Superpixel Attack

Enhancing Black-Box Adversarial Attack with Image-Driven Division Areas

Issa Oe[1(✉)], Keiichiro Yamamura[1], Hiroki Ishikura[1], Ryo Hamahira[1], and Katsuki Fujisawa[2]

[1] Graduate School of Mathematics, Kyushu University, Fukuoka, Japan
issa-oe@kyudai.jp
[2] Institute of Mathematics for Industry, Kyushu University, Fukuoka, Japan

Abstract. Deep learning models are used in safety-critical tasks such as automated driving and face recognition. However, small perturbations in the model input can significantly change the predictions. Adversarial attacks are used to identify small perturbations that can lead to misclassifications. More powerful black-box adversarial attacks are required to develop more effective defenses. A promising approach to black-box adversarial attacks is to repeat the process of extracting a specific image area and changing the perturbations added to it. Existing attacks adopt simple rectangles as the areas where perturbations are changed in a single iteration. We propose applying superpixels instead, which achieve a good balance between color variance and compactness. We also propose a new search method, versatile search, and a novel attack method, Superpixel Attack, which applies superpixels and performs versatile search. Superpixel Attack improves attack success rates by an average of 2.10% compared with existing attacks. Most models used in this study are robust against adversarial attacks, and this improvement is significant for black-box adversarial attacks. The code is available at https://github.com/oe1307/SuperpixelAttack.git.

Keywords: adversarial attack · security for AI · computer vision · deep learning

1 Introduction

Deep learning models have recently found applications in automatic driving and face recognition tasks. These tasks are critical for safety, involving potential risks to life and information privacy. It has been observed that even small perturbations to the model input can significantly alter predictions [26], leading to worst-case scenarios like accidents in automatic driving or information leakage in face recognition. Adversarial attacks are used to identify such perturbations that cause misclassifications. To counter these attacks, defense methods such as adversarial training [28,32] and adversarial detection [3,19] have been explored. However, more potent attacks are needed to develop more effective defenses.

T. Liu et al. (Eds.): AI 2023, LNAI 14471, pp. 141–152, 2024.
https://doi.org/10.1007/978-981-99-8388-9_12

This study targets black-box adversarial attacks, which operate under real-world constraints where only the model's predictions can be accessed. We focus on black-box adversarial attacks that aim to maximize attack success rates within allowed perturbations. A promising approach is to repeat the process of extracting a specific image area and changing the perturbations added to it.

Existing attacks use simple rectangles as the areas where perturbations are changed in a single iteration (Sect. 3.1). However, it is natural to determine the areas based on the image's color information, as it directly influences the perturbation to be added. Therefore, we focus on the color variance of the area where perturbations are changed in a single iteration (Sect. 3.2). Additionally, we focus on the compactness of the area, because existing attacks have adopted rectangles (Sect. 3.3). Through our analysis of the relationship among color variance, compactness, and attack success rates (Sect. 3.4), we discovered that areas that are compact and have a low color variance result in higher attack success rates (Sect. 3.5). Consequently, we propose applying superpixels, which achieve a good balance between color variance and compactness.

Additionally, we introduce versatile search, a new search method that restricts the search to the boundary of perturbation and allows for searches using areas beyond rectangles. With these advancements, we propose Superpixel Attack, a novel attack method that applies superpixels and performs versatile search (Sects. 4.1 and 4.2). To evaluate the performance of Superpixel Attack, we conducted comparison experiments with existing attacks using 19 models trained on the ImageNet dataset [13] and available on RobustBench [8] (Sect. 5). Superpixel Attack significantly enhances attack success rates, resulting in an average improvement of 2.10% compared to existing attacks. Considering that most models used in this study are robust against adversarial attacks, this improvement becomes especially noteworthy for black-box adversarial attacks. Our contributions can be summarized as follows:

1. We analyze the relationship among the color variance, compactness, and attack success rates.
2. We propose applying superpixels to black-box adversarial attacks and a new search method called versatile search.
3. We conducted comparison experiments on Superpixel Attack, which applies superpixel and performs versatile search, and found improvement in attack success rates by an average of 2.10% compared to existing attacks.

