. Moreover, based on these results, we propose a detailed classification for black-box attackers and discuss countermeasures against the above attacks.

key words: adversarial examples, black-box attack, deep learning

1. Introduction

The growth of neural networks and their related techniques have improved accuracy in various tasks. In particular, models based on Convolutional Neural Networks (CNNs) show good performance in the fields of image recognition etc. CNN based models have many applications such as autonomous driving [1], malware classification/detection [2], [3] etc. On the other hand, the spread of a technology to the real world also increases the benefit for attacking it. Image recognition by CNNs is not an exception. There are many scenarios in which adversaries can take advantage of causing misrecognition. For example in autonomous driving, the adversary may intentionally cause traffic accidents by modifying traffic signs. Therefore, we should study attacks on CNNs themselves and countermeasures against them to enable use of CNNs in many real-world scenarios. Security in machine learning has been studied for a long time (e.g., [4]). In particular, we focus on maliciously crafted data called Adversarial Examples [5] in this paper. Since this perturbation is relatively small, it does not affect recognition by humans; however, (machine) classifiers misrecognize them with high confidence.

There have been many methods for generating adversarial examples a lot [6]–[11]. Many of these assume that the information about the target classifier (loss function, structure, weight values, etc.) is known. In other words, we can regard these as a white-box attack. Considering more realistic scenarios, however, it is natural to assume that an attacker cannot obtain perfect knowledge about the classifier. We can classify these kind of attacks by the amount of obtainable information and this changes the difficulty of the attack. We will discuss them in Sect. 3.1.

In previous results on black-box attacks, various approaches have been proposed [12]–[18]. We start from a simple and direct approach such as [12] and focus on decreasing the number of queries* since we cannot ignore the cost of communications with the target classifier in many limited environments. The attacks may fail in practice due to time constraints even though succeed in theory. Therefore, the number of queries is an important factor in research on black-box attacks. However, reducing the number of queries in black-box attacks is not a trivial task. Even if we fix the intensity of the perturbation, the space we have to explore is quite huge; that is, when we add $\pm \epsilon$ to each pixel for a $D$-dimensional image, for example, the number of possible perturbations is $2^D$. Note that we cannot take a strategy that we pick a random perturbation, add it to the original image, query its crafted one to the target classifier, check whether we obtain adversarial examples or not by chance, and repeat this procedure until we succeed in generating the image we want to obtain. This is because CNNs are noise-tolerant, and the randomly chosen perturbation does not affect generating adversarial examples in most cases. In this paper, we propose a simple and query-efficient black-box attack and clarify how many queries are required to make a classifier misclassify in this setting. Moreover, we also discuss countermeasures against black-box attacks.

*Here, we call query the input fed to the classifier. We feed this input to the target model, and it returns a recognition result of it as an output.
1.1 Related Work

1.1.1 Generation of Adversarial Examples

Vulnerability against maliciously crafted samples in neural networks is initially indicated by Szegedy et al. [5] and they call these samples as adversarial examples. They reduce the generation of adversarial examples to a constrained optimization problem and solve it using box constraint L-BFGS. This method takes a long time for generating adversarial examples since it requires to solve an optimization problem many times. To solve this problem, Goodfellow et al. [7] propose a method for generating adversarial examples at high speed. This is called Fast Gradient Sign Method (FGSM). We will describe details of this method in Sect. 2.2. After that, various methods for generating adversarial examples have been proposed [6], [8]–[11]. There are some research directions (e.g., reducing the number of modified pixels, balancing the tradeoff between the success rate of attacks and the time we need for generating adversarial examples) in this field. Here, we note most of them assume that adversaries perfectly know inner information of target classifiers. The attack methodology that is adopted in above results is called a “white-box attack”. The white-box attack is strong; that is, it achieves high attack success rate, and we can obtain high-quality adversarial examples in most cases. However, it is not considered as real-world threats in general since it is difficult to fulfill the conditions we need for the attack in practice.

1.1.2 Generation with Limited Information

Several methods for generating adversarial examples without inner information of target classifiers have been proposed. We can classify those methods as one of the three following types; that are, constructing substitute model, estimating gradient, and genetic algorithm:

Construct substitute model [13], [14], [19] These methods are based on the transferability, which is as a property of adversarial examples. This property means the adversarial examples made for cheating one classifier also succeed in cheating other classifiers with high probability, even if their structures are different from the original one. This makes defense difficult since simple ensembles of countermeasures do not work well [20]. In the method proposed by Papernot et al. [13], [14], the attacker trains a substitute classifier that mimic the target classifier in its local environment and generates adversarial examples for it. Thanks to the transferability, the adversarial examples for the local (substitute) classifier are also effective for the target classifier. This substitute model is trained using queries issued by the attacker and their responses obtained by the target classifier as training data. Constructing substitute classifiers is known as model extraction attacks and black-box adversarial examples generation. Therefore, our black-box attack may lead to the efficient model extraction attack.

