Search Space of Adversarial Perturbations against Image Filters

Dang Duy Thang[1], Toshihiro Matsui[2]
Institute of Information Security,
Yokohama, Japan

***Abstract*—The superiority of deep learning performance is threatened by safety issues for itself. Recent findings have shown that deep learning systems are very weak to adversarial examples, an attack form that was altered by the attacker's intent to deceive the deep learning system. There are many proposed defensive methods to protect deep learning systems against adversarial examples. However, there is still lack of principal strategies to deceive those defensive methods. Any time a particular countermeasure is proposed, a new powerful adversarial attack will be invented to deceive that countermeasure. In this study, we focus on investigating the ability to create adversarial patterns in search space against defensive methods that use image filters. Experimental results conducted on the ImageNet dataset with image classification tasks showed the correlation between the search space of adversarial perturbation and filters. These findings open a new direction for building stronger offensive methods towards deep learning systems.**

***Keywords*—Deep neural networks; image filters; adversarial examples; image classification**

I. Introduction

Over the past decade, there has been the rise of deep learning in many tasks such as computer vision [1], automatic driving [2], natural language processing [3], and so on. Deep learning models are designed based on an assumption of inputs and outputs distribution being benign. This leads to when training deep learning models, we only focus to fine-tune the weights, parameters or the number of nodes and hidden layers while setting aside the validity of data. This has created a security issue against deep learning systems. Szegedy et al. [4] explored that deep neural networks are at risk of attacks from adversarial example attacks. Afterward, many research work on technologies that delude AI models has gradually become a hot spot, and researchers have continued to propose new methods of attack and defense. Adversarial attacks have been regularly adapted in both research and commerce. In the computer vision area, many adversarial attacks are proposed in image classification [5], [6], [7], [8], and object detection [9]. There are also many researches work on the adversarial example in text [10], [11], [12], [13]. In the physical world attack, Kurakin et al. [14] first exposed that hazards of adversarial examples. They use an application of Tensor-Flow Camera Demo to capture original images. After that they use Google Inception V3 [15] for classifying those images. The implementation has been shown that a large portion of the image has been misclassified even when observed via the camera lens. Eykholt Kevin et al. [16] invented a new method based on [7] and [17] to create robust adversarial perturbation in the real world. They indicated variation in view angles, distance, and resolution are almost defeated by the robust adversarial examples in physical settings. The proposed algorithm used a term as RP_2 for Robust Physical Perturbations, which was used to craft adversarial examples for road sign recognition systems that perform a high deceiving rate in an efficient setting. And many physical adversarial attacks are proposed in face recognition [18], machine vision [19], and road sign recognition [20]. In the cyberspace security field, there are adversarial attacks in cloud service [21], malware detection [22], [23], and network intrusion detection [24]. Besides the attack methods, many defensive approaches have been proposed and they can be branched into four main categories include adversarial training, denoising, transformation and compression. Szegedy et al. [4] used adversarial examples to train an AI model with the ground truth labels, and it made that model more robust. Goodfellow et al. [5] also used the adversarial training strategy to improve the classification rate on adversarial examples with the MNIST dataset. Tramèr et al. [25] combined the adversarial examples created from many different AI models to increase the robustness of those models. [26], [27] proposed new methods based on the image transformation to reduce the misclassification rate of an AI model. [28], [29] assumed almost adversarial examples are created in the high-frequency domain and they proposed the new method based on image filters to remove the adversarial perturbations. Das et al. [30] introduced a defensive method based on JPEG compression to deceive FGSM [5] and DeepFool [31] attacks. However, newer adversarial attacks such as Carlini&Wagner attacks [7] overcame these compression defensive strategies.

Our Contributions. In this work, we investigate the search space of adversarial perturbation. A challenge in the process of understanding the effects of adversarial noises is very limited so far. How to determine the available space of adversarial noises is very important. Understanding and identifying this space will help us develop better protection systems for deep learning against adversarial examples.

We describe our main contributions of this research as below:

- We have recapped the numerous adversarial defensive and attack methods. Moreover, we have provided a perceptive review of these current methods.
- We discovered the close relationship between search space of adversarial perturbation and image filters.
- Our research opens up a new perspective on creating stronger and more effective attacks on deep learning systems.

Paper outlines. The remainder of our paper is described

as follows. Section II introduces the literature review and the background of adversarial examples. Section III describes our approach on search space of adversarial examples, and Section IV demonstrate our implementation and evaluation results. Section V summaries our work.

II. Literature Review and Background

A. Literature review

In this work, we focus on the relation between feasible space of adversarial perturbation and defensive methods based on frequency domain. So we make a literature review on these defensive methods in this section. Eliminating the adversarial features and retaking the classification rate have been considered in many works. Xu et al. [32] proposed a new defensive approach by using the feature squeezing strategies to remove the adversarial features. There are two key ideas in [32]. The first one considered the bit depth in an input image. By increasing or reducing the bit depth of image, the method removed some adversarial features. The second one used the median filter to defeat the adversarial features. However, [32] required a range of thresholds to separate between adversarial and legitimate features. So the selection of a relevant threshold for a specific dataset or setting is a nontrivial task and it is heuristic. Dang et al. [28] proposed a detection system for automatically identifying adversarial examples with the image filters (Gaussian, Median filter). The system doesn't require to setup any threshold for distinguishing adversarial and benign images. However, there is unclear how the system is able to suffer the stronger and new adversarial attacks. Our paper shows that Gaussian blurring only works well on the small adversarial perturbation, and it is futile to larger and stronger adversarial perturbation.

B. Background

1) Convolutional Neural Networks: Convolutional Neural Networks (CNNs) are designed to learn the important features from the training dataset to match them with the given labels. CNNs are used in many areas [1], [3] and provided open-source [15]. CNNs include multilayers with many operations to process signals from a lower layer to a higher layer in hierarchy architecture. In this research, we emphasize in image classification task so we only cover the brief fundamentals in this area. In an image classification task, CNNs process an input data x and try to figure out the best matching output label y from a set of labels Y. The structure of a CNN can be described as shown in Table I. The layers are described in a top-down order from input to output. We can see for this CNN network, the input is a color image of size 299×299. The first layer is a convolutional layer whose kernel size is 3×3 with a stride of 2. The next convolutional layers also use the same kernel size with a difference with the number of kernels as well as stride. In an inception network, it appears layers called inception layers. These inception layers are different from convolutional layers in that they combine several different kernel sizes at once to extract more important features. The inception layer can also be called inception filters. The last adjacent layer is the logits layer before the softmax function is implemented to calculate the probability for each output label corresponding to the input.

