ADVERSARIAL WATERMARKING TO ATTACK DEEP NEURAL NETWORKS

Gengxing Wang[†]*, Xinyuan Chen*[‡]*, Chang Xu*[†]
[†]School of Computer Science, The University of Sydney, Australia.
[‡]MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University, China.
gwan8849@uni.sydney.edu.au, xychen91@sjtu.edu.cn, c.xu@sydney.edu.au

ABSTRACT

Watermark is one of the most fundamental approaches for avoiding potential copyright infringement activities. However, whether its introduction would effect the understanding of deep learning models remains unstudied. In this work, we propose a visible adversarial attack method that transforms and places a provided watermark on the target image to interfere the classification result from an Inception V3 model, which is pretrained on ImageNet. Specifically, the watermark is adjusted iteratively on location, transparency, color, angle and size which are determined by only 9 parameters. We define two types of attack to better simulate the watermark approaches in reality, respectively the watermark is constrained in either transparency or size. Experiments show that the generated adversarial samples are not only capable of fooling the Inception V3 model with high success rates, but also transferable to other models with high confidence, such as the Rekognition developed by Amazon.

Index Terms— Adversarial attack, Watermark, Deep Neural Networks

1. INTRODUCTION

The success of deep learning has drawn great attention in the recent, particularly in computer vision area where Convolutional Neural Network(CNN) is widely utilized. On the other hand, the robustness and transferability of CNN have always been suspected due to its black-box mechanism. Arising since 2014, the concept of adversarial attack has been introduced by Goodfellow [1], in which the target images are modified (referred as adversarial samples after modification) to be misclassified by some state-of-the-art deep learning models such as Inception [2] and Residual Net [3], while the modification does not confuse the perception of human beings.

The attacks can be categorized based on different rules. Depending on the type of perturbation, there exists visible adversarial attacks in which the perturbation is perceptible and restricted by size, location etc. to ensure the target image is still recognizable by humans such as [4]. On the contrary, non-visible adversarial attacks usually generate an imperceptible "mask" with the same size of target image and is applied via addition or subtraction, e.g. [5]. These methods can be further categorized into black-box attack [6] and white-box attack [7] depending on how much knowledge of the model the attacking methods have, and similarly into universal attack [8] and sample-wise attack [1] where in the former a single perturbation is generated and applied to all testing samples, and in the latter each sample is perturbed accordingly.

However, a common problem exists in most approaches is that operations that modify the exact pixel value of adversarial samples, such as image compression and taking screenshots, are likely to eliminate the effect so that attack would fail. Also, perturbations from these attacks are generated following a certain distribution which is artificial and not natural, and consequently are not threatening as long as images are not polluted purposely. Further, most of the perturbations are particularly designed for the target model but whether the success can be transferred onto other models remains unknown.

In reality, watermark is one of the most frequently used approach for identification of the image source and prevention of potential copyright infringement activities such as [9] and [10], yet whether its introduction would affect the content and to which extent remains unanswered. In this work, we propose a visible adversarial attack method using watermarks as perturbations. The input images, which originally being correctly classified, are misclassified by the target model after being watermarked via our method, while the watermarks are constrained in either transparency or size. During the process of each attack, we iteratively adjust only 9 parameters to perform the transform of location, angle, transparency, color and size of the watermark while leaving the input image untouched. Additionally, we experiment the transferability of adversarial samples on other models including Rekognition [11] for black-box attack purposes. Our method present promising results in all experiments, and more importantly has identified the problem that inappropriate watermarks are able to perturb the judgment of state-of-the-art deep learning models.

2. METHODS

2.1. Adversarial Attack with Watermarks

The entire process of the proposed method can be divided into two parts: image recognition and adversarial watermarking, where the former is to feed the watermarked target image I'

This work was supported in part by the Australian Research Council under Project DE180101438 and Project DP180103424.