2 Preliminaries

2.1 Problem Definition

Let $H \in \mathbb{N}$ be the height, $W \in \mathbb{N}$ be the width, and $C \in \mathbb{N}$ be the number of color channels of the input image. Let $\mathcal{D} = [0,1]^{H \times W \times C}$ denote the image space, $Y \in \mathbb{N}$ denote the number of classes of the model, and $f : \mathcal{D} \to [0,1]^Y$ denote the classification model. The output of f is the predicted probability of each class, and we denote $f_i(x) \in [0,1]$ the predicted probability of class i when image

$x \in \mathcal{D}$ is the input. Adversarial attacks are to find an image $x_{adv} \in \mathcal{D}$ with the predicted label differs from the ground truth label $y \in \{1, \ldots, Y\}$ of the original image $x_{org} \in \mathcal{D}$ by adding perturbations that are imperceptible to humans. The inputs generated by adversarial attacks are called adversarial examples. This study focuses on adversarial attacks that maximize attack success rates within the allowed perturbations. We set the allowed perturbation size $\epsilon \in \mathbb{R}^+$ and the loss function $L : [0,1]^Y \times \{1, \ldots, Y\} \to \mathbb{R}$, and solve the following constrained nonlinear optimization problem:

$$\begin{aligned} \max_{x_{adv} \in \mathcal{D}} \quad & L\left(f(x_{adv}), y\right) \\ \text{s.t.} \quad & ||x_{adv} - x_{org}||_\infty \leq \epsilon \end{aligned} \tag{1}$$

2.2 Related Work

Parsimonious attack [20], Square Attack [4], and SignHunter [2] have been proposed as black-box adversarial attacks defined by Eq. (1). Parsimonious attack restricts the search space to the boundaries of allowed perturbations because attacks mostly succeed even on the boundaries. Square Attack achieves high success rates despite its reliance on random sampling. It is a part of AutoAttack [10], a well-known white-box adversarial attack. SignHunter searches for adversarial examples by repeating image division and gradient direction estimation.

Black-box adversarial attacks that minimize perturbations under misclassification [22,27] and those that reduce the number of perturbed pixels [9,11] have also been investigated. Attacks that generate adversarial examples from gradient information of surrogate models have also been proposed [21,31]. These methods are based on transferability, that is, adversarial examples of one model often become those of others. However, training is required to make surrogate models resemble an attacking model and incurs high computational costs.

3 Research on Update Areas

3.1 Update Areas of Existing Methods

The most promising approach for black-box adversarial attacks defined by Eq. (1) involves searching for adversarial examples by repeating the following steps: i. Extract a specific area from the image, ii. Collectively change the perturbation added to the extracted area, iii. Calculate the value of the loss function and update the perturbations when the loss increases. In this paper, we refer to the area where perturbations are changed in a single iteration as *Update Area*. Existing black-box adversarial attacks have adopted simple rectangles as Update Areas. Parsimonious attack sets them using squares that divide the image equally. Square Attack sets them using randomly sampled squares from a uniform distribution. SignHunter sets them using rectangles that divide the image into equal horizontal sections.

3.2 Color Variance of Update Areas

As described in the previous section, Update Areas of the existing attacks are set using simple rectangles. However, it is natural to determine the area by considering the color information of the image because it determines the perturbation to be added. Therefore, we focus on the color variance of Update Areas. As a metric to express the color variance in divided areas of an image, Intra-Cluster Variation (ICV) [5] is proposed. ICV is calculated based on the following equation:

$$\mathrm{ICV} = \frac{1}{\#\tilde{S}} \sum_{s \in \tilde{S}} \frac{\sqrt{\sum_{p \in s} (I(p) - \mu(s))^2}}{|s|}, \tag{2}$$

where $\tilde{S}$ is the set of image segmentations. In this paper, it refers to the set of all Update Areas used in an attack. $s \in \tilde{S}$ denotes a single Update Area, and $p \in s$ denotes a pixel. $I(p)$ is the value of the pixel p in the LAB color space[1] and $\mu(s)$ is the average value in the LAB color space within a single Update Area. $\#\tilde{S}$ is the number of Update Areas and $|s|$ is the number of pixels in a single Update Area. Smaller ICV indicates smaller color variance in each Update Area.

3.3 Compactness of Update Areas

Furthermore, considering that existing attacks use rectangles to set Update Areas, we focus on the compactness of Update Areas. The compactness (CO) [24] is a metric calculated by dividing the size of the segments by that of a circle with the same perimeter length. The following equation defines this:

$$\mathrm{CO} = \frac{\sum_{s \in \tilde{S}} Q(s) \cdot |s|}{\sum_{s \in \tilde{S}} |s|}, \qquad Q(s) = \frac{4\pi |s|}{|R(s)|^2}, \tag{3}$$

where $|R(s)|$ is the perimeter length of the Update Areas (number of pixels on the boundary). Higher CO indicates more centrally clustered Update Areas. We examined ICV and CO and attack success rates for various Update Areas construction in Sect. 3.5.