TABLE I. Google Inception Architecture [15]

layer	patch/stride or note	input
conv	3 × 3/2	299 × 299 × 3
conv	3 × 3/1	149 × 149 × 32
conv padded	3 × 3/1	147 × 147 × 32
pool	3 × 3/2	147 × 147 × 64
conv	3 × 3/1	73 × 73 × 64
conv	3 × 3/2	71 × 71 × 80
conv	3 × 3/1	35 × 35 × 192
3 × Inception	Inception filters	35 × 35 × 288
5 × Inception	Inception filters	17 × 17 × 768
2 × Inception	Inception filters	8 × 8 × 1280
pool	8 × 8	8 × 8 × 2048
linear	logits	1 × 1 × 2048
softmax	classifier	1 × 1 × 1000

2) Adversarial Attacks: Adversarial examples are defined as malicious patterns created by the slightly modified aim to fool an AI model but indistinguishable from humans.

FGSM (Fast Gradient Sign Method) was proposed by Ian Goodfellow et al. [5]. In a normal training process, the input and output data distributions are assumed as fixed and unchangeable, so there are only trainable parameters and weights that are fine-tuned respect to a loss objective function between input x and label y. [5] used a very simple idea to reverse that normal process when they fine-tuned input data distribution respect to a new loss objective function between new sample x^{adv} with new specific label y^{adv}:

$$x^{adv} = x - \beta \cdot sign(\nabla_x Loss(x^{adv}, y^{adv})) \quad (1)$$

where β denotes the perturbation size to create an adversarial example x^{adv} from a legitimate input x. From a legitimate input x, FGSM looks for the best adversarial perturbation β to add into x to create a new image x^{adv}. The value of β has to satisfy two requirements include the magnitude of β is as small as possible and respect to the loss objective function between (x,y). For the first requirement, the magnitude of β is smaller, x^{adv} is more similar as x but the convergence rate of the algorithm 1 is slower, while the bigger β makes x^{adv} is more different from the x but the FGSM algorithm converges faster. For the second requirement, the loss objective function between (x,y) is maximized and $Loss(x^{adv}, y^{adv})$ is minimized. Because the total of probabilities of output is equal to one, so the algorithm 1 only needs to consider to minimize $Loss(x^{adv}, y^{adv})$.

In this paper, we use the FGSM [5] method with l-norm optimization as a baseline to conduct assessments of the possible value areas of β during the creation of adversarial examples. Our attack method is based on a white-box attack where victim AI model information is known in advance and can be accessed.

3) Defensive approaches: There are many methods of protection that have been proposed. The typical strategy is adversarial training [4], [33], [25]. The idea of this strategy of protection is that the AI models will be trained with adversarial examples and ground truth labels. With the assumption that the more AI models are learned, the more accurate they will regain and the more likely it will be to misidentify adversarial examples. However, the major drawback of the adversarial training method is that it takes a lot of time to create adversarial examples and training time for AI models. In addition,

this method does not guarantee resistance to new adversarial examples created by other methods than those created by the previous method. Other defensive methods that are often investigated to be pre-processing data. These defensive methods include preprocessing methods based on image transformation [26], [27], filter [28], [29] or compression [30]. Those methods of defense have very impressive results in helping AI systems identify which input is adversarial or legitimate. One of the defenses which also attracts high attention is gradient masking. The adversarial attack methods are largely based on gradient calculations to optimize the loss objective function when creating adversarial examples. For that reason, the idea of hiding the gradient value was proposed. [25] proposed a gradient masking method based on smoothing the gradient gradients that made the global optimal calculation based on gradient slope is more difficult. Author in [34] uses another strategy that is distillation synthesized from different models to create a stronger model against adversarial examples.

III. Search Space of Adversarial Perturbation

A. Search Space on Attacking Phase

One of the important factors in the process of creating adversarial examples is the adversarial perturbation coefficient β. However, how to find out the optimal value of β and its relationship to the currently most powerful defense methods in relation to image filter [28] is unclear. That is the purpose of this study. In this research, we investigate on a white-box attack in creating adversarial examples. This is the setting defined as the attacker can access and use the AI model parameters for conducting an attack pattern. This is possible because currently, the most powerful AI models in image classification tasks are open-source. Many attack methods have been proposed, but most of them rely on FGSM for development, generally, we also use FGSM for creating adversarial examples. One thing to note, it is possible to classify adversarial attacks into two different types based on the purpose of the attacker include non-targeted and targeted attacks. The non-targeted attack is defined as the attacker only focuses on maximize the loss function of (x, y) in order to deceive the AI system. Meanwhile, a targeted attack is defined as the attacker wants to trick the AI system into a misclassifying new pattern in an intentional label rather than merely misidentifying it. Because of this, targeted attacks are more commonly used than non-targeted attacks and we also use it in attacking phases. Our main purpose to decide the size of adversarial perturbation, it means the search space of adversarial perturbation. We consider the norm operation to determine the size of the adversarial noises. Mathematically, the norm operation is used to calculate the distance, or the length of the vectors or the matrixes according to element-wise. The bigger the norm value, the bigger the difference between vectors or matrices and vice versa. Formally, the l_p-norm of vector x is defined as: $\|x\|_p = \sqrt[p]{\sum_i |x_i|^p}$, where $p \in \mathbb{R}$. This is a p^{th}-root of a summation of all elements to the p^{th} power is what we call a norm. The important point is even though every l_p-norm is all looked very similar to each other, their mathematical properties are very different and thus their application is completely different when we use to create the adversarial examples. In this work, we consider three common norm methods: l_1-norm, l_2-norm, and l_∞-norm for evaluating the size of the search space of adversarial perturbation.

