1-2021

On the (Im)Practicality of Adversarial Perturbation for Image Privacy

Arezoo Rajabi
Oregon State University

Rakesh B. Bobba
Oregon State University

Mike Rosulek
Oregon State University

Charles Wright
Portland State University, cvw@pdx.edu

Wu-chi Feng
Portland State University, wuchi@pdx.edu

Citation Details

Rajabi, A., Bobba, R. B., Rosulek, M., Wright, C. V., & Feng, W. (2021). On the (Im)Practicality of Adversarial Perturbation for Image Privacy, Proceedings on Privacy Enhancing Technologies, 2021(1), 85-106.

Arezoo Rajabi*, Rakesh B. Bobba*, Mike Rosulek, Charles V. Wright, and Wu-chi Feng

On the (Im)Practicality of Adversarial Perturbation for Image Privacy

Abstract: Image hosting platforms are a popular way to store and share images with family members and friends. However, such platforms typically have full access to images raising privacy concerns. These concerns are further exacerbated with the advent of Convolutional Neural Networks (CNNs) that can be trained on available images to automatically detect and recognize faces with high accuracy.

Recently, *adversarial perturbations* have been proposed as a potential defense against automated recognition and classification of images by CNNs. In this paper, we explore the practicality of adversarial perturbation-based approaches as a privacy defense against automated face recognition. Specifically, we first identify practical requirements for such approaches and then propose two practical adversarial perturbation approaches – (i) learned *universal ensemble perturbations (UEP)*, and (ii) *k-randomized transparent image overlays (k-RTIO)* that are *semantic adversarial perturbations*. We demonstrate how users can generate effective transferable perturbations under realistic assumptions with less effort.

We evaluate the proposed methods against state-of-the-art online and offline face recognition models, Clarifai.com and DeepFace, respectively. Our findings show that UEP and k-RTIO respectively achieve more than 85% and 90% success against face recognition models. Additionally, we explore potential countermeasures that classifiers can use to thwart the proposed defenses. Particularly, we demonstrate one effective countermeasure against UEP.

Keywords: Image Privacy, Convolutional Neural Networks (CNNs), Adversarial Examples, Transferable Perturbations, Transparent Image Overlays

DOI 10.2478/popets-2021-0006

Received 2020-05-31; revised 2020-09-15; accepted 2020-09-16.

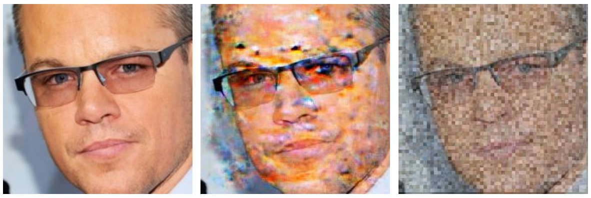


Fig. 1. From left to right: original image, UEP perturbed image, and k-RTIO perturbed image.

1 Introduction

Integration of high-resolution cameras into ubiquitous handheld devices (*e.g.,* mobiles, tablets) has made it easy to capture everyday moments in digital photos. Combined with the availability of fast mobile networks many of the captured digital images and videos are shared over the network or transmitted to third-party storage providers. It has been reported that users of social networks share more than 1.8 billion new photos each day [2]. While cloud-based photo storage and sharing platforms provide virtually unlimited storage space for saving and sharing images and are highly reliable and available, such services also raise privacy concerns. Although most social networks and photo storage and sharing platforms now provide privacy settings to limit public accessibility of images, they are often complicated and insufficient [54, 65]. Furthermore, those platforms have direct access to users' images and it is not inconceivable for such service providers to exploit the users' image data for their own benefit [62, 68].

The advent of scalable Convolutional Neural Networks (CNNs) for automated face detection and recognition [35, 38, 45, 70, 77] exacerbates this concern. While such tools provide benign and beneficial service to users, they nevertheless pose a significant privacy threat. For example, Shoshitaishvili et al. [65] recently proposed a method for tracking relationships among people by analyzing pictures shared on social platforms using face recognition and detection CNNs. Recently, a face-recognition based image search application, Clearview AI app [26], has emerged that can take a given picture

***Corresponding Author: Arezoo Rajabi:** Oregon State University, rajabia@oregonstate.edu

***Corresponding Author: Rakesh B. Bobba:** Oregon State University, rakesh.bobba@oregonstate.edu

Mike Rosulek: Oregon State University, rosulekm@oregonstate.edu

Charles V. Wright: Portland State University, cvw@pdx.edu

Wu-chi Feng: Portland State University, wuchi@pdx.edu

of a person and search a database of more than 3 billion images scraped from Facebook, YouTube and millions of other websites and provide matching images along with links to websites containing those photos. News of the use of this application by law enforcement agencies has raised concerns about severe erosion of privacy. Emergence of such applications combined with the ineffective privacy settings of social networks, necessitates the development of practical tools that allow end users to protect their privacy against automated classifiers. In this paper, we primarily focus on thwarting automated face recognition.