Estimate gradient [12], [15], [16], [18] Instead of taking advantage of the transferability property, these methods estimate gradients of the target classifier and use them for finding the appropriate perturbations.

Narodytska and Kasiviswanathan [12] propose an attack that estimates gradients using local-search based technique and finds the pixels called “critical pixels”, which effectively affect outputs of the target classifier. While the attacker needs to issue queries to the target classifier for many times to find such pixels, the amount of modified pixels is fewer than other methods.

Chen et al. [15] propose the attack called “ZOO”. They approximately solve the problem considered in (white-box) “C&W attack” [6] via zeroth-order optimization. While this method has a high success rate, it requires huge amount of queries. They also proposed the method for downsizing the attack-space to decrease the number of queries, which is similar to our method. Later, Tu et al. [18] propose “Auto-ZOOM”. This is an improvement of [15] for reducing the number of queries. Their methodology is using an autoencoder or bilinear mapping for resizing images, and this is similar to ours. Although the attack success rate of [18] is higher than ours, we show that our attack requires fewer queries than theirs via experiments.

A black-box attack proposed by Ilyas et al. [16] requires fewer queries than the above two approaches. Their approach is based on natural evolutionary strategies, which are methods designed for derivative-free optimization. The attacker can generate adversarial examples with fewer queries since we can efficiently estimate gradient in this strategy. In [16], they consider some scenarios in which the adversary can obtain partial information of the target classifier. Unlike to this method, our proposed attack is not targeted, and its lower complexity allows us to succeed with fewer queries.

Genetic algorithm [17] Alzantot et al. [17] propose “GenAttack”, which is based on genetic algorithms. This is one of the gradient-free optimization strategies, inspired by natural evolution. Since genetic algorithms do not require gradients, the attacker does not have to estimate them and can decrease the amount of queries. Also in the field of malware detection, Xu et al. [23] similarly propose the attack based on genetic algorithms. Our approach is more simple than above results since we do not use genetic algorithms and directly calculate perturbations.

1.1.3 Countermeasures for Adversarial Examples

Countermeasures against adversarial examples have also been actively studied. There are roughly two ways to oppose to adversarial examples. One is making classifiers more robust (e.g., [24], [25]), and the other is limiting the input to classifiers (typically by detectors) (e.g., [26], [27]); As many of the proposed defense methods have already been
broken [20], [28], [29], more effective defense approaches are required to counter adversarial examples.

1.2 Our Contribution

There are three contributions in this paper.

- we propose an extended attack model classification for black-box attacks.
- we propose a new method for generating adversarial examples in a black-box manner. More concretely, we propose not pixel-wise but block-wise perturbations for reducing the attack complexity.
- we discuss countermeasures against black-box attacks.

The technical overviews of these results are as follows:

Attack Model Classification

We propose an attack model classification for black-box attacks. We should consider the attack model of black-box attacks for generating adversarial examples since it drastically changes attack difficulty. Previous classification by Papernot et al. [9] only considers the quantity of information that an adversary can obtain. Considering countermeasures against attacks, however, we should include the quality of information that the adversary can obtain in the classification of the attack model. More concretely, regarding the outputs for queries from the target classifier, we separate the class as to whether the adversary can obtain raw information (means, confidence values themselves) or not. By doing so, we can remove an implicit assumption in the previous classification. Moreover, we treat network structure information and internal parameters (weights/biases) separately, since model extraction attacks (a black-box weights recovery attacks) have been proposed. Based on the above considerations, we propose a new attack model classification to treat black-box attacks and their countermeasures systematically. For more details, please see Sect. 3.1.