$\mathbf{l_1}$-norm. We define x_{true} as the original input vector, l_1-norm of x_{true} is defined as:

$$\|x_{true}\|_1 = \sum_i \left| x_{true}^{(i)} \right| \tag{2}$$

This norm is also well-known as the Manhattan norm and it is one of very common norm operations.

$\mathbf{l_2}$-norm. is the most popular norm and also known as the Euclidean norm. The l_2-norm and other norms are equivalent in the sense that all of them are defined in the same topology. The l_2-norm is defined as:

$$\|x_{true}\|_2 = \sqrt{\sum_i (x_{true}^{(i)})^2} \tag{3}$$

We use the l_2-norm to measure the difference between two vectors x_{true} and x_{adv}, the l_2-norm is re-defined:

$$\|x_{true} - x_{adv}\|_2 = \sqrt{\sum_i (x_{true}^{(i)} - x_{adv}^{(i)})^2} \tag{4}$$

where x_{adv} defines the adversarial example.

$\mathbf{l_\infty}$-norm. The l_∞-norm is defined as equation below:

$$\|x_{true}\|_\infty = \sqrt[\infty]{\sum_i (x_{true}^{(i)})^\infty} \tag{5}$$

Let consider the vector x, if $x^{(i)}$ is each element in vector x, from the property of the infinity itself, we have: $x_i^\infty \approx x_k^\infty \forall i \neq k$, then $\sum_i x_i^\infty = x_k^\infty$. And we have $\|x\|_\infty = \sqrt[\infty]{\sum_i x_i^\infty} = \sqrt[\infty]{x_k^\infty} = |x_k|$. Now we have simple definition of l_∞-norm as: $\|x\|_\infty = max(|x_i|)$.

So our attack phase is denoted as Algorithm 1 by using FGSM. Where x_{true} defines the original input, x_{adv} is adversarial example, y_{true} defines the ground-truth label, y_{adv} is an adversarial label, f is the activation function of machine learning model, β is the maximum adversarial value, l_i defines the norm. For crafting adversarial example, we set a learning rate lr is equal to 0.01, the number of iteration is 500 times.

B. Filter Methods

Most adversarial attack methods look for the optimal values of adversarial perturbation respect to loss objective function to modify the original image. Therefore, the pixels that are incidentally edited are located in the high-frequency domain. Therefore current protection methods based on image filters have proved very effective in eliminating these adversarial noises. However, in order to better understand the search space of this adversarial perturbation and the ability to resist image filters, we studied the two most common image filters, the Gaussian and the Median filter. Mathematically, a Gaussian filter modifies the input image by calculating a convolution the area of a specific image area with a Gaussian function; this transformation is also known as the Weierstrass transform. The area of convolution is often called kernel size and is usually 3x3 or 5x5. When using a Gaussian filter, the kernel window will move across the surface of the input image and compute the kernel window that corresponds to the image area being

Algorithm 1: Crafting Adversarial Examples with l-norm optimization

input : x_{true}, y_{true}, y_{adv}, f, β, l_i
output : x_{adv}
parameter: lr = 0.01, iterations = 500

$x_{adv} \leftarrow x_{true}$ // initial adversarial example
$\delta \leftarrow \vec{0}$ // initial adversarial perturbation
$it \leftarrow 1$ // initial iteration loop
while $\delta < \beta$ *and* $f(x_{adv}) \neq y_{adv}$ *and* $it <= iterations$ **do**
  $x_{adv} \leftarrow x_{true} - \delta \cdot sign(\bigtriangledown Loss(y_{adv}|x_{adv}))$
  $\delta \leftarrow$ norm(l_i)
  maximize $Loss(y_{adv}|x_{adv})$ respect to δ
  $\delta \leftarrow clip(x_{adv}, x - \beta, x + \beta)$
  $it \leftarrow it + 1$
end
return x_{adv}

processed. The second image filter to be considered in this research is the median filter. This is a very common filter used to highlight the edges of an image. The Median filter also uses kernel windows that move across the input image surface. However, the median filter processes that area simply by finding the median value of the image area being processed, then replacing that median value in the pixel position in the center of the windows kernel while preserving the pixel values in neighbors. Our filtering system proceeds by Algorithm 2, where x defines the input image, φ denotes the kernel sizes, f is a machine learning function that computes the predicted label with the highest probability, y_{true} defines the ground truth label, y_{adv} defines the adversarial label, and s is the filter function. Output are the probabilities of the ground truth label (p_{true}) and the adversarial label (p_{adv}).

Algorithm 2: Image Filters on input for image classification task

input : $x_{true}, s, f, y_{true}, y_{adv}$
output : p_{true}, p_{adv}
parameter: $\varphi = [(3 \times 3); (5 \times 5)]$

for i *in* φ **do**
  $x_{filtered} \leftarrow s(x, i)$
  $P \leftarrow f(x_{filtered})$
  $p_{true} \leftarrow P(y_{true})$
  $p_{adv} \leftarrow P(y_{adv})$
end

IV. Implementation and Results

A. Datasets and AI model

The target AI model that we use in the implementation is Google Inception V3 [15] that was trained with 1,000 common categories in the ImageNet [35] dataset. Our attacking phase is a white-box attack setting and a targeted label is "street sign" label. We use FGSM with l_1-norm, l_2-norm, and l_∞-norm to craft adversarial images.

B. Results

Intuitively, because of the copyright issue of the ImageNet dataset, we use our own images (include pictures of vending machine, computer mouse and keyboard) for analysis. We randomly selected targeted labels for the creation of adversarial images. By using the FGSM method in combination with l_1-norm, l_2-norm, and l_∞-norm, from each original image we create three different adversarial images.