3.4 Superpixel Calculated by SLIC

Superpixel is a set of pixels that are close in color and position. They have applications in object detection [30], semantic segmentation [16], and depth estimation [7]. Dong et al. proposed a white-box adversarial attack that adds the same perturbation to each superpixel to avoid disrupting the local smoothness of a natural image [14]. We use superpixels to improve the efficiency of black-box adversarial attacks. To the best of our knowledge, no black-box adversarial attacks that apply superpixels have been proposed. Various methods have been proposed for computing superpixels. We use one of the most popular methods:

[1] LAB color spaces in this paper refer to CIELAB (L, a*, b*) color space.

Simple Linear Iterative Clustering (SLIC) algorithm [1]. It places representative points at equal intervals according to the maximum number of segments and clusters pixels based on the k-means method. Let (h_i, w_i) and (h_j, w_j) be the positions in the image, and (l_i, a_i, b_i) and (l_j, a_j, b_j) be the values in the LAB color space. Clustering is performed based on similarity k.

$$\begin{aligned} k_{color} &= \sqrt{(l_i - l_j)^2 + (a_i - a_j)^2 + (b_i - b_j)^2} \\ k_{space} &= \sqrt{(h_i - h_j)^2 + (w_i - w_j)^2} \\ k &= \max(0,\ k_{color} + \alpha \cdot k_{space}), \end{aligned} \tag{4}$$

where α is a hyperparameter that weighs the positional distance relative to the color distance. $\alpha = 10$ is generally set to calculate superpixels. We examine the relationship between ICV, CO, and attack success rates for $\alpha = \pm 0.1, \pm 1, \pm 10, \pm 100, \pm 1000$ in Sect. 3.5. In addition, the SLIC implementation of scikit-image has the option to force each superpixel to be connected. The experiment in Sect. 3.5 examine both cases. For $\alpha = 1000$, Update Areas are constructed as squares that divide the image equally, regardless of whether they are forced to be connected.

3.5 Analysis of Color Variance and Compactness

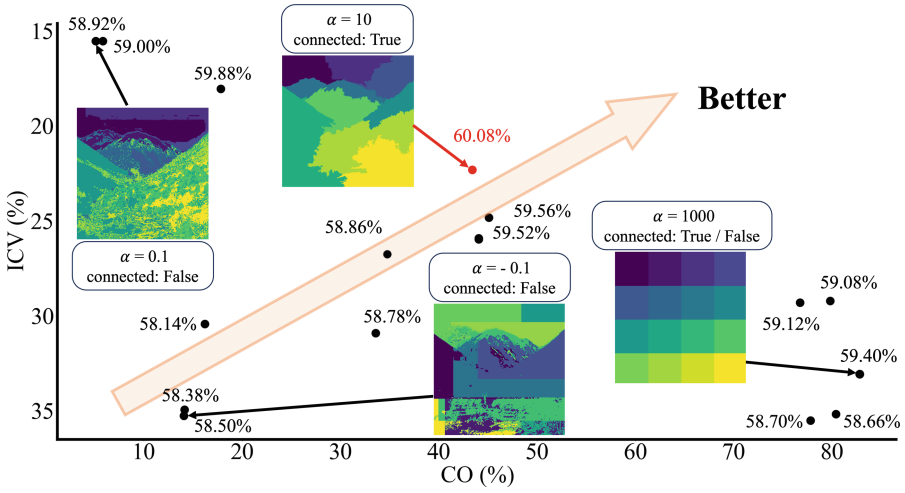

Fig. 1. Relationship between ICV, CO and attack success rates

The experiments use Salman et al. (ResNet-18) [23] trained on the ImageNet dataset and available on RobustBench. According to the RobustBench settings, we use 5,000 images randomly sampled from the ImageNet dataset, and the

allowed perturbation size is set to $\epsilon = 4/255$. We adopted versatile search, a new search method proposed in Sect. 4.2. We examine attack success rates at the maximum iterations $T = 500$ for each Update Area construction. The attack success rate is calculated as follows: (number of misclassified images after the attack)/(total number of images), where the higher the attack success rate, the more powerful the attack. The seed value is fixed at 0. We used a CPU: Intel(R) Xeon(R) Gold 5220R CPU@2.20 GHz×2, GPU: Nvidia RTX A6000, RAM:768 GB. The results are shown in Fig. 1.

