Hindawi
Security and Communication Networks
Volume 2017, Article ID 1897438, 15 pages
https://doi.org/10.1155/2017/1897438

WILEY | Hindawi

Research Article

Protecting Privacy in Shared Photos via Adversarial Examples Based Stealth

Yujia Liu, Weiming Zhang, and Nenghai Yu

University of Science and Technology of China, Hefei, China

Correspondence should be addressed to Weiming Zhang; zhangwm@ustc.edu.cn

Received 19 July 2017; Revised 1 October 2017; Accepted 10 October 2017; Published 14 November 2017

Academic Editor: Lianyong Qi

Online image sharing in social platforms can lead to undesired privacy disclosure. For example, some enterprises may detect these large volumes of uploaded images to do users' in-depth preference analysis for commercial purposes. And their technology might be today's most powerful learning model, deep neural network (DNN). To just elude these automatic DNN detectors without affecting visual quality of human eyes, we design and implement a novel *Stealth algorithm*, which makes the automatic detector blind to the existence of objects in an image, by crafting a kind of *adversarial examples*. It is just like all objects disappear after wearing an *"invisible cloak"* from the view of the detector. Then we evaluate the effectiveness of *Stealth algorithm* through our newly defined measurement, named *privacy insurance*. The results indicate that our scheme has considerable success rate to guarantee privacy compared with other methods, such as mosaic, blur, and noise. Better still, *Stealth algorithm* has the smallest impact on image visual quality. Meanwhile, we set a user adjustable parameter called *cloak thickness* for regulating the perturbation intensity. Furthermore, we find that the processed images have transferability property; that is, the adversarial images generated for one particular DNN will influence the others as well.

1. Introduction

With the pervasiveness of cameras, especially smartphone cameras, coupled with the almost ubiquitous availability of Internet connectivity, it is extremely easy for people to capture photos and share them on social networks. For example, according to the statistics, around 300 million photos are uploaded onto Facebook every day [1]. Unfortunately, when users are eager to share photos online, they also hand over their privacy inadvertently [2]. Many companies are adept at analyzing the information from photos which users upload to social networks [3]. They collect massive amounts of data and use advanced algorithms to explore users' preferences and then perform more accurate advertising [4]. The owner's life behind each photo is like being peeped.

Recently, we may shudder at a news report about fingerprint information leakage from the popular two-fingered pose in photos [5]. The researchers are able to copy fingerprints according to photos taken by a digital camera as far as three metres away from the subject. Another shocking news is that a new crop of digital marketing firms emerge. They aim at searching, scanning, storing, and repurposing images uploaded to popular photo-sharing sites, to facilitate marketers to send targeted ads [6, 7] or conduct market research [8]. These behaviors of large-scale continuous accessing users' private information will, no doubt, make the photo owners very disturbed.

Moreover, shared photos may contain information about location, events, and relationships, such as family members or friends [9, 10]. This will inadvertently bring security threats to others. After analyzing more than one million online photos collected from 9987 randomly selected users on Twitter, we find that people are fairly fond of sharing photos containing people's portrait on social platforms, as shown in Table 1. We test on 9987 users and take 108.7 images on average from each person. The result shows that about 53.4% of the photos contain people's portrait and 97.9% of the users have shared one or more photos containing people's portrait, which shows great risks of privacy disclosure. In addition to portrait, photos containing other objects may reveal privacy as well, such as road signs and air tickets.

Table 1: Some statistics on photos from Twitter.

Number of randomly collected users	9987
Number of collected photos per user	108.7
Photos containing people's portrait	53.4%
Users sharing photos containing portrait	97.9%

Traditional methods of protecting personal information in images are mosaic, blur, partial occlusion, and so on [11, 12]. These approaches are usually very violent and destructive. A more elegant way is to use a fine-grained access control mechanism, which enforces the visibility of each part of an image, according to the access control list for every accessing user [13]. More flexibly, a portrait privacy preserving photo capturing and sharing system can give users, who are photographed, the selection to choose appearing (select the "tagged" item) in the photo or not (select the "invisible" item) [14].

These processing methods can be good ways to shield people's access. But for many companies which push large-scale advertising, they usually use automated systems rather than manual work to detect user uploaded images. For instance, Figure 1 shows the general process of obtaining privacy through online photos. First, a user shares a photo on the social network unguardedly. Then this photo is collected by astute companies and put into their own automatic detection system. Based on the detection results from a simple photo, the user's privacy information might be at their fingertips. The traditional processing methods (mosaic, blur, etc.) will not only greatly reduce image quality undesirably, but also not work well to the automatic detection system based on DNN, as shown in the later experimental results (Figure 6). Users' purpose of sharing photos is to show their life to other people, but not to give detection machine any opportunity to pry into their privacy. Therefore, we need a technique to deal with images, so that the automatic detection system is unable to work well, but humans cannot be aware of the subtle changes in images.

From Figure 1, we can see, whether for commercial or wicked purposes, the basic model of infringing image privacy follows the same patterns: first, the system gives object proposals, that is, to find where objects may exist in the picture and outline bounding boxes of all possible objects; then the system identifies the specific category of each proposal.

With regard to the detection process, the most advanced algorithm is based on deep neural networks. The unparalleled accuracy turns them into the darling of artificial intelligence (AI). DNNs are able to reach near-human-level performance in language processing [15], speech recognition [16], and some vision tasks [17–19], such as classification, detection, and segmentation.

Although they dominate the AI field, recent studies have shown that DNNs are vulnerable to *adversarial examples* [20], which are well designed to mislead DNNs to give an incorrect classification result. But, for humans, the processed images still remain visually indistinguishable with the original ones. Since adversarial examples have a great deal of resistance on the *classification task*, then for the more complex *detection task*, can we produce adversarial examples with a similar effect? Even if the classification result is incorrect, knowing the existence of an object (not knowing its specific category) is a kind of privacy leakage to some extent. So disenabling the detection machine to see anything is both meaningful and challenging.

As we mentioned above, the detection process is divided into two steps, region proposal and proposal box classification. If we can successfully break through either of these two and visual quality of the original image does not deteriorate, then we are able to produce a new kind of adversarial examples specifically for detection task. A successful resistance involves two cases. One is failing in object proposal, that is, proposing nothing for the next step; and the other is going wrong in recognition on the given right proposal boxes. Our work focuses on the first case. It makes DNNs turn a blind eye to the objects in images; in other words, DNNs will fail to give any boxes of possible objects. Intuitively, our approach is implemented as if objects in an image are wearing an *"invisible cloak."* Therefore, we call it *Stealth algorithm*. Furthermore, we define *cloak thickness* to evaluate the strength of perturbation and *privacy insurance* to measure the capacity of privacy preservation, and their interconnections are also discussed. In addition, we find the *cloak* can be shared; that is, adversarial examples which we make specially for one DNN can also resist other DNN detectors.

In previous work, adversarial examples were usually used to attack various detection systems, such as face recognition [21, 22], malicious code detection [23], and spam filtering [24], all of which are aggressive behaviors out of malice. But, in our work, adversarial examples are made to protect users' privacy. It is an unusually positive and helpful use. Overall, this paper makes the following contributions:

(i) We realize the privacy protection for image content by means of resisting automatic detection machine based on deep neural networks.

(ii) We propose the *Stealth algorithm* of manufacturing adversarial examples for detection task. And this algorithm makes the DNN detection system unable to give object bounding boxes.

(iii) We put forward two new definitions, *cloak thickness* and *privacy insurance*. Measured by them, our experiment shows that *Stealth algorithm* far outdoes several common methods of disturbing image, no matter in effectiveness or in image visual quality.

(iv) We conduct some experiments to show that adversarial examples produced by *Stealth algorithm* have satisfactory transferability property.