Fig. 1 shows the probabilities of the original vending machine label when the input is an vending machine image. Fig. 2 shows the probabilities of the adversarial label with

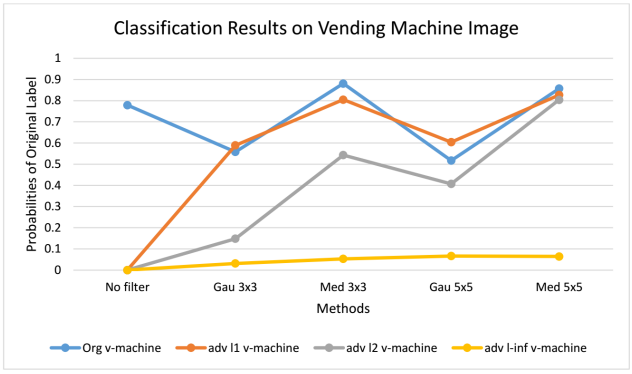


Fig. 1. Classification Results on Vending Machine Image with observation on the probabilities of Original Label

vending machine input. Fig. 3 shows the observations on the

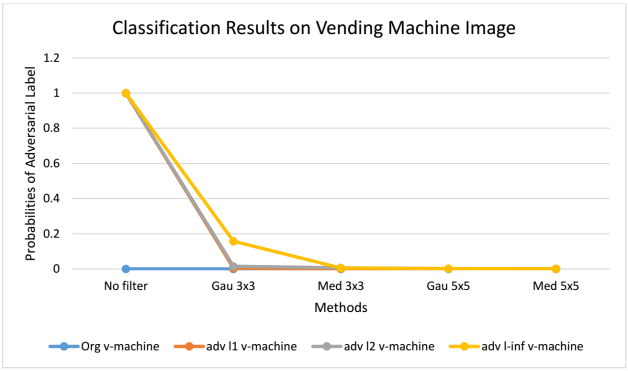
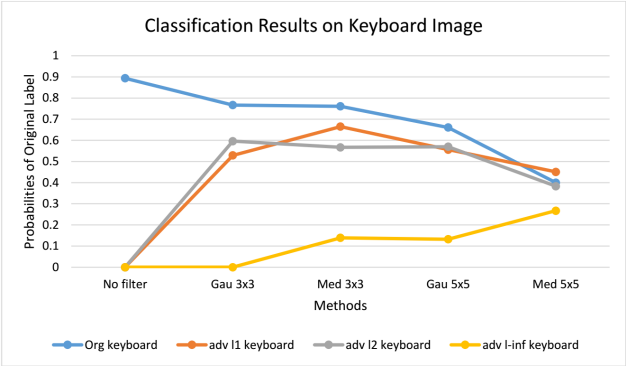


Fig. 2. Classification Results on Vending Machine Image with observation on the probabilities of Adversarial Label

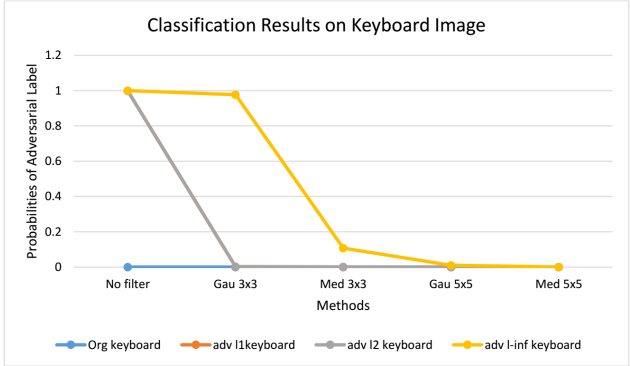
images of computer mouse and keyboard.

Fig. 4 shows the results of creating adversarial images from the original image of the vending machine. We find that the deep learning system is easily fooled with adversarial images. In addition, we intuitively observe that adversarial images created with l_1-norm and l_2-norm are harder to detect than l_∞-norm.

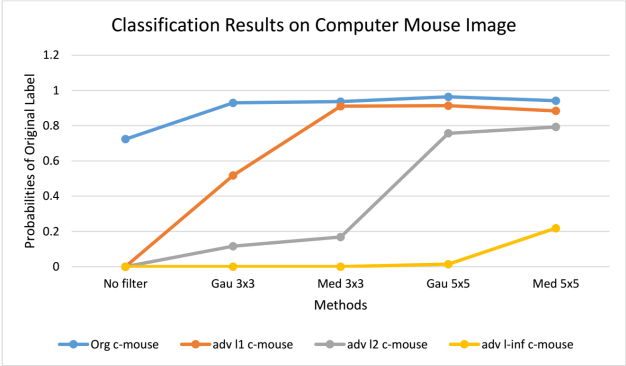
Fig. 5 shows the experimental results when we use the image filters method on the original image of the vending machine. We find that the Gaussian filter reduces classification accuracy more than the median filter. Especially in the case of the median with size filter 3×3 and 5×5, the classification results are better than the original image.



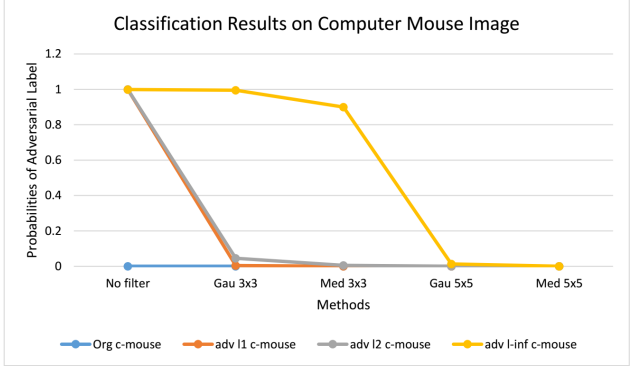
(a) Observation on the probabilities of Original Keyboard Label



(b) Observation on the probabilities of Adversarial Label



(c) Observation on the probabilities of Original Computer Mouse Label



(d) Observation on the probabilities of Adversarial Label

Fig. 3. Classification Results on Keyboard and Computer Mouse Images

Similar to the original image, we also apply image filter methods to adversarial images. Fig. 6 shows classification results on adversarial images created by the FGSM method in combination with l_1-norm. Fig. 7 illustrates classification results on adversarial images created by the FGSM method in combination with l_2-norm. We observed that Gaussian kernel size 3x3 could not restore identity to ground truth label on adversarial image with l_2-norm. The probability for vending machine label is only 14.8%. Meanwhile, the median filter still works effectively in removing adversarial noises. Fig. 8 shows classification results on adversarial images created by the FGSM method in combination with l_∞-norm. We observed that Gaussian kernel size 3×3 could not eliminate the effect of adversarial noise with l_∞-norm on deep learning system classification. Gaussian 5×5 gives better results, but the label with the highest probability of identification is "tabacco shop". The Median filter removes adversarial noises but cannot help the deep learning system correctly identify ground truth labels.