Method for generating adversarial examples

We propose a new method for generating adversarial examples in a black-box manner. We assume the black-box classifier returns confidence values with inference results. We show an overview of our method in Fig. 1. In FGSM, perturbations added to an image are either $+\varepsilon$ or $-\varepsilon$, which are calculated using a gradient of a loss function of the target classifier. In black-box attacks, however, we cannot use internal information itself. Therefore, previous work on black-box attacks effectively uses confidence values that are returned by the target classifier. We also adopt this strategy in this paper. More concretely, an adversary in our algorithm adds perturbation $+\varepsilon$ to an image and queries it to the target classifier. The adversary can check whether the sign of perturbation is correct or not by observing the change of the confidence value. By iterating this procedure for all pixels, the adversary succeed in generating adversarial examples in a black-box manner. For more details, please see Sect. 3.2.

The above method needs a large number of queries, since the size of an image is large in general. We generate adversarial examples via (standard) FGSM in our preliminary experiments and find the sign of perturbations is not uniformly random and there are areas added to the perturbations of the same sign. This fact implies that we may not need to estimate pixel-wise perturbations. Based on above observation, we extend our method for generating adversarial examples with fewer queries. More concretely, we estimate not pixel-wise perturbations but $N \times N$ block-wise ones. In this method, we need fewer queries for attacks compared to the original method; however, we can easily expect that the success rate for attacks decreases, especially if we set large $N$. Therefore, we evaluate the effectiveness of this method via empirical evaluations. Our experiments show that this attack variant succeeds in attacking even if we set slightly larger $N$. Surprisingly, the minimum number of queries (one and three in MNIST and CIFAR-10 dataset, respectively) are enough in some cases. For more details, please see Sect. 3.3 (method) and Sect. 4 (experiments).

Countermeasures

Based on the proposed attack model classification, we discuss the countermeasures against black-box attacks. Since black-box attacks, including ours, effectively use confidence values obtained from target classifiers, we can expect an impact for countermeasures that modify confidence values. We show some modification
propositions and discuss them. For more details, please see Sect. 5.2.

1.3 Paper Organization

In Sect. 2, we recall a background of this work. In Sect. 3, we explain our classification of black-box attacks and a novel black-box attack method. In Sect. 4, we show our experimental results. In Sect. 5, we discuss our attack and countermeasures for it. Section 6 is a summary of this work.

2. Preliminaries

2.1 Classification with Neural Networks

 Neural networks are based on a collection of units called perceptron. The output \( y \in \mathbb{R} \) from each unit corresponding to the input \( x \in \mathbb{R}^D \) is calculated as follows:

\[
y = w \cdot x + b
\]

Here, \( w \in \mathbb{R}^D \) and \( b \in \mathbb{R} \) are weights and biases, respectively. These parameters are optimized through learning. After this calculation, we apply a non-linear function (e.g., ReLU [30]) to \( y \). The output has strong non-linearity since neural networks are stacking of this unit.

When we use neural networks for classification tasks, we usually insert a softmax function as last layer. We can handle outputs of neural networks as probability distribution thanks to this function. Using above functions, for example, in \( c \)-class classification, we train a classifier such as it outputs \( y = \{y_0, y_1, ..., y_c\} \) where \( \sum_{i=0}^{c} y_i = 1, 0 \leq y_i \leq 1, \) and \( \arg\max(y) \) is the predicted class of input; that is, the predicted class for the input \( x \) is the argmax of the classifier’s output. Here, we can regard \( \max(y) \) as a confidence value of predictions. This value satisfies \( 0 \leq \max(y) \leq 1 \) and becomes larger when the classifier has confident for that precision. In this paper, we denote the output of model \( y = \{y_0, y_1, ..., y_c\} = F(x) \) and the confidence value (for the class \( t \)) \( F_t(x) = y_t \).

2.1.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are neural networks with convolutional layers. These layers convolve each region of input with filters of scalar values. Values in filters are optimized through learning. This is equivalent to applying Gaussian filters or edge extraction filters in the field of image processing. These layers are said to be effective not only for improving noise-robustness, but also for acquiring translation invariance.

2.2 Fast Gradient Sign Method

As described in Sect. 1, Fast Gradient Sign Method (FGSM) is a method for generating adversarial examples. In this method, we can generate adversarial examples at high speed since we avoid solving optimization problems. In FGSM, the perturbation \( \eta \) for the input \( x \) is calculated as follows:

\[
\eta = \epsilon \cdot \text{sign}(\nabla_x J(\theta, x, y))
\]

Here, \( \theta \) are parameters of the target classifier, \( x \) and \( y \) are an input and a target class, respectively, \( J \) is a loss function of the target classifier, sign is the sign function, and \( \epsilon \) is a parameter of modification strength. That is, FGSM uses the gradient of the loss function w.r.t the input \( x \) to calculate the perturbation. If we input samples with this perturbation to the target classifier, the loss corresponding to the class \( y \) increases; that is, it is harder to be predicted as the class \( y \) (= appropriate class).