The rest of the paper is organized as follows. In Section 2, we review the related work. In Section 3, we introduce several DNN-based detectors and highlight the Faster RCNN detection framework, which we use in our algorithm. In Section 4, we illustrate the approach we design to process an image into an adversarial one for eluding a DNN detector. Then, in Section 5, we evaluate our approach in multiple

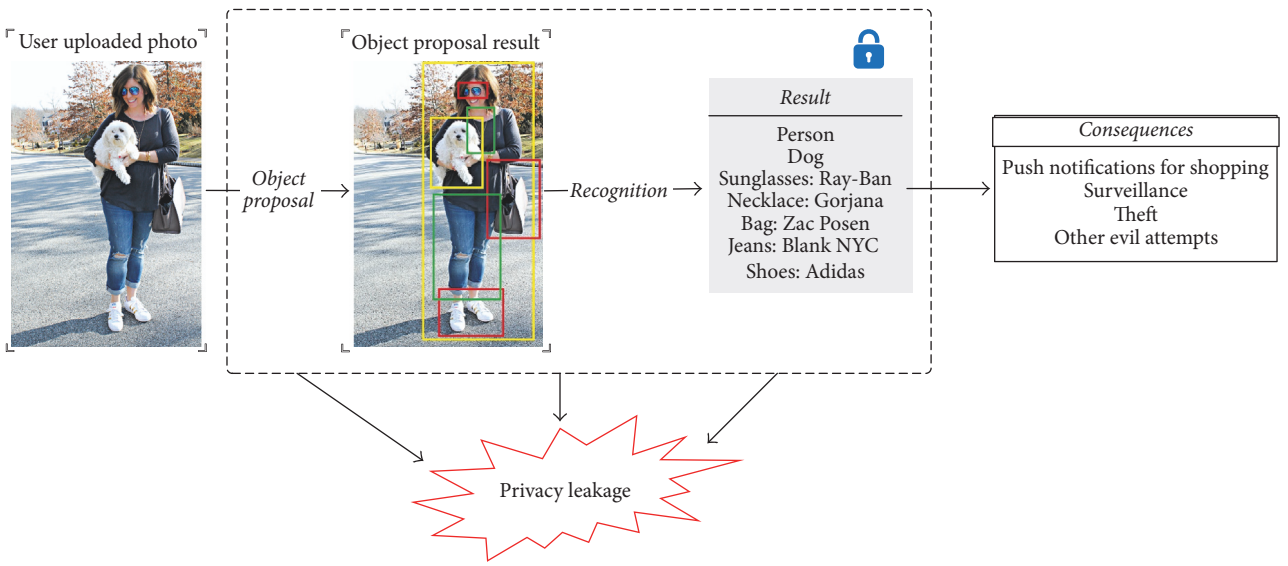


Figure 1: The general process of obtaining privacy through online photos.

aspects. Finally, in Section 6, we make conclusions and discuss the future work.

2. Related Work

Over the past few years, many researchers are committed to studying the limitation of deep learning and it is found to be quite vulnerable to some well-designed inputs. Many algorithms spring up in classification tasks to generate this kind of adversarial input. Christian et al. [25] first discovered that there is a huge difference between DNN and human vision. Adding an almost imperceptible interference into the original image (e.g., a dog seen in human eyes) would cause DNN to misclassify it into a completely unrelated category (maybe an ostrich). Then the fast gradient sign method was presented by Ian Goodfellow et al. [20], which can be very efficient in calculating the interference to an image for a particular DNN model. An iterative algorithm of generating adversarial perturbation by Papernot et al. [26] followed it, which is based on a precise understanding of the mapping between inputs and outputs of DNNs by constructing adversarial saliency maps, and the algorithm can choose any category as the target to mislead the classifier. Nguyen et al. [27], along the opposite line of thinking, synthesized a kind of "fooling images." They are totally unrecognizable to human eyes, but DNNs classify them into a specified category with high confidence. More interestingly, Moosavi-Dezfooli et al. [28] found that there exists a universal perturbation vector that can fool a DNN on all the natural images. Adversarial examples have also been found by Ian Goodfellow et al. [20] to have the transferability property. It means an adversarial image designed to mislead one model is very likely to mislead another as well. That is to say, it might be possible for us to craft adversarial perturbation in circumstance of not having access to the underlying DNN model. Papernot et al. [29, 30] then put forward such a black-box attack based on cross-model transfer phenomenon. Attackers do not need to know the network architecture, parameters, or training data. Kurakin et al. [31] have also shown that, even in the physical world scenarios, DNNs are vulnerable to adversarial examples. Followed by an ingenious face recognition deceiving system by Sharif et al. [32], it enables the subjects to dodge face recognition when they just wear printed paper eye glasses frame.

It can be seen that most of the previous studies on the confrontation against DNNs are usually for classification task. Our work is about the detection task, which is another basic task in computer vision. It is quite distinct from classification, since the returned values of detection are usually both several bounding boxes indicating object positions and labels for categories. Also, its implementation framework is more complicated than classification. Higher dimensions of the result, continuity of the bounding box coordinates, and more complex algorithm make deceiving DNNs on detection become more challenging work.

Viewed from another aspect, Ilia et al. [13] proposed an approach that can prevent unwanted individuals from recognizing users in a photo. When another user attempts to access a photo, the designed system determines which faces the user does not have permission to view and presents the photo with the restricted faces blurred out. Zhang et al. [14] presented a portrait privacy preserving photo capturing and sharing system. People who do not want to be captured in a photo will be automatically erased from the photo by the technique of image inpainting or blurring.

Previous work is to protect the privacy on the level of human vision, whereas these methods have proven less effective for computer vision. In this article, we attempt to design a privacy protection method for computer vision, and meanwhile it ensures human visual quality. This method can

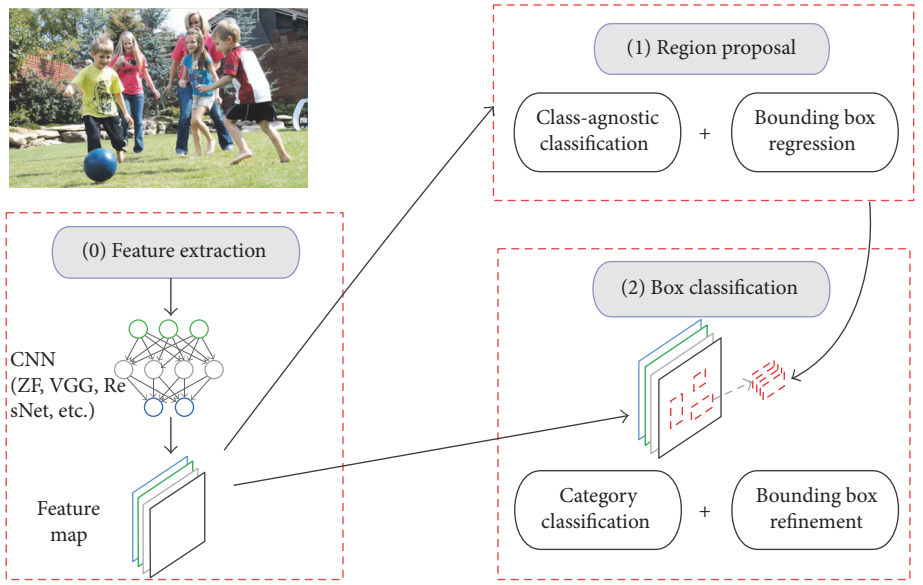


Figure 2: Faster RCNN detection architecture.

be applied in conjunction with the above-mentioned photo-sharing system by Zhang et al. [14] in the future work. And it will allow users to choose whether their purpose of privacy protection is against computer vision or human vision.

3. Object Detectors Based on DNNs

Object detection frameworks based on DNNs have been emerging in recent years, such as RCNN [33], Fast RCNN [34], Faster RCNN [18], Multibox [35], R-FCN [36], SSD [37], and YOLO [38]. These methods generally have excellent performance, many of which have even been put into practical applications. In order to avoid the practitioners hesitating to choose detection frameworks, some researchers have made some detailed test and evaluation on the speed and accuracy of Faster RCNN, R-FCN, and SSD, which are prominent on detection task [39]. Results reflect, in general, that Faster RCNN exhibits optimal performance on the trade-off between speed and accuracy. So we choose to resist the detection system employing the *Faster RCNN* framework, as shown in Figure 2.

Technically, it integrates RPN (region proposal network) and Fast RCNN together. The proposal obtained by RPN is directly connected to the ROI (region of interest) pooling layer [34], which is an end-to-end object detection framework implemented with DNNs. First of all, images are processed to extract features by one kind of DNN (ZF-net, VGG-net, ResNet, etc.). And then the detection happens in the following two stages: *region proposal* and *box classification*. At the stage of region proposal, the features are used for predicting class-agnostic bounding box proposals (object or not object). At the second stage, which is box classification, the same features and corresponding box proposals are used to predict a specific class and bounding box refinement.

Here, we do some explanation of the notations. $\mathbf{X} \in \mathbb{R}^m$ is an input image composed of m pixels, and κ is the number of classes that can be detected. The trained models of the two processes in detection, region proposal, and box classification are f_{rp} and f_{cl}, respectively. And of course there is a feature extraction process f_{feat} before both of them at the very beginning.