Adversarial image perturbation has been recently proposed as a way to protect against automated face recognition by fooling CNNs into misclassifying the image [20, 31, 63]. Adversarial perturbations [22, 23] have the nice property that the perturbed images are perceptually indistinguishable from the original image but are misclassified by a CNN. Thus adversarial perturbation provides a nice balance of privacy against automated classifiers without degrading the usability of the images for end users. However, these methods are not always practical for real-world use due to the unrealistic assumption of full knowledge about the adversaries' CNNs (*i.e.,* white-box model).

While black-box approaches for adversarial perturbations do exist (*e.g.,* [42, 48, 56]), they either assume that end users have access to large datasets (*e.g.,* millions of images) and can train large local CNNs (*e.g.,* with thousands of classes), or that they can query the target CNN making them unrealistic as well. Semantic adversarial perturbations [28, 29] are a newer class of adversarial perturbations that can target unknown CNNs. They do not require any learning and are therefore computationally very efficient. However, they are not reversible making them unsuited for image storage applications.

Contributions: We explore the practicality of adversarial perturbations to thwart automated image classification by CNNs, and make the following contributions:

1. We identify key requirements for practical adversarial perturbation based image privacy against automated face recognition.
2. We propose two novel schemes for adversarial perturbation of images that do not require special knowledge of the target CNNs (*i.e.,* black-box adversarial methods) and that satisfy the identified requirements.
   (a) Our learning-based *Universal Ensemble Perturbation (UEP)* approach (2nd image in Figure 1) learns a single transferable adversarial perturbation that can be applied to multiple images. Compared to previous adversarial perturbation learning techniques, our UEP approach requires fewer and smaller local CNNs trained over smaller (by an order of magnitude) training sets.
   (b) Our *k-Randomized Transparent Image Overlays (k-RTIO)* is a novel *semantic adversarial perturbation* [28] approach that generates many unique image perturbations (3rd image in Figure 1) using only a small number of overlay source images and a secret key, and without requiring users to train their own CNNs. In contrast to previous semantic adversarial perturbation approaches, k-RTIO perturbations are reversible.
3. We evaluate the effectiveness of both methods against highly accurate celebrity recognition and face detection models from `clarifai.com`, `Google Vision API`, and DeepFace [57], two state-of-the-art online classifiers and one offline classifier, respectively. Our results show that while UEP and k-RTIO are effective at thwarting automated face recognition, k-RTIO is computationally much cheaper than UEP.
4. We discuss potential counter measures against our schemes and demonstrate one effective countermeasure against UEP.

The rest of this paper is organized as follows: We discuss related work in Section 2. In Section 3, we present the system model and requirements for a practical privacy mechanism based on adversarial perturbations. We present some preliminaries on adversarial examples and transferable adversarial methods in Section 4. We introduce our two proposed approaches in Section 5 and evaluate them in Section 6. In Section 7, we discuss our findings and some future directions. Finally, we conclude in Section 8.

2 Related Work

Many cryptographic and non-cryptographic techniques for protecting image privacy have been proposed. Here we primarily focus on works most relevant to the problem of thwarting recognition by automated classifiers while allowing recognition by humans.

Adversarial Perturbations: The problem of thwarting recognition by automated classifiers while allowing recognition by humans has recently received attention in the research community [20, 31, 44, 63, 74]. Adversarial perturbation based approaches [20, 31, 63] among them are most closely related to this paper. While those approaches can fool CNNs and preserve images' recognizability by humans they have some limitations. Oh *et al.* [31] assume that end users have full knowledge about and access to target CNNs (*i.e.,* white-box model). Other works [20, 63], while they use a black-box model, assume users are able to train large CNNs to learn perturbations. However, in real-world settings, users typically do not have access to the target CNNs or other automated tools used by adversaries or online service providers to analyze images. Moreover, all those approaches learn a perturbation for each image making it expensive to protect a large set of images. Fawkes [63] aims to perturb images to prevent target CNNs from learning a model on them and can be complementary to our effort. Semantic adversarial perturbations [28, 29] that can fool unknown CNNs without requiring any learning have been recently proposed. As they do not require any learning, they are computationally very efficient. However, they are not reversible making them ill-suited for image storage applications. In contrast, we aim to propose practical methods for generating reversible transferable perturbations that do not need high computation or storage resources.