Each point in Fig. 1 represents the values of CO and ICV for different Update Area construction. The numerical values represent the attack success rate at the point. The horizontal axis represents the value of CO, and the right side indicates that more centrally clustered Update Areas are constructed. The vertical axis represents the value of ICV, where the upper side indicates that Update Areas with lower color variance are constructed. Note that the same Update Areas are constructed for some parameters of α, and the points with equal ICV, CO, and attack success rates coincided with each other. For some representative points, the Update Areas generated by the SLIC algorithm are shown in different colors. This result indicates that it is effective to set Update Areas that are compact and have a low color variance.

4 Superpixel Attack

Based on the analysis in Sect. 3, we consider applying superpixels, which achieve a good balance between color variance and compactness, to black-box adversarial attacks. In this section, we describe the construction of Update Areas using superpixels (Sect. 4.1) and a new search method called versatile search (Sect. 4.2). We propose a novel attack method called *Superpixel Attack* that sets Update Areas using superpixels and performs versatile search. An overview of Superpixel Attack is shown in Fig. 2, and the pseudo-code is shown in Algorithm 1.

4.1 Update Areas Using Superpixels

Below, we describe the construction of Update Areas using superpixels. Inspired by existing attacks, Update Areas are set using a few segments of superpixels at an early stage and many segments of superpixels as the attack progresses. Specifically, the segment ratio r is given and superpixels $\mathcal{S}$ are computed following the maximum number of segmentations $n = r^j$ $(j = 1, 2, \dots)$. Let S be the set of Update Areas constructed for each maximum number of segments n. The original image x_{org} is divided into superpixels $\mathcal{S}$ for each RGB color channel $\{1, \dots, C\}$, which are set as Update Areas $S = \mathcal{S} \times \{1, \dots, C\}$. Note that the maximum number of superpixel segments n is not always equal to the number of superpixels computed $\#\mathcal{S}$ in the SLIC algorithm employed in this study. The segment ratio is set to $r = 4$ based on pre-examination. We set $\alpha = 10$ and force the areas to be connected.

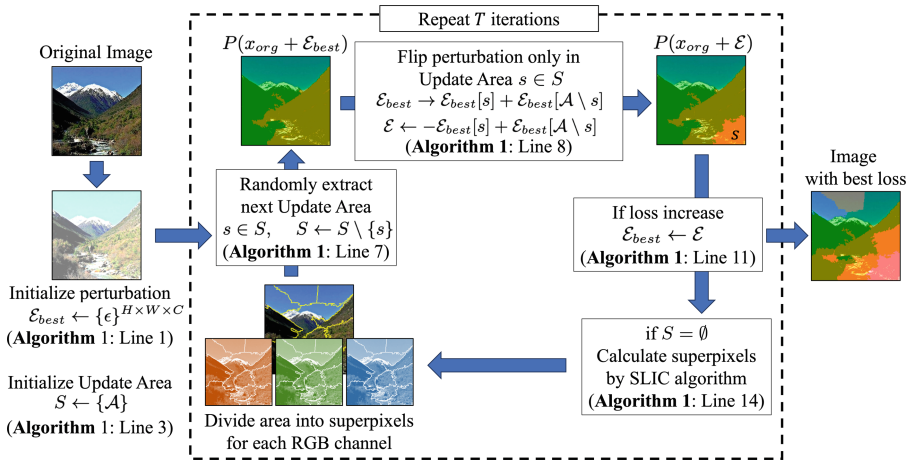

Fig. 2. Flow of proposed method: Superpixel Attack

Algorithm 1. Superpixel Attack

Input: Image height $H \in \mathbb{N}$, Image width $W \in \mathbb{N}$, Number of color channels $C \in \mathbb{N}$, Allowed perturbation size $\epsilon \in \mathbb{R}^+$, Maximum iterations $T \in \mathbb{N}$, Original image $x_{org} \in \mathcal{D}$, Ground truth label $y \in \{1, \ldots, Y\}$, Segments ratio $r \in \mathbb{N}$, Classification model $f : \mathcal{D} \rightarrow [0, 1]^Y$, Loss function $L : [0, 1]^Y \times \{1, \ldots, Y\} \rightarrow \mathbb{R}$, Projection function $P : \mathbb{R}^{H \times W \times C} \rightarrow \mathcal{D}$