In the process of feature extraction, some translation-invariant reference boxes, called anchors, are generated based on the extracted features, denoted by

$$f_{\text{feat}}(\mathbf{X}) = \begin{pmatrix} x_{a1} & y_{a1} & w_{a1} & h_{a1} \\ x_{a2} & y_{a2} & w_{a2} & h_{a2} \\ \vdots & \vdots & \vdots & \vdots \\ x_{ar} & y_{ar} & w_{ar} & h_{ar} \end{pmatrix} = \mathbf{A}(\mathbf{X}). \quad (1)$$

The value r represents the number of anchors. $x_{ai}, y_{ai}, w_{ai}, h_{ai}$ $(i = 1, 2, \ldots, r)$ are, respectively, the vertical and horizontal coordinates of the upper left corner of the anchors and its width and height. Each anchor corresponds to a nearby ground truth box, which can be denoted by

$$b_{\text{gt}}(\mathbf{X}) = \begin{pmatrix} x_{\text{gt1}} & y_{\text{gt1}} & w_{\text{gt1}} & h_{\text{gt1}} \\ x_{\text{gt2}} & y_{\text{gt2}} & w_{\text{gt2}} & h_{\text{gt2}} \\ \vdots & \vdots & \vdots & \vdots \\ x_{\text{gtr}} & y_{\text{gtr}} & w_{\text{gtr}} & h_{\text{gtr}} \end{pmatrix}. \quad (2)$$

Then, in the region proposal stage, f_{rp} predict r region proposals, which are parameterized relative to r anchors.

$$f_{\text{rp}}(\mathbf{X}) = \left(\begin{array}{cccc|c} x_1 & y_1 & w_1 & h_1 & p_1 \\ x_2 & y_2 & w_2 & h_2 & p_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_r & y_r & w_r & h_r & p_r \end{array}\right) = \left(\mathbf{B}(\mathbf{X}) \,\middle|\, \mathbf{P}(\mathbf{X})\right). \quad (3)$$

x_i, y_i, w_i, h_i $(i = 1, 2, \ldots, r)$ are, respectively, the vertical and horizontal coordinates of the upper left corner of the region proposal and its width and height. The value p_i is the probability of it being an object (only two classes: object versus background). For convenience, we let $\mathbf{B}(\mathbf{X})$ be the first four columns, which contain the location and size information of all the bounding boxes and let $\mathbf{P}(\mathbf{X})$ be the last column containing their probability information.

The region proposal function is followed by a function for box classification $f_{\text{cl}}: \mathbb{R}^m \times \mathbb{R}^{r\times 5} \to \mathbb{R}^{n\times(4+\kappa)}$. Here, except the image $\mathbf{X}$, the above partial result $\mathbf{B}(\mathbf{X})$ is also as one of inputs.

$$
\begin{aligned}
&f_{\text{cl}}\left(\mathbf{X}, \mathbf{B}\left(\mathbf{X}\right)\right) \\
&= \left(\begin{array}{cccc|ccca}
\widetilde{x}_1 & \widetilde{y}_1 & \widetilde{w}_1 & \widetilde{h}_1 & p_{11} & p_{12} & \cdots & p_{1\kappa} \\
\widetilde{x}_2 & \widetilde{y}_2 & \widetilde{w}_2 & \widetilde{h}_2 & p_{21} & p_{22} & \cdots & p_{2\kappa} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
\widetilde{x}_n & \widetilde{y}_n & \widetilde{w}_n & \widetilde{h}_n & p_{n1} & p_{n2} & \cdots & p_{n\kappa}
\end{array}\right) \\
&= \left(\ \widetilde{\mathbf{B}}\left(\mathbf{X}, \mathbf{B}\left(\mathbf{X}\right)\right) \,\middle|\, \widetilde{\mathbf{P}}\left(\mathbf{X}, \mathbf{B}\left(\mathbf{X}\right)\right)\ \right).
\end{aligned} \tag{4}
$$

The value n is the number of final bounding boxes results $(n \leq r)$. And similarly, $\widetilde{x}_i$, $\widetilde{y}_i$, $\widetilde{w}_i$, $\widetilde{h}_i$ $(i = 1, 2, \ldots, n)$ represent their location and size information. $p_{i1}, p_{i2}, \ldots, p_{i\kappa}$ are, respectively, the probability of each box result belonging to each class (κ classes in total). We also let $\widetilde{\mathbf{B}}(\mathbf{X}, \mathbf{B}(\mathbf{X}))$ and $\widetilde{\mathbf{P}}(\mathbf{X}, \mathbf{B}(\mathbf{X}))$ be the two parts of the result matrix. In short, Faster RCNN framework is the combination of region proposal and box classification.

4. Stealth Algorithm for Privacy

4.1. Motivation and Loss Function. Our *Stealth algorithm* is aimed at the first stage, region proposal. The processing method which directs at the first stage could be the simplest and most effective, because if the detector does not give any proposal boxes, the next stage (box classification) will be even more impossible to succeed. In a word, we deceive a DNN detector from the source.

Our aim is to find a small perturbation $\delta\mathbf{X}$, $\mathbf{X}^{\mathbf{st}} = \mathbf{X} + \delta\mathbf{X}$, s.t.

$$
\begin{aligned}
&\Pr\left[\mathbf{P}\left(\mathbf{X}^{\mathbf{st}}\right) < \left(\mathbf{th}_{\mathbf{rp}}\right)_r \mid \mathbf{P}\left(\mathbf{X}\right) \geq \left(\mathbf{th}_{\mathbf{rp}}\right)_r, \delta\mathbf{X} < \varepsilon\right] > \eta_{\text{rp}} \\
&\text{where, } \left(\mathbf{th}_{\mathbf{rp}}\right)_r = \text{th}_{\text{rp}} \times \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}_{r\times 1}.
\end{aligned} \tag{5}
$$

Here th_{rp} is a threshold, according to which the detection machine decides each box to be retained or not. Formula (5) expresses that we want to add some small perturbations, so that in region proposal stage any object proposals cannot be detected with considerable probability η_{rp}. In other words, at this stage, all the boxes with low scores (probability of being an object) will be discarded by the system.

Likewise, we can also interfere with the subsequent box classification stage, which can be expressed as

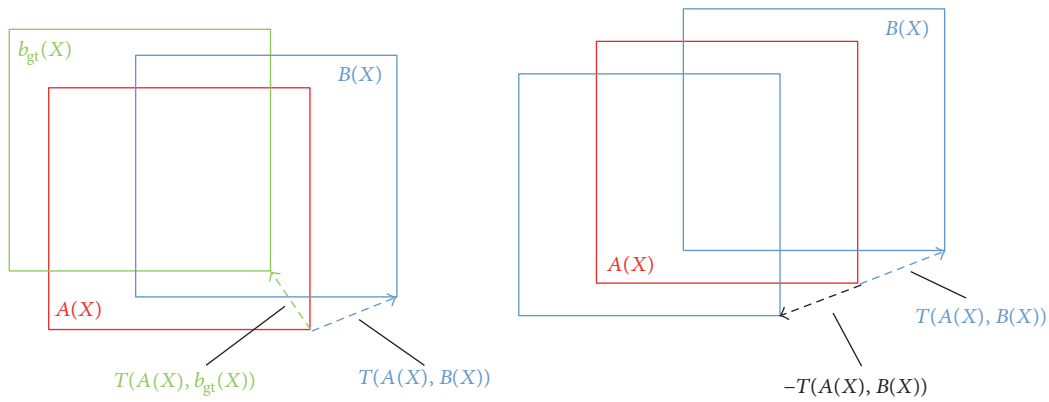
$$
\begin{aligned}
&\Pr\left[\max\left(\widetilde{\mathbf{P}}\left(\mathbf{X}^{\mathbf{st}}, \mathbf{B}\left(\mathbf{X}^{\mathbf{st}}\right)\right)\right) < \left(\mathbf{th}_{\mathbf{cl}}\right)_n \mid \max\left(\widetilde{\mathbf{P}}\left(\mathbf{X}, \mathbf{B}\left(\mathbf{X}\right)\right)\right) \geq \left(\mathbf{th}_{\mathbf{cl}}\right)_n, \delta\mathbf{X} < \varepsilon\right] > \eta_{\text{cl}}, \\
&\text{where, } \left(\mathbf{th}_{\mathbf{cl}}\right)_n = \text{th}_{\text{cl}} \times \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}_{n\times 1}, \quad \max\left(\widetilde{\mathbf{P}}\left(\mathbf{X}, \mathbf{B}\left(\mathbf{X}\right)\right)\right) \triangleq \begin{pmatrix} \max\left\{p_{11}, p_{12}, \ldots, p_{1\kappa}\right\} \\ \max\left\{p_{21}, p_{22}, \ldots, p_{1\kappa}\right\} \\ \vdots \\ \max\left\{p_{n1}, p_{n2}, \ldots, p_{n\kappa}\right\} \end{pmatrix}.
\end{aligned} \tag{6}
$$

Some other bounding boxes will be discarded, because the probability that they belong to any class among the κ classes is less than the threshold th_{cl} with great probability.