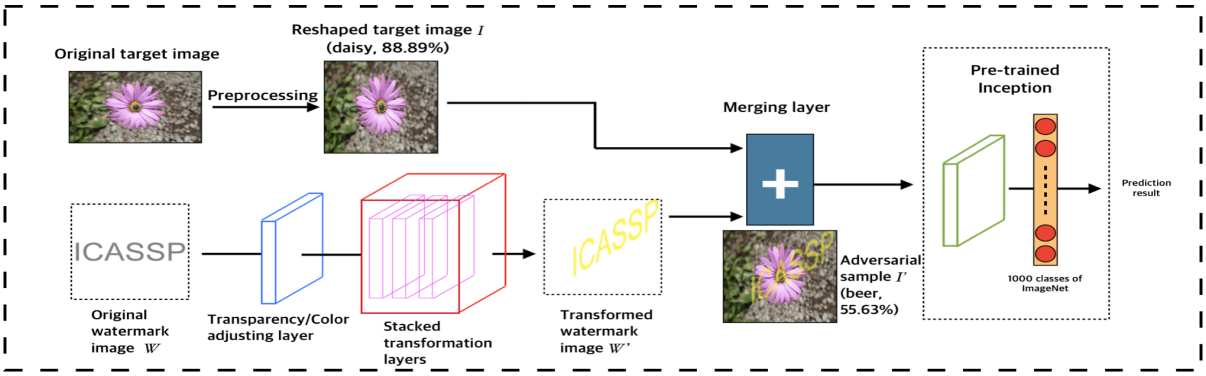

Fig. 1. The overview of our proposed adversarial watermarking method.

into a well-trained deep neural network T for image recognition, and the latter is to update the parameters to determine how the watermark is transformed and placed according to the recognition result. The framework is shown in Fig. 1.

Firstly, the chosen watermark image W, which originally in 4 channels (RGB + alpha channel containing transparency information), is modified by the watermark transformation layer in terms of color, transparency, location, angle and size. Here we use W' to denote the transformed watermark image. In the next step, a pixel-wise summation operation is then performed on the original image I by W', denoted as $W' \oplus I = I'$. Lastly, we optimize the status of W', so that the resulting image I' can fool the well-trained image recognition network T. Given the loss function $\mathcal{L}$ (e.g. the widely-used cross-entropy), we have the optimization problem seeking to maximize:

$$\mathcal{L}_{source} = \mathcal{L}(T(I'), y_{source}), \tag{1}$$

where y_{source} refers to the prediction result from T given the original image I. $T(I')$ denotes the activation values before the last soft-max layer by feeding I' through T.

If we wish the recognition network T to misclassify the modified image I' into label y_{target} given $y_{target} \neq y_{source}$, we minimize the function:

$$\mathcal{L}_{target} = \mathcal{L}(T(I'), y_{target}). \tag{2}$$

Hence, we update the parameters of the watermark transformation layer, while preserving the weights of image recognition network T. For each adjustable parameter p_i for water transformation, we perform an update via a stochastic gradient based algorithm:

$$p_i = p_i + \frac{\partial \mathcal{L}_{source}}{\partial p_i} \times \epsilon_i, \tag{3}$$

or

$$p_i = p_i + (\frac{\partial \mathcal{L}_{source}}{\partial p_i} - \frac{\partial L_{target}}{\partial p_i}) \times \epsilon_i. \tag{4}$$

By experiment, the average misclassification confidence is higher when y_{target} is provided, so Eq.(1) and Eq.(2) are utilized together in experiments. ϵ is the learning rate used to scale the extent of update into a reasonable range, which is set to be $6\mathrm{e}{-3}$ for color/transparency and $1\mathrm{e}{-3}$ for the others. The details of watermark generation will be introduced in the next two sections.

2.2. Transformation Layers

On the purpose of producing appropriate watermark for interfering the judgment of T, we perform three steps of transformation on W as illustrated in Fig. 3. Firstly, we perform rotation with parameter Θ, and each pixel originally at (x, y) is moved to (x', y') where:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} cos(\Theta) & -sin(\Theta) & 0 \\ sin(\Theta) & cos(\Theta) & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}. \tag{5}$$

Similarly, the size is adjusted via two parameters h, w in height and width respectively:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} w & 0 & 0 \\ 0 & h & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \tag{6}$$

and the location is via the update of x^{loc}, y^{loc}:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 & x^{loc} \\ 0 & 1 & y^{loc} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}. \tag{7}$$

We utilize the spatial transformation network (STN) [12], which employs bilinear interpolation to eliminate the problem that the result coordinates might not be integers, for actual implementation to ensure the module is differentiable. As described in Sect.2.1, the parameters are adjusted in each iteration until an adversarial sample is produced or the number of iterations reaches limit.