Output: Image with best loss x_{best}

$\mathcal{E}_{best} \leftarrow \{\epsilon\}^{H \times W \times C}$ Initialize perturbation
$\mathcal{A} \leftarrow \{(h, w, c) | h \in [1, H], w \in [1, W], c \in [1, C]\}$ Entire area of image
$S \leftarrow \{\mathcal{A}\}$ Initialize Update Area
$\mathcal{L}_{best} \leftarrow -\infty$ Best loss
$n \leftarrow 1$ Maximum number of superpixel
for $t = 1, 2, \ldots, T$ **do**
  $s \in S, \quad S \leftarrow S \setminus \{s\}$ Randomly extract next Update Area
  $\mathcal{E}_{best} \rightarrow \mathcal{E}_{best}[s] + \mathcal{E}_{best}[\mathcal{A} \setminus s], \quad \mathcal{E} \leftarrow -\mathcal{E}_{best}[s] + \mathcal{E}_{best}[\mathcal{A} \setminus s]$
    Flip perturbation only in Update Area $s \in S$
  $\hat{x} \leftarrow P(x_{org} + \mathcal{E}), \quad \mathcal{L} \leftarrow L(f(\hat{x}), y)$
  if $\mathcal{L} \geq \mathcal{L}_{best}$ Loss increase **then**
    $\mathcal{L}_{best} \leftarrow \mathcal{L}, \quad \mathcal{E}_{best} \leftarrow \mathcal{E}$
  end if
  if $S = \emptyset$ All areas are searched **then**
    $n \leftarrow n \times r, \quad \mathcal{S} \leftarrow SLIC(x_{org}, n)$ Calculate superpixels by SLIC algorithm
    $S \leftarrow \mathcal{S} \times \{1, \ldots, C\}$ Divide area into superpixels for each RGB channel
  end if
end for
$x_{best} \leftarrow P(x_{org} + \mathcal{E}_{best})$

4.2 Procedure of Versatile Search

Below, we describe a new search method called *versatile search*. It searches only the boundaries of the allowed perturbations $\{-\epsilon, \epsilon\}^{H \times W \times C}$ according to the analysis by Moon et al. [20]. At the beginning of the search, the perturbations are initialized with $\mathcal{E}_{best} = \{\epsilon\}^{H \times W \times C}$. Let $\mathcal{A}$ be the entire area of the image and initialize the set of Update Areas with $S = \{\mathcal{A}\}$. The best loss is initialized as $\mathcal{L}_{best} = -\infty$. The following steps are repeated until the number of iterations t reaches the maximum iterations T.

First, the next area where the perturbations are changed is randomly extracted $s \in S$. In the first iteration, Update Area is set to the entire image ($s = \mathcal{A}$). Only the perturbations in the extracted Update Area $\mathcal{E}_{best}[s]$ is flipped to generate new perturbations $\mathcal{E}$. These perturbations $\mathcal{E}$ are added to the original image x_{org}, and the loss $\mathcal{L}$ is calculated. When the calculated loss $\mathcal{L}$ is higher than the best loss $\mathcal{L}_{best}$, the best loss $\mathcal{L}_{best}$ and the perturbation $\mathcal{E}_{best}$ are updated. Superpixels are computed when all Update Areas are searched ($S = \emptyset$), and new Update Areas are set using them.

When the attack is completed, the image with the best loss x_{best} is returned. Superpixel Attack employs CW loss [6] (L_{cw}) as the loss function based on preexamination. CW loss is calculated as follows:

$$L_{cw}(f(x), y) = \max_{i \neq y} f_i(x) - f_y(x) \tag{5}$$

5 Experiments

In this section, we describe the comparison experiments conducted to confirm the performance of Superpixel Attack. We compare it to Parsimonious attack (Parsimon) [20], Square Attack (Square) [4], SignHunter (SignH) [2], and Accelerated SignHunter (AccSignH) [17] as a baseline. All of these are black-box adversarial attacks with the same problem settings. The experiments use 19 models trained on the ImageNet dataset and available on RobustBench. According to the RobustBench settings, we use 5,000 images randomly sampled from the ImageNet dataset, and the allowed perturbation size is set to $\epsilon = 4/255$. We examine the attack success rates at the maximum iterations $T = 100$ and 1000. The baseline hyperparameters are the same as those in the original paper. The seed value is fixed at 0. We use the same computational environment as in Sect. 3.5. Table 1 presents the results. The highest attack success rate for each iteration is bolded, and the difference between the best baseline method and Superpixel Attack is noted on the right side.