Table II shows experimental results on vending machine (v-machine), computer mouse (c-mouse) and keyboard sets. This result shows us a large correlation between norm operations in search space of adversarial examples. It is clear that for the l_∞-norm, the Gau (3×3, 5×5) and median (3×3) methods are more difficult to completely eliminate adversarial noises based on the l_1 and l_2 norm. Median (5×5) still proved superior in removing adversarial noises in all settings.

V. Conclusion

In this study, we focus on investigating the connection between the search space of adversarial examples and the defense based on the frequency domain. Our empirical results demonstrate that the FGSM method in combination with l_∞-norm produces the strongest adversarial examples. In this case, both the Gaussian and the Median filters are unable to restore identification to the ground truth label. However, when using l_∞-norm to create adversarial examples, we also significantly reduce the quality of the original image compared to using l_1 and l_2 norm. In terms of similarities with the original image, l_1 and l_2 norm produce much better adversarial examples than l_∞ norm.

Acknowledgment

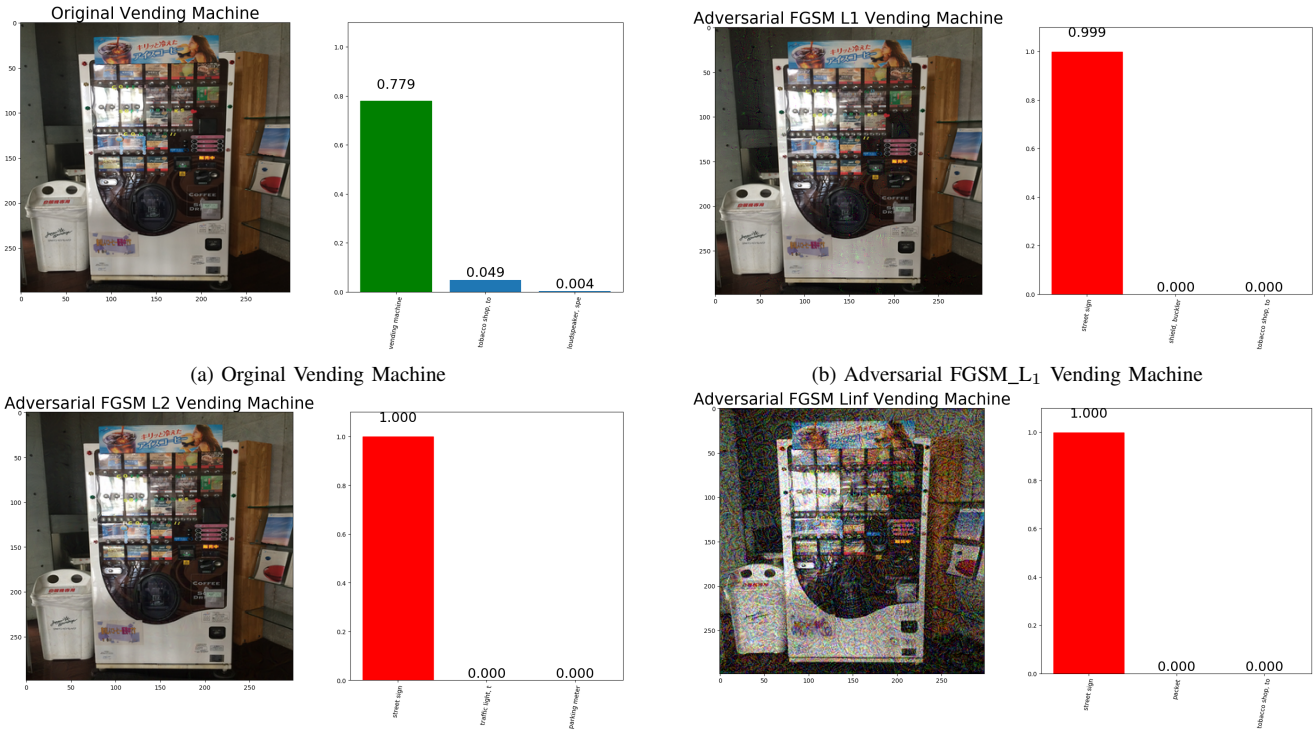
We would like to thank Professor Akira Otsuka for the valuable suggestions on this research. This work was supported by the Iwasaki Tomomi Scholarship.

References

[1] A. Ioannidou, E. Chatzilari, S. Nikolopoulos, and I. Kompatsiaris, "Deep learning advances in computer vision with 3d data: A survey," *ACM Computing Surveys (CSUR)*, vol. 50, no. 2, p. 20, 2017.

[2] D. Watzenig and M. Horn, *Automated driving: safer and more efficient future driving*. Springer, 2016.

[3] J. Ray, O. Johnny, M. Trovati, S. Sotiriadis, and N. Bessis, "The rise of big data science: A survey of techniques, methods and approaches in the field of natural language processing and network theory," *Big Data and Cognitive Computing*, vol. 2, no. 3, p. 22, 2018.



(a) Orginal Vending Machine

(b) Adversarial FGSM_L$_1$ Vending Machine

(c) Adversarial FGSM_L$_2$ Vending Machine

(d) Adversarial FGSM_L$_\infty$ Vending Machine

Fig. 4. Adversarial Vending Machine (targeted class: Street Sign)