2.3. Other Layers

In addition to the transformation layers, the color/transparency adjusting layer and merging layer are also indispensable for our method. The former contains 4 parameters corresponds to the 4 channels of W and modifies the value of via channel-wise multiplication. The result is clipped in range $[0, 255]$ to ensure the image is valid except for the alpha channel, which is clipped at $[0, t_{max}]$ where t_{max} is pre-defined and varies depending on the type of attack.

91.8% Wombat 65.3% Llama	99.9% Banana 56.2% Jack-o'-lantern	99.8% Baseball 52.9% Ping-pong ball	95.8% Hamster 50.7% Feather boa	100% Broccoli 50.1% Head cabbage	99.9% Hourglass 52.6% Breaker
89.2% Animal 98.6% Flora	99.2% Banana 98.3% Human	93.5% Baseball Cannot be classified	96.2% Hamster 50.6% Human	99.2% Broccoli 66.5% Cabbage	99.2% Hourglass 80.7% Water jug

Fig. 2. $T-Attack$ adversarial samples. First row presents the prediction result from InceptionV3 (top for original, bottom for perturbed), while the second row presents the result from Rekognition. Even screenshots taken from this paper of above samples are still capable of fooling the Rekognition demo.

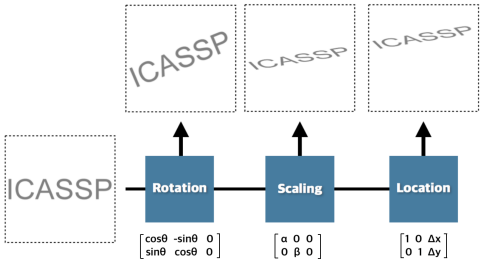

Fig. 3. Three steps of transformations of the watermark (see stacked transformation layers in Fig. 1).

The merging layer produces I' via $W' \oplus I$ with $\oplus$ defined as:

$$A \oplus B = \frac{A_{alpha}}{255} \odot A_{RGB} + \frac{mat(A^w, A^h, 255) - A_{alpha}}{255} \odot B,$$

provided both A is a four channel image (which is W' in our method), B is a standard RGB image, $\odot$ represents element-wise multiplication and $mat(W, H, V)$ denotes a matrix with size $W \times H$ filled with value V.

3. EXPERIMENTS

3.1. Experiment Protocol

In order to validate the feasibility of our method, we utilize images from ImageNet as attacking targets. On 450 images collected from 18 different classes (25 each and are randomly chosen, the size of our dataset is similar with [13] which tested on 400 images and [4] which tested on 100 images), we perform two types of attack on a pretrained Inception V3 model. The raw images are pre-processed using the identical procedure the target model has taken when training for classification, such as resizing and normalization.

Whether each attack is successful obeys the following rules: denote target model as T, each sample image as I with the correct label as L, the target label as L' with $(L' \neq L)$, the perturbed image as I' and $T(I)$ as the prediction result respect to the ImageNet. Firstly, the image will only be used if its origin is correctly classified, i.e. $T(I) = L$. We define three level of success in the evaluation, where difficulty increases as level goes up. The first level (LV1) considers it a success if $T(I) \neq L$ regardless of the confidence; the second level (LV2) considers it a success if $T(I) \neq L$ and $Score(T(I)) \geq 0.5$. Lastly, the third level (LV3) considers it successful only if $T(I) = L'$ and $Score(T(I)) \geq 0.5$, and we use the second most confident label in $T(I)$ as L' by default. The details of attacks will be elaborated in the next two sections.

Additionally, we experiment whether randomly adding watermarks is able to achieve the LV1∼LV3 metrics. For each image, Watermarks are randomly generated following same constraints as in $S/T-Attack$ and the result is presented in Tab. 1. It is noteworthy that statistics of all experiments in Tab. 1 are based on the third (rightmost) $ICASSP$ watermark in Fig. 4, while all other watermarks can be obtained from [14]. We allow 2,000 iterations at most for all attacks while having more iterations would potentially increase the performance. Each adversarial attack takes ∼25 mins at most (if unsuccessful) on a single NVIDIA K80 GPU.