The results in Table 1 show that Superpixel Attack improves the attack success rates by an average of 1.65% for 100 iterations and 2.10% for 1000 iterations compared to existing attacks. Most models used in this study are robust against adversarial attacks, and this improvement is significant for black-box adversarial attacks. In fact, the difference between the second-best and next-best existing attacks averaged 0.67% for 100 iterations and 0.71% for 1000 iterations. For

Table 1. Comparison experiments with baselines

100 iter		Attack Success Rate (%)					
source	Architecture	Parsimon	Square	SignH	AccSignH	**Superpixel**	diff
Wong [29]	ResNet-50	48.32	49.10	50.86	49.48	**53.86**	3.00
Engstrom [15]	ResNet-50	42.40	41.68	42.92	42.08	**45.26**	2.34
Salman [23]	ResNet-50	41.24	40.42	41.98	41.06	**44.44**	2.46
Salman	ResNet-18	52.08	51.50	52.58	52.06	**56.06**	3.48
Salman	WideResNet-50-2	36.84	35.64	37.82	36.54	**39.84**	2.02
PyTorch[a]	ResNet-50	33.92	47.56	**50.08**	38.80	47.52	-2.52
Debenedetti [12]	XCiT-S12	31.72	30.66	32.36	31.64	**33.86**	1.50
Debenedetti	XCiT-M12	30.36	29.38	31.06	30.14	**32.84**	1.78
Debenedetti	XCiT-L12	30.12	29.58	30.66	29.94	**32.32**	1.66
Singh [25]	ViT-S+ConvStem	31.16	30.10	31.40	30.92	**33.48**	2.08
Singh	ViT-B+ConvStem	27.12	26.40	27.56	26.68	**29.22**	1.66
Singh	ConvNeXt-T+ConvStem	30.52	29.76	30.64	30.04	**32.78**	2.14
Singh	ConvNeXt-S+ConvStem	29.26	28.46	29.72	28.98	**31.34**	1.62
Singh	ConvNeXt-B+ConvStem	26.90	26.20	27.38	26.82	**28.86**	1.48
Singh	ConvNeXt-L+ConvStem	25.36	24.82	25.94	25.34	**26.94**	1.00
Liu [18]	ConvNeXt-B	26.48	25.88	26.84	26.44	**28.36**	1.52
Liu	ConvNeXt-L	25.08	24.26	25.78	24.90	**26.88**	1.10
Liu	Swin-B	26.86	26.06	27.20	26.74	**28.88**	1.68
Liu	Swin-L	24.16	23.36	24.62	23.80	**26.06**	1.44
1,000 iter		Attack success rate (%)					
source	Architecture	Parsimon	Square	SignH	AccSignH	**Superpixel**	diff
Wong	ResNet-50	56.62	56.62	52.46	50.34	**59.96**	3.34
Engstrom	ResNet-50	48.92	48.16	45.10	44.12	**51.84**	2.92
Salman	ResNet-50	46.96	46.70	44.06	43.08	**50.16**	3.20
Salman	ResNet-18	58.60	58.72	54.92	54.26	**61.98**	3.26
Salman	WideResNet-50-2	42.94	42.22	39.66	38.32	**44.86**	1.92
PyTorch	ResNet-50	72.04	84.64	80.80	55.80	**87.28**	2.64
Debenedetti	XCiT-S12	37.44	36.48	33.74	32.96	**39.66**	2.22
Debenedetti	XCiT-M12	36.04	35.10	32.70	31.86	**37.64**	1.60
Debenedetti	XCiT-L12	35.32	34.64	32.38	31.52	**37.02**	1.70
Singh	ViT-S+ConvStem	35.50	35.30	32.68	32.44	**37.58**	2.08
Singh	ViT-B+ConvStem	31.02	30.38	28.76	28.20	**32.56**	1.54
Singh	ConvNeXt-T+ConvStem	34.88	34.50	32.24	31.58	**37.12**	2.24
Singh	ConvNeXt-S+ConvStem	33.36	32.80	30.94	30.62	**35.28**	1.92
Singh	ConvNeXt-B+ConvStem	30.78	30.14	28.62	28.24	**32.44**	1.66
Singh	ConvNeXt-L+ConvStem	29.24	28.80	27.30	26.42	**30.64**	1.40
Liu	ConvNeXt-B	30.32	29.76	28.22	27.62	**31.94**	1.62
Liu	ConvNeXt-L	28.92	28.34	26.86	26.40	**30.20**	1.28
Liu	Swin-B	30.88	30.44	28.64	28.08	**32.60**	1.72
Liu	Swin-L	28.18	27.44	25.86	25.32	**29.90**	1.72