On the surface, formula (5) and formula (6) are two modification methods. But in the detection framework Faster RCNN, its two tasks (region proposal and box classification) share the convolution layers; that is, the two functions (f_{rp} and f_{cl}) regard the same deep features as their input. We modify the image for purpose of resisting either of the two stages, which may mislead the other function inadvertently. Therefore, we just choose to deal with the image as formula (5). This operation will obviously defeat the region proposal stage, and it will be even very likely to defeat the following box classification process in formula (6). A more straightforward explanation is that, in the view of the detection machine, our algorithm makes the objects in the image no longer resemble *an object*, let alone *an object of a certain class*. The image seems to be wearing an invisible cloak. So, in the machine's eyes, an image including a lot of content looks completely empty, which lives up to our expectation.

We are more concerned about the region proposal stage, and its loss function in Faster RCNN framework is

$$
\begin{aligned}
&\mathscr{L}\left(\mathbf{T}\left(\mathbf{A}\left(\mathbf{X}_\mathbf{i}\right), \mathbf{B}\left(\mathbf{X}_\mathbf{i}\right)\right), \mathbf{T}\left(\mathbf{A}\left(\mathbf{X}_\mathbf{i}\right), b_{\text{gt}}\left(\mathbf{X}_\mathbf{i}\right)\right), \mathbf{P}\left(\mathbf{X}_\mathbf{i}\right),\right. \\
&\left.\phi\left(\mathbf{X}_\mathbf{i}\right); \theta\right) = \lambda \cdot \mathbf{P}\left(\mathbf{X}_\mathbf{i}\right) \ell_{\text{box}}\left(\mathbf{T}\left(\mathbf{A}\left(\mathbf{X}_\mathbf{i}\right), \mathbf{B}\left(\mathbf{X}_\mathbf{i}\right)\right),\right. \\
&\left.\mathbf{T}\left(\mathbf{A}\left(\mathbf{X}_\mathbf{i}\right), b_{\text{gt}}\left(\mathbf{X}_\mathbf{i}\right)\right)\right) + \mu \cdot \ell_{\text{prb}}\left(\mathbf{P}\left(\mathbf{X}_\mathbf{i}\right), \phi\left(\mathbf{X}_\mathbf{i}\right)\right).
\end{aligned} \tag{7}
$$



(a) In training phase (b) In our algorithm

Figure 3: Region proposal process in the training phase and in our algorithm.

Here $\mathbf{T}(\mathbf{A}(\mathbf{X_i}), \mathbf{B}(\mathbf{X_i}))$ represents a certain distance between anchors and the predicted region proposals, and $\mathbf{T}(\mathbf{A}(\mathbf{X_i}), b_{gt}(\mathbf{X_i}))$ is that between anchors and ground truth boxes (in Figure 3, we represent it as a vector). In training phase, the goal of the neural network is to make $\mathbf{T}(\mathbf{A}(\mathbf{X_i}), \mathbf{B}(\mathbf{X_i}))$ closer to $\mathbf{T}(\mathbf{A}(\mathbf{X_i}), b_{gt}(\mathbf{X_i}))$, as shown in Figure 3(a). More specifically,

$$\begin{aligned}
&\mathbf{T}\left(\mathbf{A}\left(\mathbf{X}\right), \mathbf{B}\left(\mathbf{X}\right)\right) \\
&= \begin{pmatrix}
\frac{(x_1 - x_{a1})}{w_{a1}} & \frac{(y_1 - y_{a1})}{h_{a1}} & \log\left(\frac{w_1}{w_{a1}}\right) & \log\left(\frac{h_1}{h_{a1}}\right) \\
\frac{(x_2 - x_{a2})}{w_{a2}} & \frac{(y_2 - y_{a2})}{h_{a2}} & \log\left(\frac{w_2}{w_{a2}}\right) & \log\left(\frac{h_2}{h_{a2}}\right) \\
\vdots & \vdots & \vdots & \vdots \\
\frac{(x_r - x_{ar})}{w_{ar}} & \frac{(y_r - y_{ar})}{h_{ar}} & \log\left(\frac{w_r}{w_{ar}}\right) & \log\left(\frac{h_r}{h_{ar}}\right)
\end{pmatrix} \\
&\triangleq \left(\frac{(\mathbf{x} - \mathbf{x}_a)}{\mathbf{w}_a} \quad \frac{(\mathbf{y} - \mathbf{y}_a)}{\mathbf{h}_a} \quad \log\left(\frac{\mathbf{w}}{\mathbf{w}_a}\right) \quad \log\left(\frac{\mathbf{h}}{\mathbf{h}_a}\right)\right).
\end{aligned} \tag{8}$$

Similarly,

$$\begin{aligned}
&\mathbf{T}\left(\mathbf{A}\left(\mathbf{X}\right), b_{gt}\left(\mathbf{X}\right)\right) \\
&\triangleq \left(\frac{(\mathbf{x}_{gt} - \mathbf{x}_a)}{\mathbf{w}_a} \quad \frac{(\mathbf{y}_{gt} - \mathbf{y}_a)}{\mathbf{h}_a} \quad \log\left(\frac{\mathbf{w}_{gt}}{\mathbf{w}_a}\right) \quad \log\left(\frac{\mathbf{h}_{gt}}{\mathbf{h}_a}\right)\right).
\end{aligned} \tag{9}$$

And $\phi(\mathbf{X_i})$ in the loss function is the probability of the ground truth object labels ($\phi(\mathbf{X_i}) \in \{0, 1\}$: 1 represents the box is an object and 0 represents not). θ is the parameter of the trained model. At the region proposal stage, the total loss $\mathscr{L}$ is composed of two parts, box regression loss ℓ_{box} (smooth $L1$ loss) and binary classification loss ℓ_{prb} (log loss). λ and μ are the weights balancing the two losses.

4.2. Algorithm Details. Here we elaborate on our *Stealth algorithm* of generating adversarial examples in our experiment. Algorithm 1 shows our *Stealth* idea. It takes a benign image $\mathbf{X}$, a trained feature extraction and detection model f_{feat} and f_{rp}, iteration number Γ, and a user-defined cloak thickness τ as input. Users can control how much privacy to protect as needed, by adjusting the parameter τ to change the interference intensity added to an image. It outputs a new adversarial example $\mathbf{X^{st}}$ against detection. In general, the algorithm employs two basic steps over multiple iterations: (1) Get the anchors $\mathbf{A}(\mathbf{X_i})$ on the basis of the features extracted from DNN. $\mathbf{X_i}$ is the temporary image in the ith iteration. (2) Compute the forward prediction $f_{rp}(\mathbf{X_i})$. This indicates the position of the prediction boxes. (3) Get the adversarial perturbation $\delta\mathbf{X_i}$ based on backpropagation of the loss. The loss function $\mathscr{L}$ is the same as that of Faster RCNN, but we change one of its independent variables. In other words, we replace $\mathbf{T}(\mathbf{A}(\mathbf{X_i}), b_{gt}(\mathbf{X_i}))$ with $-\mathbf{T}(\mathbf{A}(\mathbf{X_i}), \mathbf{B}(\mathbf{X_i}))$, as shown in Figure 3(b). We compute the backpropagation value of the total loss function:

$$\begin{aligned}
&\nabla_{\mathbf{X_i}} \mathscr{L}\left(\mathbf{T}\left(\mathbf{A}\left(\mathbf{X_i}\right), \mathbf{B}\left(\mathbf{X_i}\right)\right),\right. \\
&\left. -\mathbf{T}\left(\mathbf{A}\left(\mathbf{X_i}\right), \mathbf{B}\left(\mathbf{X_i}\right)\right), \mathbf{P}\left(\mathbf{X_i}\right), \phi\left(\mathbf{X_i}\right); \theta\right)
\end{aligned} \tag{10}$$

as the perturbation $\delta\mathbf{X_i}$ in one iteration. The role of backpropagation and loss function in the training process is to adjust the network so that the current output moves closer to the ground truth. Here we substitute the reverse of the direction towards which the box should be adjusted ($-\mathbf{T}(\mathbf{A}(\mathbf{X_i}), \mathbf{B}(\mathbf{X_i}))$) for the ground truth b_{gt}. An intuitive understanding is that we try to track the adjustment on region proposal by DNN detector. If it is found that the DNN wants to move the proposals in a certain direction, then we add some small and well-designed perturbations onto the original image. These perturbations may cause the proposals to move in the opposite direction and consequently counteract their generation.