TABLE II. IMPLEMENTATION RESULTS

Input	No filter		Gau 3x3		Med 3x3		Gau 5x5		Med 5x5	
	OL	AL	OL	AL	OL	AL	OL	AL	OL	AL
Org v-machine	0.779	0	0.558	0	0.881	0	0.517	0	0.857	0
adv l_1 v-machine	0	0.999	0.589	0.004	0.805	0.001	0.604	0.001	0.827	0
adv l_2 v-machine	0	1	0.148	0.015	0.543	**0.006**	0.407	0.001	0.804	0
adv l_∞ v-machine	0	1	0.031	**0.157**	0.053	0.005	0.066	**0.002**	0.064	**0.001**
Org keyboard	0.894	0	0.767	0	0.761	0	0.661	0	0.4	0
adv l_1 keyboard	0	0.999	0.529	0.002	0.665	0	0.556	0	0.451	0
adv l_2 keyboard	0	0.999	0.596	0.002	0.567	0.001	0.57	0	0.383	0
adv l_∞ keyboard	0	1	0	**0.977**	0.139	**0.107**	0.132	**0.01**	0.267	0
Org c-mouse	0.724	0	0.93	0	0.937	0	0.964	0	0.924	0
adv l_1 c-mouse	0	0.999	0.518	0.004	0.911	0	0.914	0	0.884	0
adv l_2 c-mouse	0	1	0.116	0.045	0.168	0.005	0.757	0	0.793	0
adv l_∞ c-mouse	0	0.999	0	**0.995**	0	**0.9**	0.014	**0.013**	0.218	0

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

[5] I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *International Conference on Learning Representations ICLR*, 2015. [Online]. Available: http://arxiv.org/abs/1412.6572

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

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

[8] 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," in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. ACM, 2017, pp. 15–26.

[9] C. Xiao, R. Deng, B. Li, F. Yu, M. Liu, and D. Song, "Characterizing adversarial examples based on spatial consistency information for semantic segmentation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 217–234.

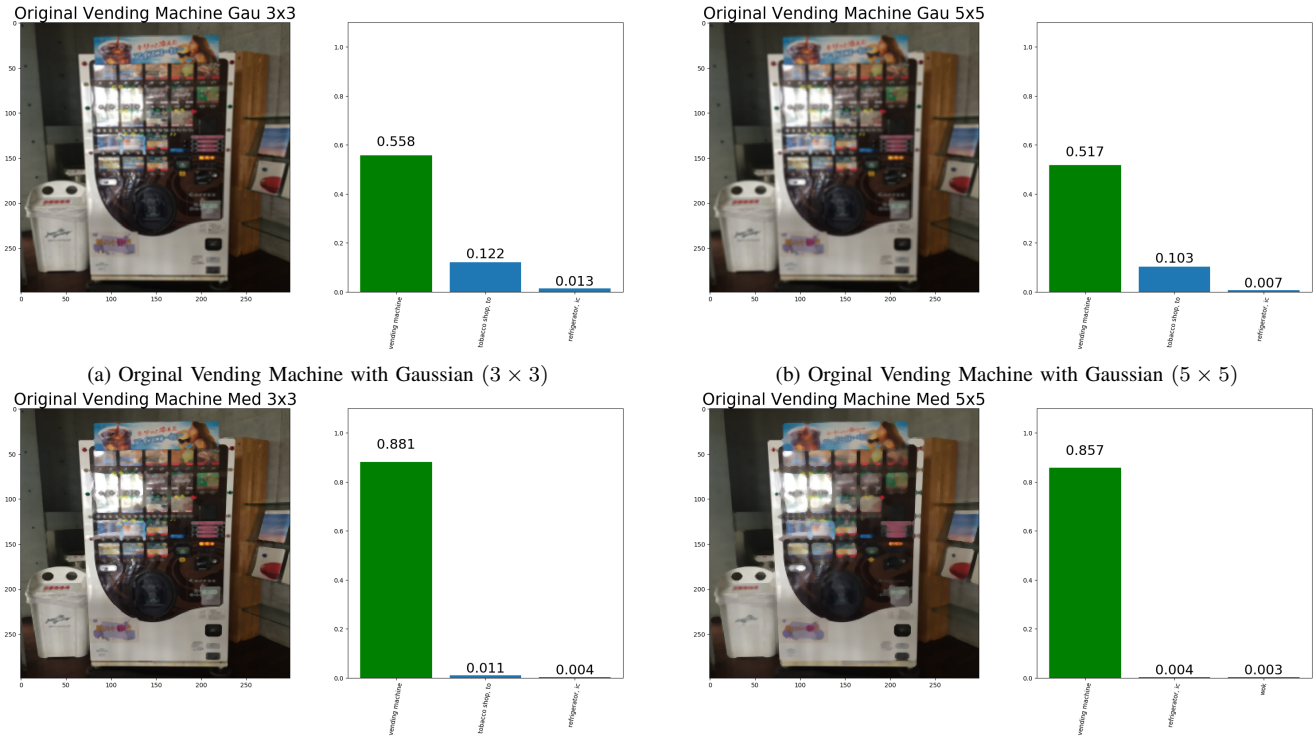
[10] T. Miyato, A. M. Dai, and I. Goodfellow, "Adversarial training methods for semi-supervised text classification," in *International Conference on Learning Representations ICLR*, 2017.

[11] J. Ebrahimi, A. Rao, D. Lowd, and D. Dou, "Hotflip: White-box adversarial examples for text classification," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2018, pp. 31–36.

[12] J. Gao, J. Lanchantin, M. L. Soffa, and Y. Qi, "Black-box generation of adversarial text sequences to evade deep learning classifiers," in *2018 IEEE Security and Privacy Workshops (SPW)*. IEEE, 2018, pp. 50–56.

[13] J. Li, S. Ji, T. Du, B. Li, and T. Wang, "Textbugger: Generating adversarial text against real-world applications," in *26th Annual Network and Distributed System Security Symposium*, 2019.

[14] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the



(a) Orginal Vending Machine with Gaussian (3×3)

(b) Orginal Vending Machine with Gaussian (5×5)

(c) Orginal Vending Machine with Median (3×3)

(d) Orginal Vending Machine with Median (5×5)

Fig. 5. Original Vending Machine with Image Filters

physical world," in *International Conference on Learning Representations ICLR*, 2017.

[15] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[16] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song, "Robust physical-world attacks on deep learning models," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition CVPR*, 2018.

[17] Y. Liu, X. Chen, C. Liu, and D. Song, "Delving into transferable adversarial examples and black-box attacks," in *International Conference on Learning Representations ICLR*, 2017.

[18] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter, "Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2016, pp. 1528–1540.