[a] https://pytorch.org/vision/stable/models.html

Wong (ResNet-50), PyTorch (ResNet-50), and Singh (ViT-S+ConvStem), we plot the trends of attack success rates per iteration for each attack method in Fig. 3.

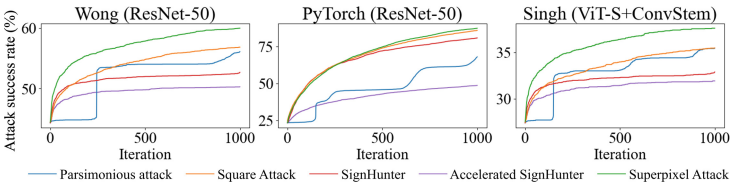

Fig. 3. Transition of attack success rates of each attack method

Figure 3 indicates that Superpixel Attack achieves high success rates in all iterations, including the PyTorch (ResNet-50) model in contrast to SignHunter, which only has high success rates in short iterations. For the other models, each attack method exhibited trends similar to those of Wong (ResNet-50) and Singh (ViT-S+ConvStem). Furthermore, Fig. 4 shows the computational time for superpixels and forward propagation in Superpixel Attack. Although it depends on the computational environment, the computation time of superpixels is less than that of the forward propagation. This indicates that applying superpixels to adversarial attacks is practical in terms of the computation time. For 1000 iterations, the superpixel computation accounts for a very small percentage of the attacks, as indicated by the orange bars.

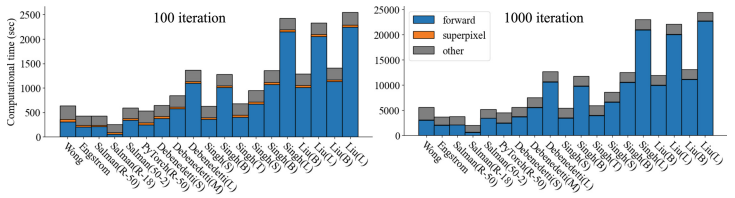

Fig. 4. Computational time of superpixels and forward propagation

6 Conclusion

This study demonstrated that the attack success rates are related to the color variance and compactness of the Update Area. The experimental results suggest that Update Areas with low color variance and high compactness is desirable. Therefore, we propose the Superpixel Attack, which employs superpixels as Update Areas to achieve a good balance between color variance and compactness. The comparison experiments show that the Superpixel Attack improves the attack success rates by an average of 2.10% compared with existing methods for 1000 iterations, which is significant for black-box adversarial attacks. This study indicates that adjusting the Update Areas according to the image can enhance the attack success rates.

Acknowledgements. This research project was supported by the Japan Science and Technology Agency (JST), the Core Research of Evolutionary Science and Technology (CREST), the Center of Innovation Science and Technology based Radical Innovation and Entrepreneurship Program (COI Program), JSPS KAKENHI Grant Number JP16H01707 and JP21H04599, Japan.