The original image and that processed by the *Stealth algorithm* will have totally different results through the DNN detector, as shown in Figure 4. The original image can be detected and labeled correctly, while as for the processed image no objects are detected by the DNN detector; that is, no information has been perceived at all. Even better, in human eyes, there is little difference between the adversarial image and the original image.

Input: Image $\mathbf{X}$, model f_{feat}, f_{rp}, iteration number Γ, invisible cloak thickness τ.
Output: Adversarial image $\mathbf{X}^{\text{st}}$.
Initialize: $\mathbf{X}_0 \Leftarrow \mathbf{X}$, $i \Leftarrow 0$.
while $i < n$ **do**
$\mathbf{A}(\mathbf{X}_i) \Longleftarrow f_{\text{feat}}(\mathbf{X}_i)$,
$(\mathbf{B}(\mathbf{X}_i), \mathbf{P}(\mathbf{X}_i)) \Longleftarrow f_{\text{rp}}(\mathbf{X}_i)$,
$\delta\mathbf{X}_i \Longleftarrow -\frac{\tau}{n} \cdot (\nabla_{\mathbf{X}_i} \mathscr{L}(\mathbf{T}(\mathbf{A}(\mathbf{X}_i), \mathbf{B}(\mathbf{X}_i)), -\mathbf{T}(\mathbf{A}(\mathbf{X}_i), \mathbf{B}(\mathbf{X}_i)), \mathbf{P}(\mathbf{X}_i), \phi(\mathbf{X}_i); \theta))$,
$\mathbf{X}_{i+1} \Longleftarrow \mathbf{X}_i + \delta\mathbf{X}_i$,
$i \Longleftarrow i + 1$,
end while
$\mathbf{X}^{\text{st}} \Longleftarrow \mathbf{X}_i$,
return $\mathbf{X}^{\text{st}}$.

Algorithm 1: Stealth algorithm for detection system.

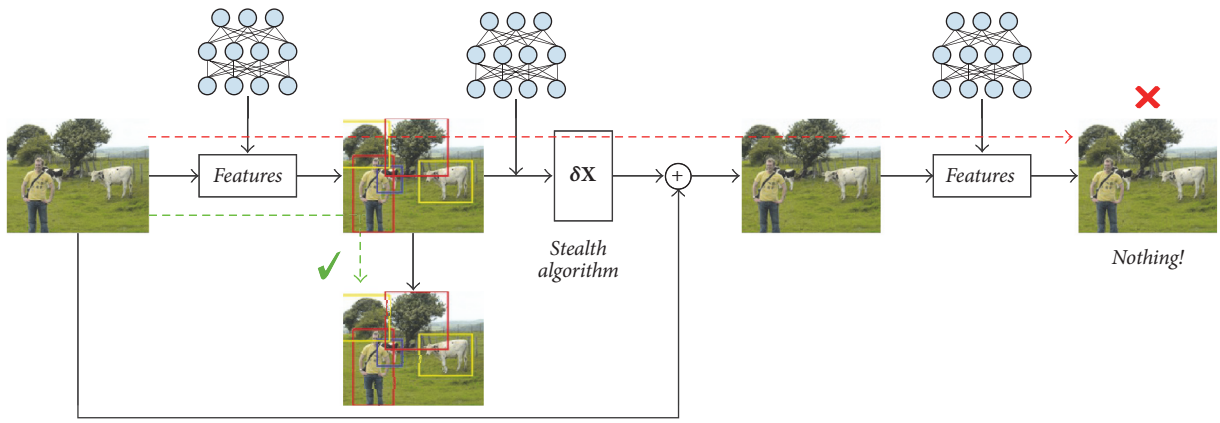


Figure 4: The original and processed image through a DNN detector.

4.3. Privacy Metric. To measure the effectiveness of our algorithm quantitatively, we define a variable PI, named *privacy insurance*. It can be interpreted as how much privacy the algorithm can protect. We let O_k be the total number of bounding boxes of the kth class ($1 \le k \le \kappa$), which are detection results based on all original images, including both correct and wrong results. And we let V_k be the number of just correct boxes of each class detected on adversarial ones and PI be the average of all PI_k values.

$$\text{PI}_k = \begin{cases} 1 - \dfrac{V_k}{O_k} & O_k \neq 0 \\ 0 & O_k = 0, \end{cases} \quad 1 \le k \le \kappa$$

$$\text{PI} = \frac{\sum_{k=1}^{\kappa} \text{PI}_k}{\sum_{k=1}^{\kappa} \delta(O_k, 0)}, \tag{11}$$

$$\text{where, } \delta(O_k, 0) = \begin{cases} 1 & O_k \neq 0 \\ 0 & O_k = 0, \end{cases} \quad 1 \le k \le \kappa.$$

We can observe from the above definition that PI means the success rate of our detection resistance actually, and it also indicates how much privacy owned by users can be preserved.

Normally, mAP (mean average precision) is usually used to measure the validity of a detector. But here our PI value is a more appropriate evaluation index. Suppose there are κ classes in the dataset, each with an independent *privacy insurance* value PI_k ($k = 1, 2, \ldots, \kappa$), because the model itself has some errors when detecting original images; that is, the accuracy is not 100%. And the major concern of our algorithm is to resist the detection model. Consider such a case: the machine's judgment itself on the original image is wrong. And after dealing with it by the algorithm, the judgment is still wrong, but it has two different wrong forms. Then this processing of resisting detection is successful theoretically. But calculating the difference of mAP value between pre- and postprocessing cannot reflect that this case is a successful one. On the contrary, PI can evaluate the validity of our work at all cases, of course including the above one.

5. Experiment and Evaluation

In order to illustrate the effectiveness of our *Stealth algorithm*, we will evaluate it from four aspects: (i) We clarify whether the processed images by our algorithm can resist DNNs effectively. We show the result of performing on nearly 5000 images in PASCAL VOC 2007 test dataset to confirm that. (ii) We compare our algorithm with other ten methods of modifying images for resisting detection. Results indicate that our method works best and has minimal impact on

(a) (b) (c) (d) (e)

Figure 5: (a) Original images; (b) original results; (c) adversarial perturbations (×20 to show more clearly); (d) processed images; (e) new results.

image visual quality. (iii) We explore the relations among cloak thickness, visual quality, and *privacy insurance* in the algorithm. (iv) We illustrate the transferability of our *Stealth algorithm* on different DNNs.

5.1. Some Experimental Setups. We test our algorithm on the PASCAL VOC 2007 dataset [40]. This dataset consists of 9963 images and is equally split into the trainval (training and validation) set and test set. And it contains 20 categories, which are common objects in life, including people, several kinds of animals, vehicles, and indoor items. Each image contains one or more objects, and the objects vary considerably in scale. As for DNNs, we use two nets trained by Faster RCNN on the deep learning framework Caffe [41]. One is the fast version of ZF-net [42] with 5 convolution layers and 3 fully connected layers, and the other is the widely used VGG-16 net [43] with 13 convolution layers and 3 fully connected layers. In addition, our implementation is completed on a machine with 64 GB RAM, Intel Core i7-5960X CPU, and two Nvidia GeForce GTX 1080 GPU cards.

5.2. Effectiveness and Comparison. Here we first illustrate the effectiveness through several samples and compare with other trivial methods. In the next subsection, we will then introduce the results of larger-scale experiments. As shown in Figure 5, one can observe that images processed by our algorithm can dodge detection successfully. And humans can hardly notice the slight changes. Consequently, we have generated a kind of machine-harm but human-friendly images. For most images in our experimental dataset, the machine cannot see where objects are (the first two rows in Figure 5), let alone identifying what specific category they belong to. For a small number of images, even if the machine is really aware that there may be some objects in the image, it cannot locate them exactly or classify them correctly (the last row in Figure 5). In short, in the vast majority of cases, the machine will give the wrong answer. To give a quantitative analysis, we introduce a new measurement, *cloak thickness*, which will be explained in detail in Section 5.3.

In addition, we show the other ten trivial but interesting ways of modifying images to interfere with detection machines in Figure 6. We use PSNR (Peak Signal to Noise Ratio) to evaluate the visual quality of the processed images. These methods include both global and local modification. Local processing here is on the location of objects, rather than a random location.