3. Blackbox Attack

In this section, we explain our new (extended) classification of black-box attacks, method for FGSM emulation in a black-box manner via confidence values returned by target classifiers, and describe how to reduce the number of queries needed to successfully attack.

3.1 Classification of Attack Model

We can consider several classes for black-box attacks and their difficulty is correspondingly changed. Papernot et al. [9] proposed a classification of black-box attacks in terms of the information that the adversary can obtain. In their classification, the adversary may dispose the following information.

1. pairs of the bounded number of inputs/outputs
2. pairs of the unbounded number of inputs/outputs
3. training data
4. network architecture
5. both training data and network architecture

In the above classification, the adversary has advantage in descending order. Here, the network architecture includes both structure information about the target classifier (network structure and what activation functions are used) and its internal parameters (weights and biases). While their classification seems to consider all the configurations, the outputs of the target classifier (that are, the informations the adversary can obtain) are implicitly assumed to meet the two following conditions.

1. The outputs are not modified.
2. The outputs contain confidence values with the sufficient number of digits.

As we will describe in Sect. 5.2, it seems the most reasonable defense against black-box attacks so far is to return modified outputs of the classifier. Therefore, we should consider the scenario in which “the adversary can obtain modified outputs of the target classifier” to treat (future) defense and attack methods. In addition, the information of the target classifier (e.g., network structure) and its internal parameters (e.g., weights) should be separated since a model.
extraction attack (a black-box weights recovery attack) [21] has already been proposed. Therefore, it is meaningful to consider the scenario in which that “the adversary can obtain the information about the structure but cannot obtain the weights”. Moreover, it is also implicitly assumed that the adversary can feed arbitrary values as an input; however, this is not generally true. For instance, we cannot feed arbitrary values if the classifier takes input that is something extracted from images and we do not know how to extract it. In addition, we can also consider the scenario that the adversary can only modify the real-world object taken by camera [31].

Based on above discussions, we can consider the following factors to classify black-box attacks.

- Restriction on inputs (amount/preprocess)
- Postprocess on outputs
- Knowledge on data
- Knowledge of the target model’s structure

To investigate the relations between above factors and attack difficulty is an interesting open problem. We should consider whether the outputs of the target classifier are modified or not to appropriately treat the black-box attacks since previous black-box attacks (including our proposed attack) highly depend on the confidence value returned by the target classifier. We propose our classification in accordance with this observation and succeed in keeping our proposal simple. In the future, however, modification of inputs may largely affect the attack/defense method. In this scenario, it is meaningful to consider the classification considering the limitation of inputs by the adversary. This is also an interesting future work. We show our classification for black-box attacks as follows. In our classification, the adversary can obtain the following information.

1. **Bounded** number of outputs corresponding inputs
2. **Unbounded** number of **processed** outputs corresponding inputs
3. **Unbounded** number of **raw** outputs corresponding inputs

   (4a) (3) + training data
   (4b) (3) + network structure
   (5) All previous informations

Note that above proposed classification does not contain the class that the adversary can directly obtain internal parameters of the target classifier since it is for black-box attacks. In this paper, we propose the class 3 black-box attack. Although class 2, which is stronger black-box attack than class 3 attack have already been proposed [16], we focus on class 3 attack in this paper since it is enough practical scenario [21]. For simplicity, we focus on the untargeted attack in this paper. Extension of our attack to the targeted one is a future work.

3.2 Blackbox Emulation of FGSM

In this subsection, we consider how to generate FGSM-style adversarial examples in a black-box setting. In FGSM, perturbations for the input \( x \) are calculated using the gradients of the loss function of the target classifier. Once we focus on each pixels of \( x \), the perturbations are either \(+\epsilon\) or \(-\epsilon\). Here, \( \epsilon \) is a parameter for modification strength. Therefore, if we can estimate the sign of the perturbations in a black-box manner, we will succeed a black-box emulation of FGSM. Then we have to consider how to estimate the sign of the perturbations. When the adversary adds the perturbation \(+\epsilon\) for the pixel and queries that modified sample to the target classifier, the confidence value of the inference will be changed in most cases. If that modification decreases the confidence value, we can understand that the modified sample is closing to the FGSM-style adversarial examples. If the modification increases the confidence value of the inference, on the other hand, it means the modified sample is receding from the FGSM-style adversarial examples. In this case, we can understand that the sign of the perturbation is wrong and the adversary adds the perturbation \(-\epsilon\) for the pixel to close the sample to the adversarial examples\(^1\). We define the above sequence of operations as Procedure 1\(^{11}\). The adversary will be able to generate FGSM-style adversarial examples for the target classifier in a black-box manner by iterating the above procedure to all pixels of the input sample \( x \). In this algorithm, the number of queries to the target classifier is equal to the dimensions (sizes) of the input sample. This value tends to become very large since the number of dimensions of the input sample (mainly a picture) is large in general.