3.2. White-box Attack

In real-world applications, there are two types of watermarking most frequently seen. The first type applies a perceptible semi-transparent watermark covering the foreground object without affecting image content, while the second utilizes a watermark that is opaque yet tiny to prevent covering too much object. To better simulate the real-world watermarking, we simulate the former by constraining $max(W'_{alpha}) \leq 128$, denoted as $T-Attack$ (i.e. transparency-constrained adversarial attack). The latter is simulated via constraining $size(W') \leq 150$ and is denoted as $S-Attack$ (i.e. size-constrained adversarial attack), provided all target images are in resolution 299×299. $T-Attack$ and $S-Attack$ samples are respectively available at Figs. 2 and 4.

It is important to identify that given the watermark is fixed and limited number of adjustable parameters, our method "intelligently" focuses on covering critical features while preserving the overall understanding, when many other works, from our perspective, focus on creating features that might do not make sense to humans yet "realistic" enough to fool

Fig. 4. $S-Attack$ adversarial samples. The prediction results are from InceptionV3 (top for original, bottom for perturbed).

	LV1																		
Class	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	Sum
$S-Attack$	0.48	0.72	0.64	0.64	0.84	0.48	0.60	0.72	0.76	0.56	0.52	0.72	0.56	0.52	0.72	0.84	0.48	0.80	0.64
$T-Attack$	0.92	0.96	0.64	0.76	0.88	0.56	0.72	0.84	0.8	0.88	0.76	0.96	0.88	0.76	0.96	0.92	0.68	0.84	0.82
$S-Attack(random)$	0.16	0.28	0.32	0.52	0.36	0.08	0.04	0.24	0.2	0.24	0.24	0.28	0.44	0.36	0.32	0.08	0.2	0.44	0.27
$T-Attack(random)$	0.2	0.44	0.32	0.4	0.32	0.04	0.12	0.32	0.24	0.2	0.28	0.32	0.4	0.32	0.44	0.08	0.32	0.56	0.30
	LV2																		
Class	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	Sum
$S-Attack$	0.24	0.28	0.20	0.28	0.40	0.24	0.36	0.20	0.40	0.24	0.44	0.28	0.24	0.44	0.28	0.52	0.28	0.24	0.31
$T-Attack$	0.48	0.52	0.20	0.52	0.52	0.16	0.44	0.44	0.24	0.36	0.40	0.48	0.36	0.40	0.48	0.56	0.28	0.44	0.40
$S-Attack(random)$	0.12	0.04	0.12	0.2	0.36	0.04	0.0	0.16	0.12	0.2	0.08	0.16	0.24	0.24	0.16	0.04	0.12	0.24	0.15
$T-Attack(random)$	0.12	0.16	0.04	0.2	0.24	0.0	0.04	0.16	0.12	0.04	0.08	0.08	0.32	0.24	0.32	0.04	0.12	0.28	0.14
	LV3																		
Class	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	Sum
$S-Attack$	0.08	0.12	0.12	0.20	0.32	0.12	0.16	0.16	0.24	0.20	0.24	0.16	0.20	0.24	0.16	0.40	0.16	0.16	0.19
$T-Attack$	0.16	0.36	0.12	0.48	0.44	0.08	0.16	0.20	0.16	0.24	0.24	0.36	0.24	0.24	0.36	0.48	0.24	0.28	0.27
$S-Attack(random)$	0.12	0.0	0.08	0.08	0.24	0.0	0.0	0.08	0.04	0.12	0.0	0.04	0.16	0.12	0.04	0.04	0.08	0.12	0.06
$T-Attack(random)$	0.12	0.04	0.0	0.12	0.2	0.0	0.04	0.04	0.0	0.0	0.0	0.0	0.16	0.08	0.12	0.0	0.04	0.04	0.08

Table 1. This table presents the percentage of samples satisfying LV1, LV2 and LV3 metrics on the corresponding class for $S/T-Attack$ (the higher the better) and random attacks as described in Sect. 3.1 (the lower the better). Randomly chosen classes from 1 to 18 respectively are: banana, barbershop, bee, cardigan, corn, daisy, dishrag, dogsled, hummingbird, iPod, jellyfish, mantis, minivan, poncho, ptarmigan, rapeseed, turnstile, velvet.