(i) Whether global mosaic in Figure 6(b), local mosaic in Figure 6(c), global blur (Gaussian blur here) in Figure 6(d), or local blur in Figure 6(e), compared to other ways, their PSNR value is a bit larger. This indicates that although the perturbation is not very considerable, the image gets disgustingly murky. People usually cannot endure viewing such images on the Web. Sadly, although people cannot bear it, the machine can still detect most objects correctly. Thus some smoothing filters (like mosaic or Gaussian blur) are unable to resist DNN-based detector. We think DNNs could compensate for the homogeneous loss of information; that is, once a certain pixel is determined, a small number of surrounding pixels are not very critical.

(ii) As shown in Figures 6(f) and 6(g), an image with large Gaussian noise has poor quality judged by its low PSNR value. But the machine is also able to draw an almost correct conclusion. This shows that

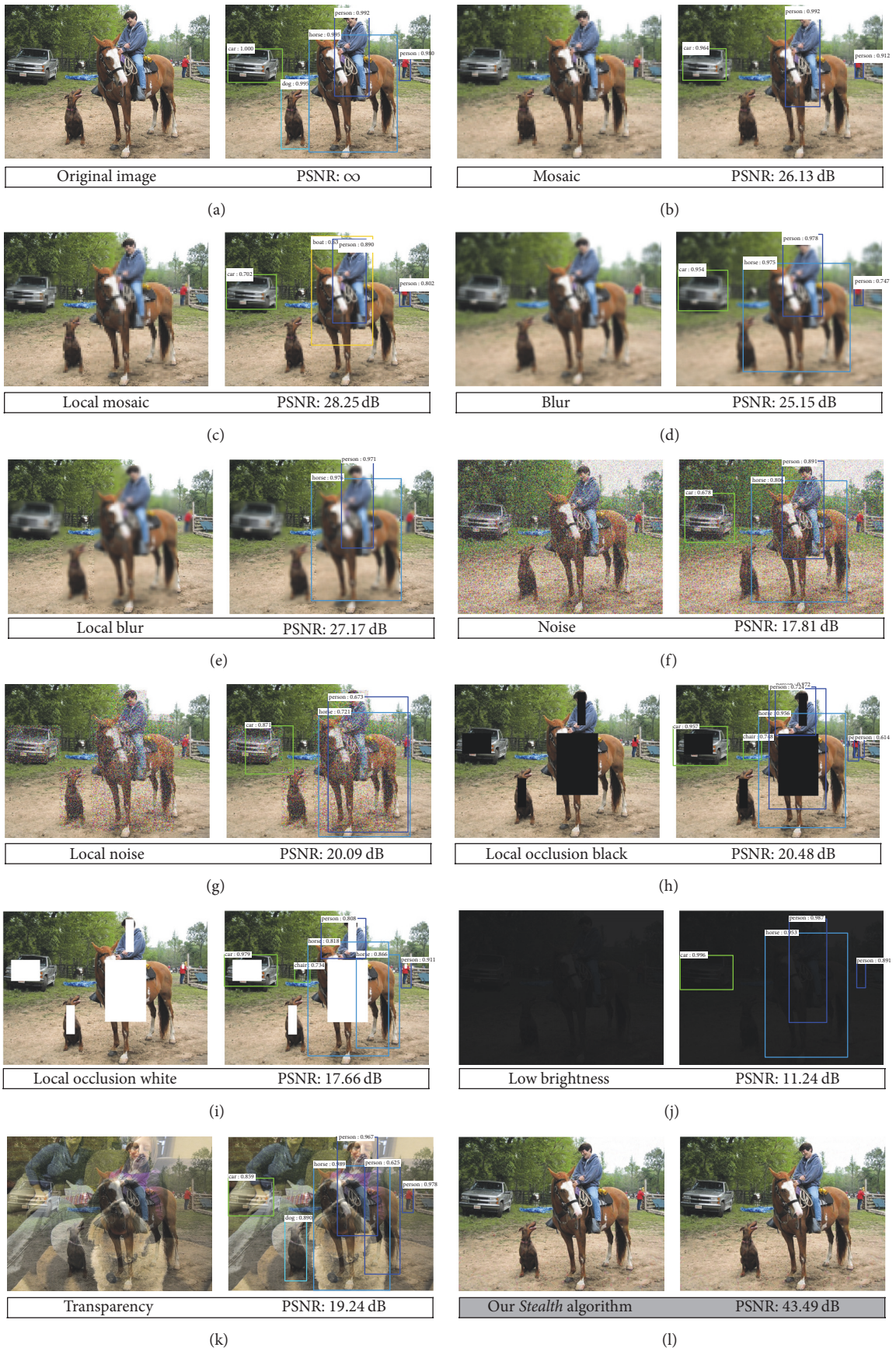


Figure 6: Images processed by diverse methods of disturbing are detected by the detection framework based on Faster RCNN. Each two horizontal images compose a pair, respectively, representing processed images and the results from the detector.

adding Gaussian noise is not a good way to deceive the detector, either.

(iii) As for a large area of occlusion on key objects, whether black occlusion in Figure 6(h) or white occlusion in Figure 6(i), they both make the quality deteriorate drastically. In spite of a large amount of information loss, the detection result is still almost accurate surprisingly.

(iv) From Figure 6(j), we can see that adjusting the image brightness to a fairly low level cannot resist the detector, either. It causes the greatest damage to the image simultaneously so that human eyes cannot see anything in the image at all. But the detector gives rather accurate results.

(v) In order to make the machine unaware of the existence of objects in the image, another natural idea is to make objects become transparent in front of the machine. So we try to change its transparency and hide it in another image, as shown in Figure 6(k). And yet it still does not work.

(vi) On the contrary, from Figure 6(l), we can see that our *Stealth algorithm* substantially has the smallest damage to image quality and it is also resistant to detection effectively. In order to better illustrate its effectiveness, we have carried out other larger-scale experiments which will be described next.

5.3. Privacy Insurance. In order to depict the degree of privacy protection in our algorithm, we define a parameter, *cloak thickness* τ, to weight the trap-door between privacy and visual quality. Users can tune this parameter to determine the adversarial disturbance intensity on each pixel. For a specific τ, the modification to each pixel is obviously uneven. What we need to do is multiplying τ by the gradient value of DNN backpropagation. This is equivalent to expanding the gradient of each pixel by τ times simultaneously, and it is considered as the final modification added to the image. Greater gradient value of pixel means further distance away from our target, so we need to add more adversarial interference on this pixel. Certainly, different τ values also influence the results. The added interference is proportional to τ value. The greater τ, the thicker the *cloak* the image is wearing, and the machine will be more blind to it. But, of course, the visual quality will go down.

We test on nearly 5000 images and calculate the PI using ZF-net and VGG-net, and the results can be found in Table 2. The 20 classes include airplane, bicycle, bird, boat, bottle, bus, car, cat, chair, cow, dining table, dog, horse, motorbike, person, potted plant, sheep, sofa, train, and tv monitor. Except for very few classes, the PI values of the vast majority are fairly high. This roughly means that we have successfully protected the users' most information in images.

Assume that a user shares many pictures and then tries to protect his privacy by using different methods of perturbing images. We test the PI values of all these methods, as shown in Figure 7. We can see from it that our *Stealth algorithm* can protect most privacy, and mosaic comes second, but it nevertheless has destructive effects on image. Other methods not only fail to protect privacy, but also cause terrible visual quality of images that users cannot put up with. Of course, users can get more insurance for their privacy by increasing the *cloak thickness* τ, but they may have to face the risk of image quality deteriorating, as shown in Figure 8. From this figure, we can find $\tau = 0.3\times10^3$ could be an appropriate value, at which we can not only get a satisfactory *privacy insurance* but also ensure the visual effects. Even if the value of *cloak thickness* is fairly large (e.g., $\tau = 1.2 \times 10^3$), the PSNR is still greater than any other methods. The *Stealth algorithm*'s modification to a pixel is related to the current value of the pixel, so it does not seem so abrupt after the processing.

From the above experimental results, we can see our algorithm works well, but the fact that there exist classes with low PI value (e.g., Class 8 "cat," Class 12 "dog," and Class 14 "motorbike") is worth thinking about. Here we present some illustrations and thoughts on this question. The extracted feature of each region proposal corresponds to a point in a high dimensional space. The correctness of the judgment is related to the classification boundary. Our work is to change positions of these corresponding points by adding perturbation to an image, so that the points can cross the boundary and jump to another class (from be-object class to not-object class).