3.3 Reducing the Number of Queries

In Procedure 1 described in Sect. 3.2, the number of queries is equal to the dimensions of the input sample \( x \). We consider how to reduce it. In Procedure 1 we estimate the perturbation for all pixels by inquiring crafted sample \( x \) to the (black-box) classifier for \( D \) times. Here \( D \) means the di-
**Procedure 2** Calculation of block-wise perturbations

```
Input: Image x, Area area, Target Class t, Original Probability probt
Output: Perturbation for area of x
for i in area do
   xi = xi + ϵ
end for
prob_i = F_i(x)
if prob_i < probt then
   return ϵ
else
   return −ϵ
end if
```

**Algorithm 1** Our proposed method for generating adversarial examples

```
Input: Image x, Target Class t, Division Number N
Output: Adversarial examples x_{adv}
probt := F_t(x)
ptb := Zeros(x)
for area in Divide(x, N) do
   ptb_{area} := procedure2(x, area, t, probt)
end for
return x + ptb
```

dimensions of x. In this methodology, we divide input x into D areas to estimate the perturbation for all D areas. We call area a subset of pixels of the input x, such as all areas form a partition of x. The size of these areas is equal to each pixel. In other words, we can regard that Procedure 1 corresponds to the special case where the areas is limited to 1 pixel. It is our proposal that we divide the target image more roughly. More concretely, we try to see multiple pixels as one block and calculate the adversarial perturbation for that block. If we can generate the adversarial examples even if we divide the target image into larger blocks, it means we succeed in reducing the number of queries to execute black-box attacks. We extend Procedure 1 in this way and define it as Procedure 2. Using this Procedure 2, we propose our new method for generating adversarial examples as Algorithm 1. Zeros(x) is the array of all zeros with the same size as x, and Divide(x, N) is the array of indices corresponding to the pixel included in each area obtained by dividing x into N areas. In this method, the number of queries we need is N per trial. We note that, there are several strategies to select N depends on the scenario. For example, we start from the minimum N and sequentially increment it. If the query budget is given, using this budget as N or trying several N under any heuristic are also possible approaches which could be studied in future work.

We show the adversarial examples generated by our method in Fig. 2.

### 4. Empirical Evaluations

We evaluate the effectiveness of our proposed method using two popular datasets MNIST [32] and CIFAR-10 [33]. MNIST consists in 70000 28 × 28 × 1 dimensional grayscale handwritten digits in 10 classes. CIFAR-10 dataset consists in 60000 32 × 32 × 3 dimensional color image in 10 classes. Both datasets are widely used in this research field. We train the target classifiers for each of these two datasets. We show the structure of these classifiers in Table 1, 2, and 3. When we use MNIST/CIFAR-10 dataset, the accuracy of the classifiers is 99.0%/78.4%, respectively. After that, we generate FGSM-style adversarial examples in the white-box setting with ϵ = 0.2. In this setting, misclassification rates in MNIST/CIFAR-10 are 91.0%/76.8%, respectively.

Next, we try to generate adversarial examples for these classifiers using our proposed method described in Sect. 3.3. This time, at first we set the minimum value to N and increase it if we fail to generate adversarial examples. Therefore, the number of queries is not N but accumulation of N when we attempt different N continuously. Note that these

![Fig. 2](image-url)  
**Fig. 2** A demonstration of proposed method on CIFAR-10 dataset. The upper part is example of division number N = 3 × 3 × 3 (height×width×channels) and the lower part is example of N = 8 × 8 × 8. In both cases, we set ϵ = 0.03.