References

1. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Süsstrunk, S.: Slic superpixels. Technical report (2010)
2. Al-Dujaili, A., O'Reilly, U.M.: Sign bits are all you need for black-box attacks. In: International Conference on Learning Representations (2020)
3. Aldahdooh, A., Hamidouche, W., Fezza, S.A., Déforges, O.: Adversarial example detection for DNN models: a review and experimental comparison. Artif. Intell. Rev. **55**(6), 4403–4462 (2022)
4. Andriushchenko, M., Croce, F., Flammarion, N., Hein, M.: Square attack: a query-efficient black-box adversarial attack via random search. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12368, pp. 484–501. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58592-1_29
5. Benesova, W., Kottman, M.: Fast superpixel segmentation using morphological processing. In: Conference on Machine Vision and Machine Learning, pp. 67–1 (2014)
6. Carlini, N., Wagner, D.: Towards evaluating the robustness of neural networks. In: 2017 IEEE Symposium on Security and Privacy (SP), pp. 39–57. IEEE (2017)
7. Chen, J., Hou, J., Ni, Y., Chau, L.P.: Accurate light field depth estimation with superpixel regularization over partially occluded regions. IEEE Trans. Image Process. **27**(10), 4889–4900 (2018)
8. Croce, F., et al.: RobustBench: a standardized adversarial robustness benchmark. arXiv preprint arXiv:2010.09670 (2020)
9. Croce, F., Andriushchenko, M., Singh, N.D., Flammarion, N., Hein, M.: Sparse-RS: a versatile framework for query-efficient sparse black-box adversarial attacks. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, pp. 6437–6445 (2022)
10. Croce, F., Hein, M.: Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In: International Conference on Machine Learning, pp. 2206–2216. PMLR (2020)
11. Dai, Z., Liu, S., Tang, K., Li, Q.: Saliency attack: towards imperceptible black-box adversarial attack. arXiv preprint arXiv:2206.01898 (2022)
12. Debenedetti, E., Sehwag, V., Mittal, P.: A light recipe to train robust vision transformers. arXiv preprint arXiv:2209.07399 (2022)
13. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: a large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255. IEEE (2009)
14. Dong, X., et al.: Robust superpixel-guided attentional adversarial attack. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12895–12904 (2020)
15. Engstrom, L., Ilyas, A., Salman, H., Santurkar, S., Tsipras, D.: Robustness (python library) (2019). https://github.com/MadryLab/robustness

16. Kwak, S., Hong, S., Han, B.: Weakly supervised semantic segmentation using superpixel pooling network. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31 (2017)
17. Li, S., Huang, G., Xu, X., Yang, Y., Shen, F.: Accelerated sign hunter: a sign-based black-box attack via branch-prune strategy and stabilized hierarchical search. In: Proceedings of the 2022 International Conference on Multimedia Retrieval, pp. 462–470 (2022)
18. Liu, C., et al.: A comprehensive study on robustness of image classification models: benchmarking and rethinking. arXiv preprint arXiv:2302.14301 (2023)
19. Metzen, J.H., Genewein, T., Fischer, V., Bischoff, B.: On detecting adversarial perturbations. arXiv preprint arXiv:1702.04267 (2017)
20. Moon, S., An, G., Song, H.O.: Parsimonious black-box adversarial attacks via efficient combinatorial optimization. In: International Conference on Machine Learning, pp. 4636–4645. PMLR (2019)
21. Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z.B., Swami, A.: Practical black-box attacks against machine learning. In: Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, pp. 506–519 (2017)
22. Rahmati, A., Moosavi-Dezfooli, S.M., Frossard, P., Dai, H.: GeoDA: a geometric framework for black-box adversarial attacks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8446–8455 (2020)
23. Salman, H., Ilyas, A., Engstrom, L., Kapoor, A., Madry, A.: Do adversarially robust ImageNet models transfer better? In: Advances in Neural Information Processing Systems, vol. 33, pp. 3533–3545 (2020)
24. Schick, A., Fischer, M., Stiefelhagen, R.: Measuring and evaluating the compactness of superpixels. In: Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012), pp. 930–934. IEEE (2012)
25. Singh, N.D., Croce, F., Hein, M.: Revisiting adversarial training for ImageNet: architectures, training and generalization across threat models. arXiv preprint arXiv:2303.01870 (2023)
26. Szegedy, C., et al.: Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199 (2013)
27. Wang, X., et al.: Triangle attack: a query-efficient decision-based adversarial attack. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds.) ECCV 2022. LNCS, vol. 13665, pp. 156–174. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-20065-6_10
28. Wang, Y., Ma, X., Bailey, J., Yi, J., Zhou, B., Gu, Q.: On the convergence and robustness of adversarial training. arXiv preprint arXiv:2112.08304 (2021)
29. Wong, E., Rice, L., Kolter, J.Z.: Fast is better than free: revisiting adversarial training. arXiv preprint arXiv:2001.03994 (2020)
30. Yan, J., Yu, Y., Zhu, X., Lei, Z., Li, S.Z.: Object detection by labeling superpixels. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5107–5116 (2015)
31. Zhang, J., et al.: Towards efficient data free black-box adversarial attack. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 15115–15125 (2022)
32. Zhao, W., Alwidian, S., Mahmoud, Q.H.: Adversarial training methods for deep learning: a systematic review. Algorithms **15**(8), 283 (2022)