Our algorithm is independent of the specific class of the object. That is to say, to offset the generation of region proposal, we use the same number of iterations (Γ) and multiple times (τ) when we superimpose the gradient disturbance for all classes. In the abstract high dimensional space, features of different classes occupy different subspaces, which are large or small. So perturbations with the same iterations and multiple times are bound to cause a problem where features of some classes are successfully counteracted, while some few other classes may fail. The reason for failure may be that the number of iterations is insufficient or the magnitude of modification is not enough for these classes. For each region proposal feature in the detector, Figure 9 gives a vivid illustration of the following four cases.

Case 1. The region proposal features of some classes are successfully counteracted after the image is processed. In other words, the corresponding feature point jumps from be-object subspace to not-object subspace. In this case, our algorithm can be deemed a success.

Case 2. Region proposal features of some classes are counteracted partly. So the feature point jumps to a be-object subspace, but features in this subspace are not strong enough to belong to any specific class. That is to say, these proposals will be discarded in the following classification stage for their scores of each class are lower than our set threshold. In this case, the final result is that objects cannot be detected, so it is an indirect success.

Case 3. The feature point jumps from one object class to another. Result is that the detector will give a bounding box approximately, but its label might be incorrect. This case is just a weak success.

Table 2: Privacy insurance of each category after using *Stealth algorithm* on ZF-net and VGG-net.

PI_k	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	PI
ZF	0.99	0.75	0.87	1.00	0.84	0.97	0.85	0.47	0.86	0.85	0.98	0.23	0.74	0.34	0.70	0.95	0.90	1.00	0.99	0.96	**0.82**
VGG	1.00	0.70	0.87	0.87	1.00	0.73	0.85	0.31	0.92	0.95	0.99	0.95	0.92	0.39	0.93	0.94	0.98	0.99	0.95	0.80	**0.85**

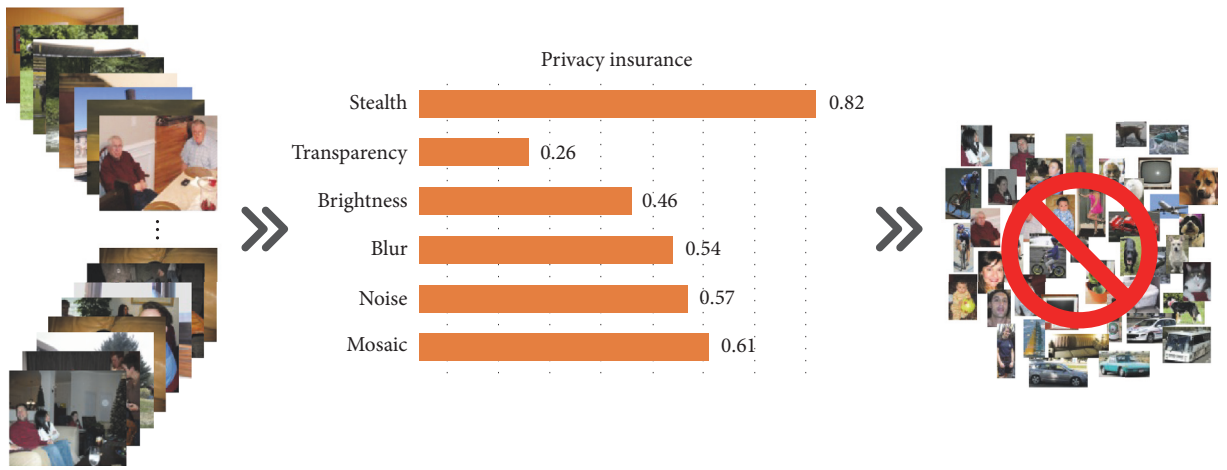


Figure 7: Different ways of fooling detection machine. Assume that the user shares many pictures and then tries to protect their privacy by different methods of image scrambling. Obviously our veil algorithm can protect the most privacy. Mosaic comes second, but it has destructive effects on image itself.

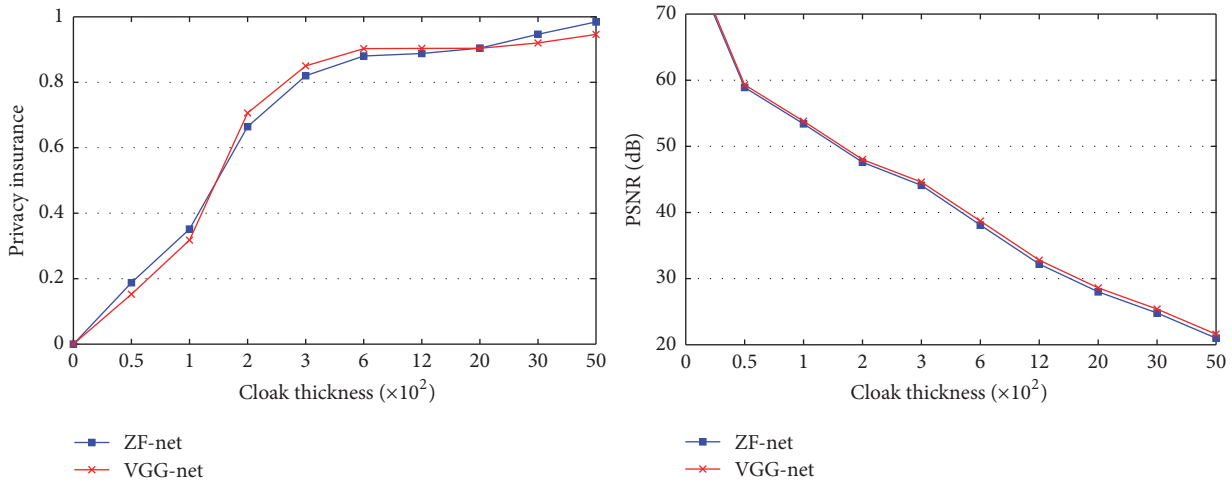


Figure 8: Privacy insurance versus PSNR with different cloak thickness.

Case 4. The feature point only jumps within an object class subspace. Its range might be larger than others or its position is farther away from the boundary of not-object class subspace. It is kind of equivalent to saying that the trained detector has better robustness for this specific class. An adversarial algorithm may fail when encountering this case.

The classes with low PI value after our *Stealth algorithm* may fall into Case 4. The iteration and multiple times which we set are not enough to make the proposal feature jump out of its original subspace. However, in order to ensure a good vision quality, we should not set them very high. It is a trade-off between human vision and machine vision.

5.4. Transferability of Cloak. The *Stealth* interference generated for one particular DNN also has an impact on another DNN, even if their network architectures are quite different. We call it the transferability of different *cloaks*. When we put the adversarial images generated for ZF-net, which is with a slightly larger *cloak thickness*, onto the VGG-net for detection, we can calculate that its *privacy insurance,* PI, is 0.66. And, at this time, the visual quality is still satisfactory. There may exist some subtle regular pattern only when seeing it from a very close distance, but it is much better than mosaic, blur, and other methods for human eyes. Likewise, we detect the VGG adversarial images on ZF-net, and the PI value is 0.69.

So far we have been focusing on the white-box scenario: the user knows the internals, including network architecture and parameters of the system. To some extent, the transferability here leads to the implementation of a black-box system. We do not need to know the details of network. What we only need to know is that the detection system we try to deceive is based on some kind of DNN. Then we can generate an adversarial example for the image to be uploaded against our local DNN. According to the above experimental results, the generated images on local machine are very likely to deceive the detection system of online social network.

Figure 9: An intuitive understanding of adversarial images for detection task in the high dimensional space. (a) Different cases that feature point moves between the be-object class and not-object class in the high dimensional feature space. (b) Different cases that feature point moves among different specific classes. Each subspace with a color represents a specific class. The subspace in the be-object region but not belonging to any specific class represents its score of belonging to any class which is lower than our set threshold.

6. Conclusion and Future Work

In this paper, we propose the *Stealth algorithm* of elaborating adversarial examples to resist the automatic detection system based on the Faster RCNN framework. Similar to misleading the classification task in previous work, we also add some interference to cheat the computer vision of ignoring the existence of objects contained in images. Users can process images to be uploaded onto social networks through our algorithm, thus avoiding the tracking of online detection system, so as to meet the goal of minimizing privacy disclosure. In effect, it is like objects in images wearing an invisibility *cloak* and everything disappearing in machine's view. As a comparison, we conduct experiments of modifying images with several other trivial but intriguing methods (e.g., mosaic, blur, noise, low brightness, and transparency). The result shows our *Stealth* scheme is the most effective and has minimal impact on image visual quality. It can guarantee both high image fidelity to human and invisibility to machine with high probability. We define a user adjustable parameter to determine the adversarial disturbance intensity on each pixel, that is, *cloak thickness*, and a measurement to indicate how much privacy can be protected, that is, *privacy insurance*. And we have further explored the relation between them. In addition, we find the adversarial examples crafted by our *Stealth algorithm* have transferability property; that is, the

interference generated for one particular DNN also has an impact on another DNN.