Image Encryption: Encryption techniques specifically designed for protecting image privacy have also been proposed (*e.g.,* [58, 76]). These techniques obfuscate the entire image and as a consequence images are rendered unrecognizable even by their owners making them unable to distinguish between images without first decrypting them. Further, it has been shown that P3 [58] is vulnerable to automated face recognition [46].

To address the problem of searching through encrypted images, Pixek App [1] uses image encryption with embedded searchable tags. However, such schemes require modifications at the service providers (this is the same for encrypted file system [21, 33] based solutions as well) and need additional effort on part of end users to tag and browse images using tags. In contrast, our focus is on techniques that are not only reversible but also allow end users to manage their image repositories naturally (*i.e.,* visually rather than using keywords/tags).

Thumbnail Preserving Encryption (TPE) schemes [44, 74, 78] have been proposed to preserve privacy of images while allowing image owners to distinguish between ciphertexts of different images naturally (*i.e.,* visually) without having to decrypt them. They achieve this by ensuring that the ciphertext is itself an image with the same thumbnail as the original image. Depending on the size of the thumbnail preserved, TPE schemes are capable of thwarting recognition by CNNs [44, 74] while still allowing image owners to distinguish between encrypted images. Their goal was to thwart recognition by any end user other than those who are already familiar with the image, and hence preserve thumbnails with very low resolution. In contrast, our work aims to thwart only classifiers and not humans.

Irreversible Obfuscations: An early approach to thwarting facial recognition software while preserving facial details was face deidentification (*e.g.,* [16, 24, 25, 32, 50, 71]) that was proposed to provide privacy guarantees when sharing deanonymized video surveillance data. In this approach instead of obfuscating the image or parts of it, faces in the image are modified to thwart automated face recognition algorithms. These approaches extend the k-anonymity [72] privacy notion to propose the k-same family of privacy notions for images. At a high-level, to provide k-same protection for a face in an image it is replaced with a different face that is constructed as an average of k faces that are the closest to the original face. These approaches have the benefit of providing mathematically well-defined protection against face recognition algorithms. However, these approaches have the downside that these modify the original image and the transformation is typically not reversible. Further, the perturbed image is typically not correctly recognizable by humans. A slightly different but related approach is where a face that needs privacy protection in an image is replaced with a similar one from a database of 2D or 3D faces (*e.g.,* [8, 41]). AnonymousNet [40] is recent de-identification approach using generative adversarial networks (GANs). These approaches are also irreversible. In contrast, we aim to create reversible perturbations where the perturbed images are recognizable by humans.

Approaches like blurring, pixelation, or redaction have long been used for protecting privacy in images and text (*e.g.,* [37, 81]). However, such techniques are not only *irreversible* but have been shown to be ineffective especially against automated recognition methods [27, 37, 46, 50, 81].

Approaches that remove people or objects from images (*e.g.,* [7, 34, 59]) or that abstract people or objects (*e.g.,* [13, 75]), or that replace faces [69] also exist. However, those approaches are irreversible. Further, it has been shown that even if redaction techniques like replacing faces with black space are used, it may be possible to deanonymize using person recognition techniques [53].

3 System Model and Requirements

In this work, our focus is on practical adversarial perturbation approaches that make scalable automated image analysis by service providers more difficult. Thwarting recognition by humans is not a goal. In fact, we aim to develop approaches that allow image recognition by end users so they can interact with images naturally. We assume that image storage and sharing service providers do not have access to meta-data or auxiliary information and treat them as an honest-but-curious or semi-honest adversary [52]. We assume that the service provider has trained an image classification CNN either using publicly available user images or images that the users have already stored with the service. A user's goal is to perturb the images that they upload to the service to prevent automated recognition or classification of their images. To be practical the perturbation mechanism should satisfy the following requirements.

Black-box Scheme: In practice, end users will not have access to the CNNs used by third-party service providers and typically will not have detailed knowledge of the target CNNs parameters and weights. Thus, any viable perturbation mechanism should not require any special knowledge of the target CNN, eliminating the use of white-box adversarial perturbation techniques. We assume that the user does not know anything about the structure of the target CNN, not even all the classes that the CNN is able to classify images into (*i.e.,* it is a black-box), and assume that the user does not have query access to the CNN.