| Table 1 | Network structure of the classifier for MNIST dataset. |
|---------|--------------------------------------------------------|
| Layer Type | Number |
| Conv + ReLU | 5x5x32 |
| MaxPooling | 2x2 |
| Conv + ReLU | 5x5x64 |
| MaxPooling | 2x2 |
| FC + ReLU | 1024 |
| Softmax | 10 |

| Table 2 | Network structure of the classifier for CIFAR-10 dataset. |
|---------|--------------------------------------------------------|
| Layer Type | Number |
| Conv + ReLU | 3x3x64 |
| Conv + ReLU | 3x3x64 |
| MaxPooling | 2x2 |
| Conv + ReLU | 3x3x128 |
| Conv + ReLU | 3x3x128 |
| MaxPooling | 2x2 |
| FC + ReLU | 256 |
| FC + ReLU | 256 |
| Softmax | 10 |

| Table 3 | Parameters for learning of each model. |
|---------|---------------------------------------|
| Parameter | MNIST | CIFAR-10 |
| Learning Rate | 1e-4 | 1e-2 (decay 0.5) |
| Momentum | - | 0.9 (decay 0.5) |
| Delay Rate | - | 10 epochs |
| Dropout | 0.5 | 0.5 |
| Batch Size | 200 | 256 |
| Epochs | 10 | 50 |
classifiers return confidence values for each class. In our experiment, we consider untargeted attacks; that is, we regard the attack as success if the estimated label changes from the original one. Therefore, we excluded originally misclassified samples before the evaluation of our proposed method. We do sequentially attack attempts while increasing the number of divisions for each channel of the target image (i.e. \(1^2, 2^2, \ldots\) divisions per channel). Precisely, the number of samples for a number of \(Q\) queries indicated in Figs. 3 and 4 are the number of samples for which all attacks with \(1^2, 2^2, \ldots, Q - 1\) divisions failed, and the one with \(Q\) divisions solved in success. Note that the attack with maximum \(N\) corresponds to the black-box emulation of FGSM described in Sect. 3.2. We describe the algorithm for this experiment as Algorithm 2. Note that we cannot compare the query-efficiency of our approach with white-box attacks, as in the latter case, the attacker dispose of the classifier itself, so the consideration of query is not relevant.

4.1 Experiment Using MNIST Dataset

We show the result of our experiment using MNIST dataset in Fig. 3. In this experiment, we use 990 (out of 1000) samples that are correctly classified by the target classifier. This result indicates that we succeed in generating adversarial examples for most samples by issuing quite few queries (< 100). The mean of the number of queries required for attack’s success is 219.32, which is relatively low compared to the black-box emulation of FGSM (which corresponds to the maximum number of queries, i.e. 784). Moreover, the attack success rate at 784 queries, which is the maximum number of possible queries, is 90.6%. This is almost the same success rate as FGSM (91.0%), even though FGSM is a white-box attack.

4.2 Experiment Using CIFAR-10 Dataset

We show the result of our experiment using CIFAR-10 dataset in Fig. 4. In this experiment, we use 784 (out of 1000) samples that are correctly classified by the target classifier. We can see that the number of queries we need to generate adversarial examples is fewer than the maximum (\(32 \times 32 \times 3 = 3072\)) ones in many cases. This is the same
result as the case in experiment using MNIST dataset. The mean of the number of queries required for attack’s success is 185.47. And the maximum attack success rate is 68.6%, which is slightly lower than standard FGSM (76.8%).

5. Discussions

5.1 Results on Our Experiments

From our experimental results, we find our attack successfully creates adversarial examples in many cases. Moreover, although our strategy (block-wise perturbations) for decreasing the number of queries is simple, we succeed in reducing the number of queries drastically that we need for generating adversarial examples. These results mean that the black-box attack can certainly be a practical threat, and we should consider countermeasures against this attack more than ever. In the experiment using CIFAR-10 dataset, the maximum attack success rate is lower than that of standard FGSM. This is different from the experiments using MNIST dataset. A possible reason for this result is that perturbations between pixels are not independent. In our approach, we partly neglect the dependency of the perturbations between pixels to reduce the complexity of attacks; that is, the perturbation for one pixel (or block) should affect one for others, and vice versa. However, it seems this effect is relatively weak in practice, and we succeed in generating adversarial examples in many cases.