One of our further researches will be a theoretical analysis about the transferability property between different network models. And, according to it, we will try to find a method of crafting adversarial examples with good generalization performance on many different DNNs. Even if its fooling performance on any one of DNN models will not be as good as the specific adversarial example, it can maximize the average performance on all models. Furthermore, it is evident that our algorithm is a global processing on images. So another ongoing study should be conducted to only add partial adversarial perturbation to achieve the same deceiving effect. That is to say, we try to modify only part of pixels, instead of processing the image globally. But this requirement may lead to significant changes on a few pixels, which will cause an uncomfortable visual effect. So we should try to find out some ways to make the processed image look more natural.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

Acknowledgments

This work was supported in part by the Natural Science Foundation of China under Grants U1636201 and 61572452. Yujia Liu, Weiming Zhang, and Nenghai Yu are with CAS Key Laboratory of Electromagnetic Space Information, University of Science and Technology of China, Hefei 230026, China.

References

[1] Zephoria, "The Top 20 Valuable Facebook Statistics," 2017, https://zephoria.com/top-15-valuable-facebook-statistics/.

[2] B. Krishnamurthy and C. E. Wills, "On the leakage of personally identifiable information via online social networks," in *Proceedings of the 2nd ACM SIGCOMM Workshop on Online Social Networks, WOSN '09*, pp. 7–12, 2009.

[3] B. Henne and M. Smith, "Awareness about photos on the web and how privacy-privacy-tradeoffs could help," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface*, vol. 7862, pp. 131–148, 2013.

[4] M. Hardt and S. Nath, "Privacy-aware personalization for mobile advertising," in *Proceedings of the 2012 ACM Conference on Computer and Communications Security, (CCS '12)*, pp. 662–673, October 2012.

[5] "Japan researchers warn of fingerprint theft from 'peace' sign," 2017, https://phys.org/news/2017-01-japan-fingerprint-theft-peace.html./.

[6] W. Meng, X. Xing, A. Sheth, U. Weinsberg, and W. Lee, "Your online interests-Pwned! A pollution attack against targeted advertising," in *Proceedings of the 21st ACM Conference on Computer and Communications Security, (CCS '14)*, pp. 129–140, November 2014.

[7] A. Reznichenko and P. Francis, "Private-by-design advertising meets the real world," in *Proceedings of the 21st ACM Conference on Computer and Communications Security, CCS ('14)*, pp. 116–128, November 2014.

[8] Icondia, "Smile Marketing Firms Are Mining Your Selfies," 2016, http://www.icondia.com/wp-content/uploads/2014/11/Image-Mining.pdf

[9] N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Optimal geo-indistinguishable mechanisms for location privacy," in *Proceedings of the 21st ACM Conference on Computer and Communications Security, CCS 2014*, pp. 251–262, November 2014.

[10] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Geo-indistinguishability: differential privacy for location-based systems," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS '13)*, pp. 901–914, ACM, Berlin, Germany, November 2013.

[11] M. J. Wilber, V. Shmatikov, and S. Belongie, "Can we still avoid automatic face detection?" in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision, (WACV '16)*, March 2016.

[12] I. Polakis, P. Ilia, F. Maggi et al., "Faces in the distorting mirror: revisiting photo-based social authentication," in *Proceedings of the 21st ACM Conference on Computer and Communications Security, CCS ('14)*, pp. 501–512, November 2014.

[13] P. Ilia, I. Polakis, E. Athanasopoulos, F. Maggi, and S. Ioannidis, "Face/off: preventing privacy leakage from photos in social networks," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, (CCS '15)*, pp. 781–792, October 2015.

[14] L. Zhang, K. Liu, X.-Y. Li, C. Liu, X. Ding, and Y. Liu, "Privacy-friendly photo capturing and sharing system," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp 2016*, pp. 524–534, September 2016.

[15] R. Collobert and J. Weston, "A unified architecture for natural language processing: deep neural networks with multitask learning," in *Proceedings of the 25th International Conference on Machine Learning*, pp. 160–167, ACM, July 2008.

[16] A. Hannun, C. Case, J. Casper, B. Catanzaro et al., "Deep speech: scaling up end-to-end speech recognition," 2014, https://arxiv.org/abs/1412.5567.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the 26th Annual Conference on Neural Information Processing Systems (NIPS '12)*, pp. 1097–1105, Lake Tahoe, Nev, USA, December 2012.

[18] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems*, pp. 91–99, 2015.

[19] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '15)*, pp. 3431–3440, IEEE, Boston, Mass, USA, June 2015.

[20] J. Ian Goodfellow, S. Jonathon, and S. Christian, "Explaining and harnessing adversarial examples," 2014, https://arxiv.org/abs/1412.6572.

[21] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification," in *Advances in neural information processing systems*, pp. 1988–1996, 2014.

[22] B. B. Zhu, J. Yan, Q. Li et al., "Attacks and design of image recognition CAPTCHAs," in *Proceedings of the 17th ACM*

Conference on Computer and Communications Security, CCS'10, pp. 187–200, October 2010.

[23] G. E. Dahl, J. W. Stokes, L. Deng, and D. Yu, "Large-scale malware classification using random projections and neural networks," in *Proceedings of the 2013 38th IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2013*, pp. 3422–3426, May 2013.

[24] T. S. Guzella and W. M. Caminhas, "A review of machine learning approaches to Spam filtering," *Expert Systems with Applications*, vol. 36, no. 7, pp. 10206–10222, 2009.

[25] S. Christian, Z. Wojciech, I. Sutskever et al., "Intriguing properties of neural networks," 2013, https://arxiv.org/abs/1312.6199.

[26] N. Papernot, P. McDaniel, J. Somesh, M. Fredrikson, Z. Berkay Celik, and S. Ananthram, "The limitations of deep learning in adversarial settings," in *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on IEEE*, pp. 372–387.

[27] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015*, pp. 427–436, June 2015.

[28] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Universal adversarial perturbations," 2016, https://arxiv.org/abs/1610.08401.

[29] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "Practical black-box attacks against deep learning systems using adversarial examples," 2016, https://arxiv.org/abs/1602.02697.

[30] N. Papernot, P. McDaniel, and I. Goodfellow, "Transferability in machine learning: from phenomena to black-box attacks using adversarial samples," 2016, https://arxiv.org/abs/1605.07277.

[31] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," 2016, https://arxiv.org/abs/1607.02533.

[32] 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 23rd ACM Conference on Computer and Communications Security, (CCS '16)*, pp. 1528–1540, October 2016.

[33] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the 27th IEEE Conference on Computer Vision and Pattern Recognition (CVPR '14)*, pp. 580–587, Columbus, Ohio, USA, June 2014.

[34] R. Girshick, "Fast R-CNN," in *Proceedings of the 15th IEEE International Conference on Computer Vision (ICCV '15)*, pp. 1440–1448, December 2015.

[35] S. Christian, R. Scott, and E. Dumitru, "Scalable, high-quality object detection," 2014, https://arxiv.org/abs/1412.1441.

[36] Y. Li, H. Kaiming, S. Jian et al., "R-fcn: Object detection via region-based fully convolutional networks," *Advances in Neural Information Processing Systems*, pp. 379–387, 2016.

[37] W. Liu, D. Anguelov, D. Erhan et al., "SSD: single shot multibox detector," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface*, vol. 9905, pp. 21–37, 2016.

[38] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*, pp. 779–788, July 2016.

[39] H. Jonathan, R. Vivek, and S. Chen, "Speed/accuracy trade-offs for modern convolutional object detectors," 2016, https://arxiv.org/abs/1611.10012.

[40] M. Everingham, L. van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (VOC) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.

[41] Y. Jia, E. Shelhamer, J. Donahue et al., "Caffe: convolutional architecture for fast feature embedding," in *Proceedings of the ACM International Conference on Multimedia*, pp. 675–678, ACM, Orlando, Fla, USA, November 2014.

[42] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer Vision—ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part I*, vol. 8689 of *Lecture Notes in Computer Science*, pp. 818–833, Springer, 2014.

[43] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, https://arxiv.org/abs/1409.1556.