Recognizability: While white-box (*i.e.,* full knowledge of target CNN) adversarial perturbations typically are very small and imperceptible [23], black-box perturbations on the other hand can be significant and are typically perceptible [48]. To be viable, the perturbed images should be perceptually recognizable by end users but not recognizable by automated classifiers. Users should be able to browse through perturbed images naturally using the standard interface, without having to reverse the perturbation.

Recoverability or Reversibility: For image storage applications, end users must be able to reverse the perturbation in order to recover the original image with minimum to no distortion, *i.e.,* no loss of perceptual quality. At the same time, the service provider should not be able to remove or reduce the perturbation.

Low Computational Cost: Since end users need to create the perturbations, learning or creating perturbations should not require excessive computational effort or large volumes of training data. We assume that, while an individual user may have access to their own images and some images of other potential users of the platform (either publicly available or those of friends using the same platform), a user does not have access to the entire dataset used by the service provider for training the target CNN.

Low Storage Cost: The storage requirements of the scheme should be small and should not increase with the number of images. Any information needed to reverse the perturbation should take up minimal storage at the end user.

Compatibility: Since service providers have incentives to learn and profit from user data, proposed schemes should not depend on their support and should be compatible with existing services to ease adoption. Hence, using embedded thumbnails, encrypted file systems (*e.g.,* [21, 33]) and more advanced solutions such as searchable encryption for images [1] are not practical. Most of the popular storage services that we tested such as Google Drive, iCloud, *etc.* do not support embedded thumbnails.

4 Preliminaries

CNNs: Convolutional Neural Networks (CNNs) have become very popular for image classification. CNNs are constructed by several convolutional and maxpooling layers and one or two fully connected layers (see Figure 18 in Appendix A). At the end, there is a softmax layer which translates the output of the last layer to probabilities. Generally, a convolutional neural network is defined by a function $F(x, \theta) = Y$ which takes an input (x) and returns a probability vector ($Y = [y_1, \cdots, y_m]$ $s.t.$ $\sum_i y_i = 1$) representing the probability of the input belonging to each of the m classes. The input is assigned to the class with maximum probability.

To learn classifiers, the loss function which represents the cost paid for inaccuracy in classification problems, is minimized. The cross entropy loss function is one of the popular and sufficient loss functions that is used in classification problems. For training a CNN, we optimize the CNN's parameters (θ) for given inputs.

Adversarial Perturbations: Adversarial generation methods find and add a small perturbation (δ) to an image that will cause the target CNN to misclassify it.

$$\begin{aligned} &\min_{\delta} \|\delta\|_2 + J(F(x, \theta)) \\ &s.t. \;\; argmax \;\; F(x + \delta) \neq y^*, \;\; x + \delta \in [0, 1]^d \end{aligned} \tag{1}$$

where δ, x, y^* and $J(.)$ are the perturbation, given input image, true label of input image, and the loss function, respectively. Many methods have been proposed to generate a small perturbation for a known CNN (*e.g.,* [11, 23, 49, 73]), but they suffer from low *transferability* to unknown CNNs (See Appendix A for different types of adversarial attacks). Therefore, transferable perturbation generation methods were proposed to address this issue and are discussed next.

Ensemble Method: Ensemble methods to create transferable perturbations were proposed in [6, 42]. Those works learn a perturbation for a given image on several local CNNs, instead of using only one CNN, such that the learned perturbation fools all local CNNs. They established that such perturbations are transferable and can fool unknown CNNs with high success rate. In other words, a perturbation for an image is generalized by learning it on multiple CNNs. To show the transferability of perturbations, authors assessed their method on, `clarifai.com`, a state-of-the-art online classifier.

Universal Perturbation: Moosavi-Dezfooli *et al.* [48], showed that it is possible to find a single (universal) perturbation that can be applied to multiple images to successfully fool a CNN into misclassifying all of them. Universal perturbation is defined as a noise pattern δ which leads a CNN misclassifying most of the input images (x's) with probability of p (p-value) when added to those input images. This work showed that a universal perturbation is transferable to other CNNs with different structures but trained on the same dataset as the original. Further, the perturbation is added directly to images which causes significant loss during image recovery because of rounding the pixels' value to 0 or 1 if they are not in range of $[0, 1]$.