5.2 Countermeasures

We consider the most reasonable countermeasure against the black-box attacks including ours is modifying outputs not to return the raw outputs of target classifiers. Most of previous black-box attacks including ours use raw outputs of the target classifier. Therefore, we can expect it will be a good countermeasure against black-box attacks to force the target classifier to return modified outputs. This means that we push the adversary into the class 2 situation described in Sect. 3.1. We can consider various modification ways of outputs. Since that modification should not affect classification results for benign samples, it must not change the values argmax(F(x)). Considering this limitation, we propose the following modifications as examples. The target classifier returns

- **only labels** – The classifier returns only argmax(F(x)) instead of F(x).
- **noised confidence values** – The classifier returns F(x) + δ with appropriate δ instead of F(x).
- **confidence values with limited number of digits** – The classifier returns floor(F(x), d) with appropriate d instead of F(x).
- **confidence labels** – The classifier returns not confidence values themselves but confidence labels (e.g., “high”, “middle”, or “low”).

Here, floor(v, d) is a function that returns a vector v with d digits obtained by truncating each value. In above candi-
dates of countermeasure, “only labels” is clearly the most effective countermeasure from the viewpoint of security. While several recent results [34], [35] showed that even the attacker with only labels can succeed in generating adversarial examples, their attacks require a large number of queries; that is, this countermeasure at least increases the cost of attacks, and it increases the probability to protect the target classifier via other techniques (e.g., anomaly detection) from black-box attacks. Other countermeasures are also useful in practice since we can consider various scenarios and applications requiring confidence values. Also in such cases, we can expect that it remarkably raises the cost of attacks by reducing the amount of output information as necessary. Of course, other defense methods (for white-box attacks) that have already proposed (e.g., adversarial training [7]) can also be effective countermeasures against black-box attacks. In general, such ones require heavy computational costs in most cases. We should adopt the optimal defense method (including our proposal) by considering the tradeoff between costs and effects.

Remark: Recently, adversarial examples have actively been studied in the field of image recognition; however, one shall be cautious not to use the technology effective only for image processing; as adversarial examples are the problems not only for the image recognition tasks but for all tasks treated using machine learning. Therefore, we consider that one should keep in mind to adopt non-domain-specific techniques (e.g., the attack-defense techniques that are effective only for image recognition tasks) for the research on adversarial examples. Such a new attack-defense method will not lead to the essential solution of this problem.

6. Summary and Future Work

In this paper, we introduced a new classification of black-box attacks and proposed generation method of adversarial examples for black-box classifier with few queries. Moreover, we showed the effectiveness of our proposed method via empirical experiments, and achieved to lower the average number of queries compared to a black-box version of FGSM. We also discussed the countermeasures for black-box attacks.

While we treated untargeted black-box attacks, it is future work to extend our approach to targeted attacks. Although targeted attacks are more difficult than untargeted ones in general, they are more critical ones in many practical situations. Moreover, we consider it is also important to investigate class 1 and 2 situations from our classification described in Sect. 3.1 to confirm the effectiveness of countermeasures. Those attacks are more difficult to perform than class 3 attacks, as the adversary dispose of less informations, but in case of succeed, they may consequently cause harm to more machine learning systems and applications.

Acknowledgements

This work was partly supported by JSPS KAKENHI Grant Number 17KT0081 and JST CREST JPMJCR19F6.

References

[1] J. Janai, F. Güney, A. Behl, and A. Geiger, “Computer vision for autonomous vehicles: Problems, datasets and state-of-the-art,” arXiv preprint arXiv:1704.05519, 2017.

[2] B. Kolosnjaji, A. Zarras, G. Webster, and C. Eckert, “Deep Learning for Classification of Malware System Call Sequences,” 29th Australasian Joint Conference on Artificial Intelligence (AI), vol.9992, pp.137–149, Dec. 2016.

[3] J. Saxe and K. Berlin, “Deep neural network based malware detection using two dimensional binary program features,” 2015 10th International Conference on Malicious and Unwanted Software (MALWARE), pp.11–20, IEEE, 2015.

[4] M. Barreno, B. Nelson, A.D. Joseph, and J. Tygar, “The security of machine learning,” Machine Learning, vol.81, no.2, pp.121–148, 2010.

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

[6] N. Carlini and D. Wagner, “Towards evaluating the robustness of neural networks,” 2017 IEEE Symposium on Security and Privacy (S&P), pp.39–57, IEEE, 2017.

[7] I.J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” International Conference on Learning Representations (ICLR), 2015.

[8] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, “Deepfool: a simple and accurate method to fool deep neural networks,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp.2574–2582, 2016.

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

[10] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” arXiv preprint arXiv:1706.06083, 2017.

[11] 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.

[12] N. Narodytska and S. Kasiviswanathan, “Simple black-box adversarial perturbations for deep networks,” 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp.1310–1318, 2017.