[19] M. Melis, A. Demontis, B. Biggio, G. Brown, G. Fumera, and F. Roli, "Is deep learning safe for robot vision? adversarial examples against the icub humanoid," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 751–759.

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

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

[22] K. Grosse, N. Papernot, P. Manoharan, M. Backes, and P. McDaniel, "Adversarial perturbations against deep neural networks for malware classification," *arXiv preprint arXiv:1606.04435*, 2016.

[23] H. S. Anderson, J. Woodbridge, and B. Filar, "Deepdga: Adversarially-tuned domain generation and detection," in *Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security*. ACM, 2016, pp. 13–21.

[24] C.-H. Huang, T.-H. Lee, L.-h. Chang, J.-R. Lin, and G. Horng, "Adversarial attacks on sdn-based deep learning ids system," in *International Conference on Mobile and Wireless Technology*. Springer, 2018, pp. 181–191.

[25] A. Kurakin, D. Boneh, F. Tramèr, I. Goodfellow, N. Papernot, and P. McDaniel, "Ensemble adversarial training: Attacks and defenses," in *International Conference on Learning Representations ICLR*, 2018.

[26] T. Dang and T. Matsui, "Image transformation can make neural networks more robust against adversarial examples," *CoRR*, vol. abs/1901.03037, 2019. [Online]. Available: http://arxiv.org/abs/1901.03037

[27] D. D. Thang and T. Matsui, "A label-based approach for automatic identifying adversarial examples with image transformation," in *2019 Seventh International Symposium on Computing and Networking (CANDAR)*. IEEE, Nov 2019, pp. 112–120.

[28] D. D. Thang and T. Matsui, "Automated detection system for adversarial examples with high-frequency noises sieve," in *International Symposium on Cyberspace Safety and Security*. Springer, 2019, pp. 348–362.

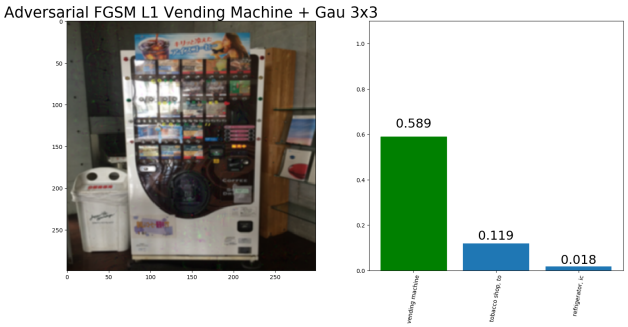
[29] D. D. Thang, T. Kondo, and T. Matsui, "A label-based system for detecting adversarial examples by using low pass filters," *Computer Security Symposium 2019 (CSS2019)*, pp. 1356–1363, 2019.

[30] N. Das, M. Shanbhogue, S.-T. Chen, F. Hohman, L. Chen, M. E. Kounavis, and D. H. Chau, "Keeping the bad guys out: Protecting and vaccinating deep learning with jpeg compression," *arXiv preprint arXiv:1705.02900*, 2017.

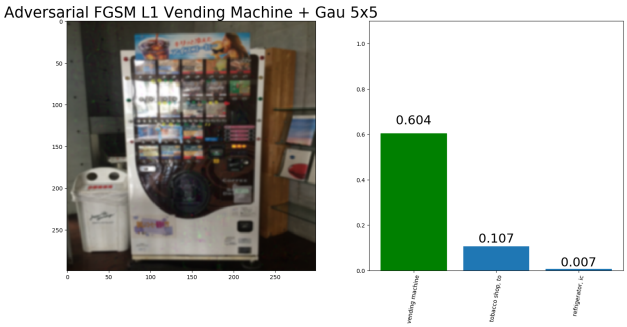
[31] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition CVPR*, 2016, pp. 2574–2582.

[32] W. Xu, D. Evans, and Y. Qi, "Feature squeezing: Detecting adversarial examples in deep neural networks," in *25th Annual Network and Distributed System Security Symposium, NDSS 2018, San Diego, California, USA, February 18-21*, 2018.

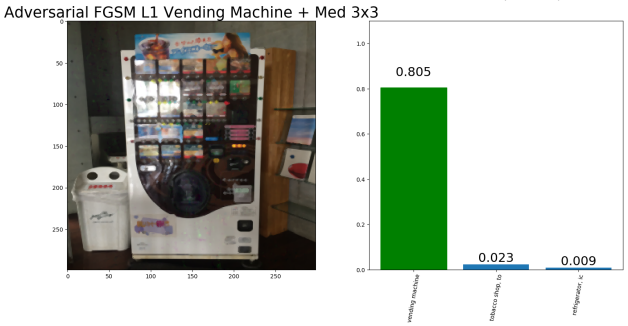
[33] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial machine learn-



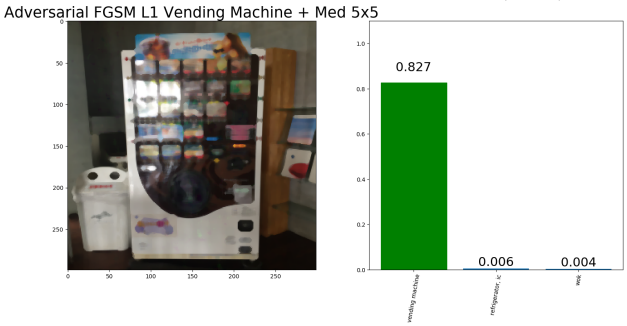
(a) Adversarial Vending Machine with Gaussian (3×3)



(b) Adversarial Vending Machine with Gaussian (5×5)



(c) Adversarial Vending Machine with Median (3×3)



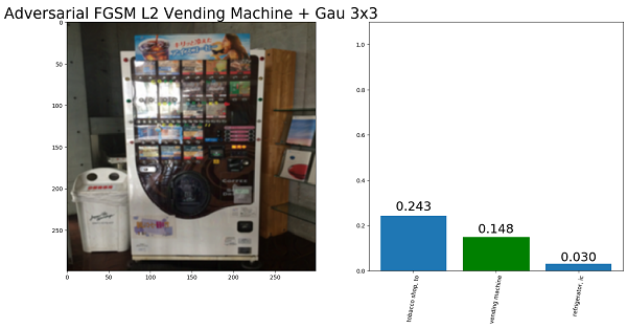
(d) Adversarial Vending Machine with Median (5×5)