While all of these approaches are black-box approaches, (i) they either need to train and learn on multiple $(4 - 5)$ large local CNN(s) (see Summary in Section 6.1) to create a transferable perturbation(s) [6, 42, 48] or (ii) generate one perturbation per image [6, 42] and do not meet *low computation* and *low storage* cost requirements. Further, they add the perturbation directly to image's pixel value leading to significant losses during recovery.

5 Proposed Approaches

In this section, we introduce two methods to perturb images in order to fool any unknown CNN. The first method is a learning-based approach called *Universal Ensemble Perturbation (UEP).* The second method is a semantic perturbation method called *k-Randomized Transparent Image Overlays (k-RTIO).* We motivate each approach and describe how we generate and apply the perturbations.

5.1 Universal Ensemble Perturbation

As discussed in Section 3, any viable perturbation approach for image privacy should be black-box as users are unlikely to have access to or knowledge about CNNs used by service providers. Since black-box perturbations tend to be significant and perceptible, reversibility of perturbations becomes important to recover the original image. Storing the perturbations used for each image so the original can be recovered will require nearly the same order of storage as storing the original images themselves. Therefore, in our approach we aim to generate a *single perturbation* that can be applied to several images to successfully fool an unknown CNN, that is, both *universal* and *transferable.* We achieve this by using an *ensemble* approach through learning on a few local CNNs. Unlike previous work [42, 48], our approach uses fewer and smaller CNNs, and requires significantly smaller training datasets than the one used to train local CNNs (see Section 6).

Perturbation Learning: We create our *Universal Ensemble Perturbation* (UEP) approach by building on the ideas from the ensemble approach [42], universal perturbation approach [48], and CW perturbation optimization and addition approach [11].

UEP learns a highly transferable, universal, and reversible perturbation using an ensemble of a few small local CNNs. Moreover, to prevent the target CNN from reducing the impact of the perturbation by using a reduced resolution version of the image [15], we learn a perturbation which works for different image resolutions by training each local CNN on different resolutions of the image. For this work, we used three small CNNs with only 10 classes each (see Section 6). We empirically found that using fewer than three CNNs makes learning transferable perturbations harder, while using more than three CNNs makes learning a perturbation

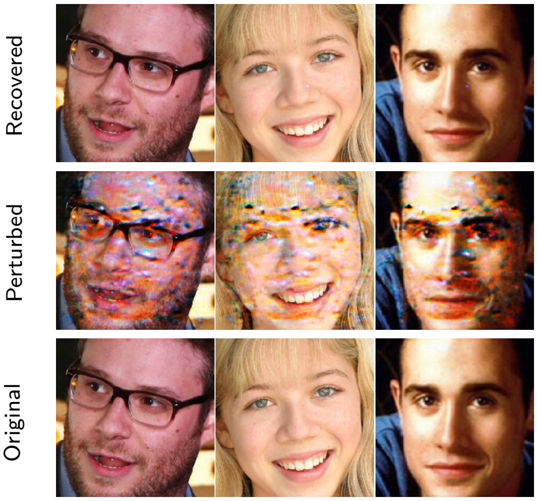


Fig. 2. From bottom to top: The last row shows the original images of celebrities. The middle row represents the UEP perturbed images. The first row displays the recovered images.

computationally expensive without significant improvement in success rate.

After learning a few (in our case only 3) local CNNs on different image resolutions, we learn a universal perturbation which can fool all the local CNNs with a given probability of p ($\in [0,1]$). To optimize the misclassification rate, we extend the objective function introduced in [11] for an ensemble of CNNs and a universal attack as follows:

$$f(x_1, \cdots, x_N) = \sum_l \sum_i \max(\max\{Y_l(x_i)_j : j \neq y_i^*\} - Y_l(x_i)_{y_i^*}, -\kappa) \quad (2)$$

where $Y_l(x_i)$ is the probability vector assigned by the l^{th} CNN to the i^{th} input which shows the probability of the input image belonging to each category. And y_i^* denotes the true label of x_i. We aim to minimize the perturbation amount as well as maximize the misclassification confidence (κ). The problem can be defined as follows:

$$\min_{\delta} \ \|\delta\|_2 + cf(x_{1,perturbed}, \cdots, x_{N,perturbed}) \quad (3)$$