	Residual Net	Dense Net
$T-Attack$	0.6	0.68
$S-Attack$	0.56	0.66

Table 2. The recognition accuracy from two other models of $S/T-Attack$ adversarial samples

the classifiers. For $T-Attack$, our method shows promising result on most of the classes, where 82%, 40% and 27% of the adversarial samples are LV1-misclassified, LV2-misclassified and LV3-misclassified. It is noteworthy that the performance is achieved by adjusting only 9 parameters and the range/effect of parameters is also very limited by tuning each. For $S-Attack$, the size restricts its possibility to cover more critical features, resulting in a slightly lower performance than the former: respectively 64%, 31% and 19% adversarial samples satisfy the requirements of LV1, LV2 and LV3 metrics, which are described in Sect. 3.1.

3.3. Black-box Attack

We immigrate adversarial samples onto residual net [3] and dense net [15] as black-box attacks. As shown in Table 2, the recognition accuracy from Res-net and Dense-net are 60% and 68% on $T-Attack$ samples, while 56% and 66% on $S-Attack$ samples respectively. $T-Attack$ adversarial samples slightly outperformed the $S-Attack$ ones on robustness regarding the black-box attack by 2% and 4%, while more importantly it shows around 30% of our samples are still functioning on other models.

Besides evaluating on other deep-learning based models, we have also attempted Rekognition [1] from Amazon. Due to the methodology of preprocessing and labelling might differ from other models, we have manually evaluated some samples for a rough result. The experiment shows that some of the $T-Attack$ adversarial samples are able to perturb the judgment of Rekognition model(as shown in Fig. 2), while most of $S-Attack$ ones failed to interfere. Yet, our experiment reveals the potential risk of utilizing deep-learning models particularly in highly-confidential applications.

4. CONCLUSION

In this paper, we proposed a visible adversarial attack approach utilizing watermarks, with two types of attack to simulate the real-world cases of watermarks and have successfully interfered the judgment from some state-of-the-art deep learning models. Moreover, partial adversarial samples show great transferability onto other models including the Rekognition. In conclusion, we believe this work suggests that the robustness of current object recognition models are yet to be further improved, and more defense approaches shall be employed.

[1] https://aws.amazon.com/rekognition/

5. REFERENCES

[1] Christian Szegedy Ian J. Goodfellow, Jonathon Shlens, "Explaining and harnessing adversarial examples," in *ICLR*, 2015.

[2] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna, "Rethinking the inception architecture for computer vision," in *CVPR*, 2016.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[4] Yoav Goldberg Danny Karmon, Daniel Zoran, "Lavan: Localized and visible adversarial noise," in *ICML*, 2018.

[5] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard, "Deepfool: A simple and accurate method to fool deep neural networks," in *CVPR*, 2016.

[6] Matthias Bethge Wieland Brendel, Jonas Rauber, "Decision-based adversarial attacks: Reliable attacks against black-box machine learning models," in *ICLR*, 2018.

[7] David Wagner Anish Athalye, Nicholas Carlini, "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples," in *ICML*, 2018.

[8] Omar Fawzi Pascal Frossard Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, "Universal adversarial perturbations," in *CVPR*, 2017.

[9] Ja-Ling Wu Yu-Hsun Lin, "Unseen visible watermarking for color plus depth map 3d images," in *ICASSP*, 2012.

[10] Manuel Cedillo-Hernandez Antonio Cedillo-Hernandez Mariko Nakano Hector Perez-Meana Oswaldo Juarez-Sandoval, Eduardo Fragoso-Navarro, "An improved imperceptible visible watermarking algorithm for auxiliary information delivery," in *IET Biometrics*, 2018.

[11] Amazon, "Rekognition," 2018, [Online; accessed 10-October-2018].

[12] Andrew Zisserman Koray Kavukcuoglu Max Jaderberg, Karen Simonyan, "Spatial transformer networks," in *NIPS*, 2015.

[13] Jamie Hayes, "On visible adversarial perturbations digital watermarking," in *CVPR*, 2018.

[14] www.picturetopeople.org, "Transparent text generator," 2018, [Online; accessed 16-October-2018].

[15] Laurens van der Maaten Gao Huang, Zhuang Liu and Kilian Weinberger, "Densely connected convolutional networks," in *CVPR*, 2017.