Fig. 6. Adversarial FGSM L_1 Vending Machine (targeted class: Street Sign) with Image Filters

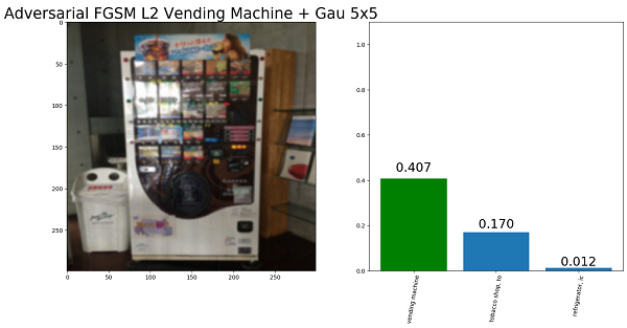
ing at scale," in *International Conference on Learning Representations ICLR*, 2017.

[34] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *2016 IEEE Symposium on Security and Privacy (S&P 2016)*. IEEE, 2016, pp. 582–597.

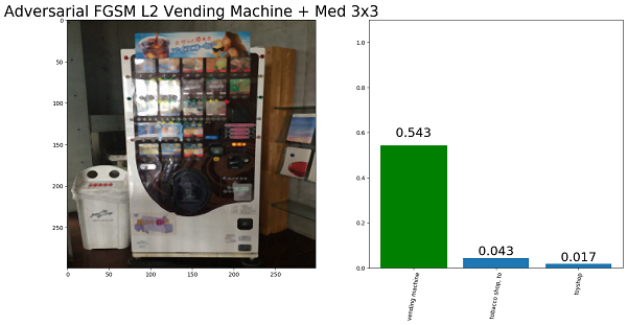
[35] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision IJCV*, vol. 115, no. 3, pp. 211–252, 2015.



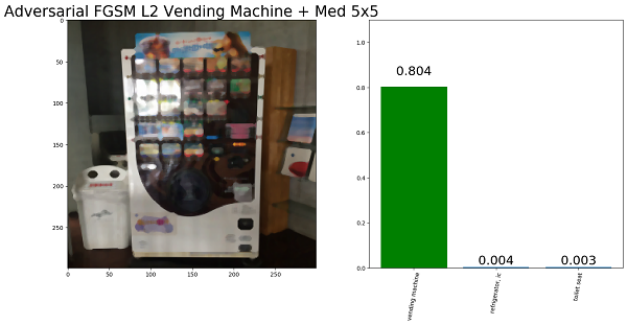
(a) Adversarial Vending Machine with Gaussian (3×3)



(b) Adversarial Vending Machine with Gaussian (5×5)

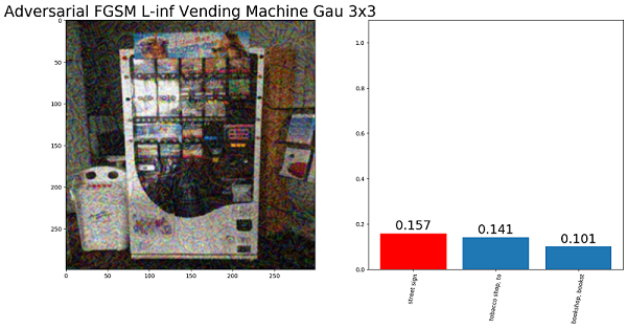


(c) Adversarial Vending Machine with Median (3×3)

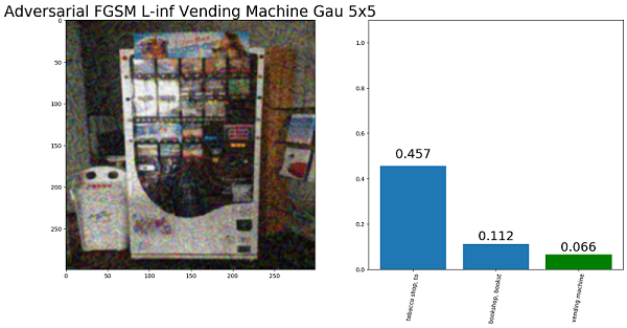


(d) Adversarial Vending Machine with Median (5×5)

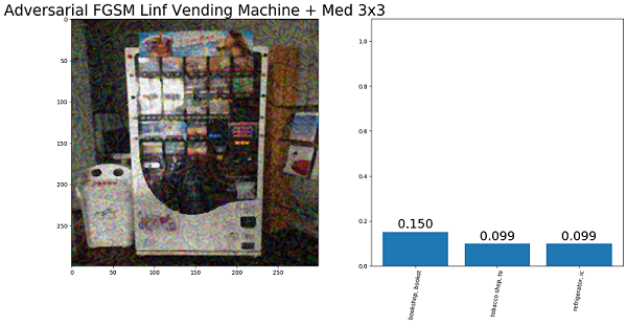
Fig. 7. Adversarial FGSM L_2 Vending Machine (targeted class: Street Sign) with Image Filters



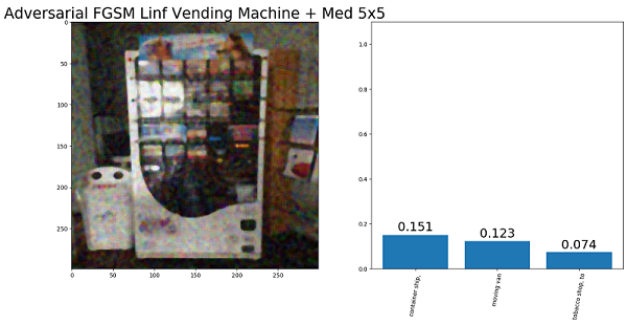
(a) Adversarial Vending Machine with Gaussian (3×3)



(b) Adversarial Vending Machine with Gaussian (5×5)



(c) Adversarial Vending Machine with Median (3×3)



(d) Adversarial Vending Machine with Median (5×5)

Fig. 8. Adversarial FGSM L_∞ Vending Machine (targeted class: Street Sign) with Image Filters