A large value for coefficient c causes the resulting perturbation to have a larger value (to force all CNNs to misclassify it), while a very small value for this parameter may lead to null solution space since it tries to fool all CNNs for all inputs with a small perturbation. We optimize the perturbation for different coefficient c values using a binary-search (see Figure 3) and pick the smallest perturbation that can fool $p \times N$ of total samples (N).

```
UEP($Y_1(x), \cdots, Y_m(x), x_1, \cdots, x_n, p$):
  $\delta = inf$
  // finding the true label of each image
  $\forall\, i\ y_i^* = argmax(Y(x_i))$
  $lbound = 0.001,\ upbound = 100000$
  while $lbound < upbound$:
    $c = (upbound + lbound)/2$
    $\forall\, i,\ z_i = x_i$
    while $\sum_i \prod_k \sigma(Y_k(z_i) \neq y_i^*) < p \times N$
      $\delta' \leftarrow \min_\delta \ \|\delta\|_2 + cf(z_1, \cdots, z_N)$
      $z_i = \frac{1}{2}(\tanh(\operatorname{arctanh}(2 \times (x_i - 0.5)) + \delta')) + 0.5$
    if $\|\delta'\|_2 < \|\delta\|_2$:
      $\delta \leftarrow \delta'$
      $upbound = (upbound + lbound)/2$
    else
      $lbound = (upbound + lbound)/2$
  return $\delta$
```

Fig. 3. UEP method learns a universal perturbation δ on a few local CNNs ($Y_1, \cdots, Y_c$) for set of given inputs $x_1, \cdots, x_n$.

Perturbation Addition: In the universal perturbation approach [48], the perturbation vector is added directly to all images, and if the new value of pixels are not in the range of $[0,1]$, then the new values are rounded to the nearest acceptable values (0 or 1). However, rounding a pixel's value leads to information loss during image recovery. Therefore, we add the learned perturbation to the arc-tangent hyperbolic value of an image instead of adding directly to the images as in [11]:

$$x_{i,pert} = \frac{1}{2}(\tanh(\operatorname{arctanh}(2 \times (x_i - 0.5)) + \beta \times \delta)) + 0.5 \quad (4)$$

where $x_{i,pert}$ is the perturbed version of the image x_i and δ is the learned perturbation. Here β is a weighting factor that tunes transferability versus perturbation amount. While this does not eliminate rounding losses completely, it significantly reduces such loss.

Further, unlike the traditional adversarial approaches, UEP perturbation can be added in with different weights to the image to control the trade-off between transferability and recognizability. While learning UEP on local CNNs, we set $\beta = 1$ to create a powerful perturbation. But, when applying the learned perturbation to an image, we can set β value to trade-off between fooling success rate and recognizability (see Section 6). The original image is recovered by reversing the process on the perturbed image. Adding the perturbation to the arc-tangent hyperbolic value of an image, both simplifies the objective function and alleviates information loss during image recovery [11].

Practicality of UEP: As the preceding discussion shows, UEP is designed to be a black-box perturbation approach that produces highly transferable and universal perturbations. It meets the reversibility and low storage cost requirements identified in Section 3 as perturbations can be reversed by storing only the universal perturbation. While the perturbations produced by UEP are significant and perceptible, as seen in Figure 2, the perturbed images are still recognizable for humans and can be recovered with very low loss by removing perturbation. Further, since UEP uses three small local CNNs with only 10 classes and does not require users to have access to large datasets. UEP is also computationally more efficient compared to previous approaches as will see in Section 6. Further, UEP does not require any changes to the service provider and can work with existing services.

5.2 k-Randomized Transparent Image Overlays

While our UEP approach requires lower computational resources compared to previous learning-based approaches, learning perturbations is still computationally expensive (see Section 6.1). Semantic perturbations [29] on the other hand do not rely on learning and tend to be computationally cheaper. Different semantic perturbation approaches target different weaknesses of CNNs (*e.g.,* coverage of training data, reliance on inter-pixel dependencies etc.) [28].

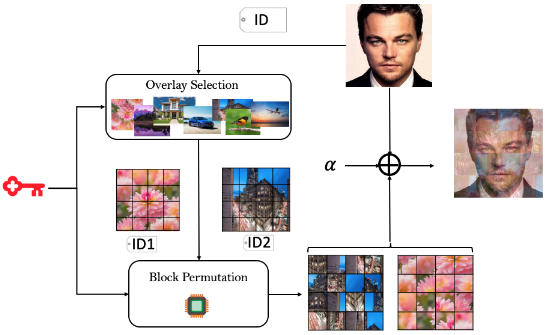


Fig. 4. k-RTIO Scheme. ID of main image is used to select k overlay images from the set S and then ID of selected images are used to generate a permutation. Both overlay selection and block permutation algorithm use the user's secret key.