[13] N. Papernot, P. McDaniel, and I. Goodfellow, “Transferability in machine learning: from phenomena to black-box attacks using adversarial samples,” arXiv preprint arXiv:1605.07277, 2016.

[14] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z.B. Celik, and A. Swami, “Practical black-box attacks against machine learning,” Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, pp.506–519, ACM, 2017.

[15] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh, “Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models,” Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, pp.15–26, ACM, 2017.

[16] A. Ilyas, L. Engstrom, A. Athalye, and J. Lin, “Black-box adversarial attacks with limited queries and information,” International Conference on Machine Learning, pp.2142–2151, 2018.

[17] M. Alzantot, Y. Sharma, S. Chakraborty, H. Zhang, C.-J. Hsieh,
and M.B. Srivastava, “Genattack: Practical black-box attacks with gradient-free optimization,” arXiv preprint arXiv:1805.11090, 2018.
[18] C.-C. Tu, P. Ting, P.-Y. Chen, S. Liu, H. Zhang, J. Yi, C.-J. Hsieh, and S.M. Cheng, “Autozoom: Autoencoder-based zeroth order optimization method for attacking black-box neural networks,” arXiv preprint arXiv:1805.11770, 2018.
[19] M. Juuti, S. Szyller, S. Marchal, and N. Asokan, “Prada: Protecting against dnn model stealing attacks,” 2019 IEEE European Symposium on Security and Privacy (EuroS&P) (to appear), pp.512–527, IEEE, 2019.
[20] W. He, J. Wei, X. Chen, N. Carlini, and D. Song, “Adversarial example defenses: Ensembles of weak defenses are not strong,” 11th USENIX Workshop on Offensive Technologies (WOOT 17), 2017.
[21] F. Tramèr, F. Zhang, A. Juels, M.K. Reiter, and T. Ristenpart, “Stealing machine learning models via prediction apis,” USENIX Security Symposium, pp.601–618, 2016.
[22] M. Kesarwani, B. Mukhoty, V. Arya, and S. Mehta, “Model extraction warning in mlaas paradigm,” Proceedings of the 34th Annual Computer Security Applications Conference (ACSAC), pp.371–380, ACM, 2018.
[23] W. Xu, Y. Qi, and D. Evans, “Automatically evading classifiers: A case study on pdf malware classifiers,” Proceedings of the 2016 Network and Distributed Systems Symposium, 2016.
[24] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, “Distillation as a defense to adversarial perturbations against deep neural networks,” 2016 IEEE Symposium on Security and Privacy (SP), pp.582–597, IEEE, 2016.
[25] F. Tramèr, A. Kurakin, N. Papernot, D. Boneh, and P. McDaniel, “Ensemble adversarial training: Attacks and defenses,” International Conference on Learning Representations (ICLR), 2018.
[26] K. Grosse, P. Manoharan, N. Papernot, M. Backes, and P. McDaniel, “On the (statistical) detection of adversarial examples,” arXiv preprint arXiv:1702.06280, 2017.
[27] D. Meng and H. Chen, “Magnet: a two-pronged defense against adversarial examples,” Proceedings of the 2017 ACM Conference on Computer and Communications Security, pp.135–147, ACM, 2017.
[28] N. Carlini and D. Wagner, “Adversarial examples are not easily detected: Bypassing ten detection methods,” Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, pp.3–14, ACM, 2017.
[29] N. Carlini and D. Wagner, “Magnet and “efficient defenses against adversarial attacks” are not robust to adversarial examples,” arXiv preprint arXiv:1711.08478, 2017.
[30] V. Nair and G.E. Hinton, “Rectified linear units improve restricted boltzmann machines,” Proceedings of the 27th international conference on machine learning (ICML-10), pp.807–814, 2010.
[31] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok, “Synthesizing robust adversarial examples,” International Conference on Machine Learning, pp.284–293, 2018.
[32] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” Proceedings of the IEEE, vol.86, no.11, pp.2278–2324, 1998.
[33] A. Krizhevsky and G. Hinton, “Learning multiple layers of features from tiny images,” Tech. Report, 2009.
[34] M. Cheng, T. Le, P.Y. Chen, J. Yi, H. Zhang, and C.J. Hsieh, “Query-efficient hard-label black-box attack: An optimization-based approach,” International Conference on Learning Representations (ICLR), 2019.
[35] W. Brendel, J. Rauber, and M. Bethge, “Decision-based adversarial attacks: Reliable attacks against black-box machine learning models,” International Conference on Learning Representations (ICLR), 2018.