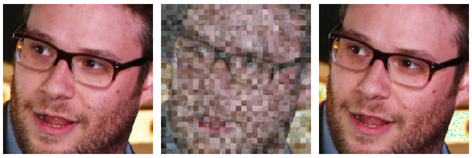


Fig. 5. Left to right: original, k-RTIO perturbed and recovered images.

The main component of CNNs are kernels in the convolutional layers. Kernels are small windows moving over image space and measuring dependencies among adjacent pixels to find a pattern or a feature. Transparent image overlay-based approach tends to decrease the pixel dependencies and makes features invisible to CNN kernels. Here we build on this transparent overlays based approach that was originally proposed for fooling Google Video API [29]. When translated to images, this approach adds a weighted overlay image to the original image as follows:

$$I_{pert} = \alpha \times I_{org} + (1 - \alpha) \times I_{ovl} \tag{5}$$

where α is mixing parameter, and I_{pert}, I_{org} and I_{ovl} are perturbed, original, and overlay images.

Despite the high transferability (success in fooling unknown CNNs) especially at low values of α, direct application of this technique has the following drawback. If the same overlay image is used for perturbing multiple original images then the original images can potentially be recovered through a well known signal processing technique called Blind Signal Separation [10] (see Figure 21 in Appendix D). If a different overlay image is used for each original image, then all those overlay images need to be stored locally to be able to recover the original images defeating the purpose of cloud-storage and violating our low storage requirement. Further, we need to make sure that perturbed image is recognizable by end users. To overcome this problem, we propose *k-Randomized Transparent Image Overlays* which generates a unique overlay for every image using only a small set of initial overlay images, a secret seed and a pseudorandom function.

Perturbation Generation: To generate *unique* overlays for each original image, we first choose a random subset of k (typically 3 or 4) candidate overlay images from a set S of stored initial overlay images. This set S is relatively small (30 images) and can be stored encrypted on a cloud platform or locally. The choice of this set does not significantly change the fooling performance of k-RTIO; we discuss selection of this set in Section 6.2. As shown in Figure 4, to select these k images for each source image, users use their secret key and ID of the original image (each image has a different

ID) as inputs to a pseudorandom function. More concretely, we employ AES as the pseudorandom function. Let key denote a 128-bit seed and let id be a unique identifier for the given source image (e.g., a filename or timestamp). Then the choice of the jth overlay image (out of $|S| = m$ possibilities) is made according to:

$$\mathsf{AES}(key, id\|0\|j) \bmod m.$$

After selecting k random candidate images, the k chosen images are divided into b blocks each and these blocks are permuted using a pseudorandom permutation. The permutation of blocks is based on the candidate overlay image ID, source image ID, and user's secret key. Specifically, the blocks of the jth overlay image are permuted using the Fisher-Yates shuffling algorithm [19]. Recall that in the ith iteration of Fisher-Yates, the algorithm chooses a random value in the range $\{0, \ldots, b-i\}$, where b is the number of items being shuffled. This random choice can be derived in a repeatable way (for the jth overlay image) via:

$$\mathsf{AES}(key, id\|1\|j\|i) \bmod b - i + 1.$$

As long as the ids of a user's images are distinct, then all inputs to AES are distinct. Use of a pseudorandom function guarantees that all its outputs are indistinguishable from random, when invoked on distinct inputs. Using this approach, the user needs to store only the short AES key, and the set of initial images S from which all unique overlay images can be re-derived as needed for recovering the original image (see Figure. 7).

The final overlay image is simply the average of these k images with permuted blocks. Although the pool S of base overlay images may be small, the number of derived overlay images that result from this process is very large. For example, for an image set of $|S| = 10$, $k = 4$ and $b = 12$, the number of possible overlay images is $\frac{10!}{10!(10-4)!} \times (12!)^4 \approx 7.3 \times 10^{31}$ making it hard for the target CNN to correctly guess the overlay used even if S is known.

Perturbation Addition: As shown in Figure 4, the generated overlay is added to the source image as follows:

$$I_{pert} = \alpha \times I_{org} + \frac{(1-\alpha)}{k} \sum_{I_{ovl_i} \in \mathsf{Overlays}(key,id,b,S)} I_{ovl_i}$$

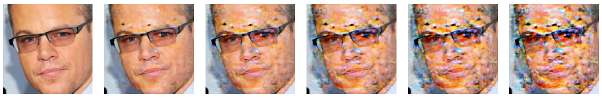

(a) Original (b) $\beta = 1$ (c) $\beta = 2$ (d) $\beta = 3$ (e) $\beta = 4$ (f) $\beta = 5$

Fig. 6. UEP recognizability vs. β that controls noise level

where $\mathsf{Overlays}(key, id, b, S)$ is a function that takes secret key, image id, number of blocks and overlays set S and returns k overlays images with permuted blocks. The perturbation can be removed to recover the original image using the reverse process as follows:

$$I_{org} = \frac{I_{per}}{\alpha} - \frac{(1-\alpha)}{k \times \alpha} \sum_{I_{ovl_i} \in \mathsf{Overlays}(key,id,b,S)} I_{ovl_i}$$

Practicality of k-RTIO: As the preceding discussion shows, k-RTIO does not require knowledge of the target CNN, nor does it require support from the service provider. Further, k-RTIO generates semantic perturbations and does not require learning using any CNNs and is therefore computationally cheaper than UEP (see Section 6.2). Since the perturbations are efficient to generate, end users do not need to store individual perturbations for recovering original images but can re-create them using a secret key and a small set of initial overlay images leading to low near constant storage overhead. The recognizability of images perturbed using k-RTIO is a function of α, b and k. In Section 6.2 we will explore values of these parameters such that k-RTIO can thwart recognition by automated classifiers while providing reasonable recognizability for humans.

6 Evaluation

In this section, we evaluate the success rate of the two proposed methods against two state-of-the-art online image classifiers (www.clarifai.com and Google Vision API) and a state-of-the-art offline recognition and detection model DeepFace [57]. We then discuss the computational costs of each approach. Finally, we explore different filtering techniques that classifiers could potentially use as countermeasures against the proposed perturbations.

6.1 UEP Performance

UEP Simulation Setup: *UEP* requires training a few small local CNNs. We trained 3 small CNNs, two VGG-16 [66] and one Facescrub CNN [46] using random initialization points. VGG-16 has 5 convolutional layers and 2 fully connected layers (see Figure 18) and FaceScrub CNN has 3 convolutional layers and 2 fully

```
Overlays($key, id, b, S$):
  $K = \emptyset$
  for $j = 1$ to $k$:
    // randomly choose $k$ overlay candidates
    $v = \mathsf{AES}(key, id \| 0 \| j) \bmod |S|$
    $\mathsf{IMG}$ = $v$th element of $S$
    remove $v$th element from $S$

    // keyed Fisher-Yates shuffle on blocks
    for $i = 1$ to $b$:
      $t = \mathsf{AES}(key, id \| 1 \| j \| i) \bmod b - i + 1$
      swap blocks $b - i$ and $t$ in $\mathsf{IMG}$

    add $\mathsf{IMG}$ to $K$

  return $K$
```

Fig. 7. Pseudorandom method for creating overlay images with permuted blocks for an image with identifier *id*. # of blocks per overlay image is b. Set of candidate overlay images is S.

connected layers (more details of the CNNs are in Appendix C). Recall that, to prevent the target CNN from reducing the impact of the perturbation by using a lower resolution version of the image [15], we train each local CNN on different image resolutions and learn a perturbation over them. To implement a resolution reduction or a thumbnail function, we used an additional convolutional layer size of 2×2 with values of $\frac{1}{2\times2} = 0.25$ to resize input images before rendering them to CNNs. For example, by setting up the stride value to 2×2 we can have a thumbnail function that reduces the size of the image to $\frac{1}{2\times2} = 0.25$ of original size (see Figure 19 in Appendix A).

We used the FaceScrub [51] dataset, an image dataset of 530 celebrities (classes) and $\approx 100K$ images for our evaluations. We selected 10 classes, uniformly at random, from among 230 classes of FaceScrub that have at least 150 images in the training dataset. The use of only 10 classes for training local CNNs is keeping inline with our assumption of limited end user access to data and computational capacity. To detect celebrity faces and crop them, we used DeepFace face detector CNN [57]. For each of the 10 celebrities (classes), we randomly selected $150 - 200$ images from the training dataset and learned three local CNNs, two at 224×224 pixel resolution and one at 112×122 pixel resolution. We then chose 200 (N) images *in total* over these 10 celebrities (classes) from the training dataset and learned a universal ensemble (UEP) perturbation over these images across the three local CNNs that we trained. As shown in Figure 3, to find the best value for c parameter (see Equation 3), which controls the perturbation amount, we used binary search in the range of $[0.001, 100000]$ and set $p = 0.7$.

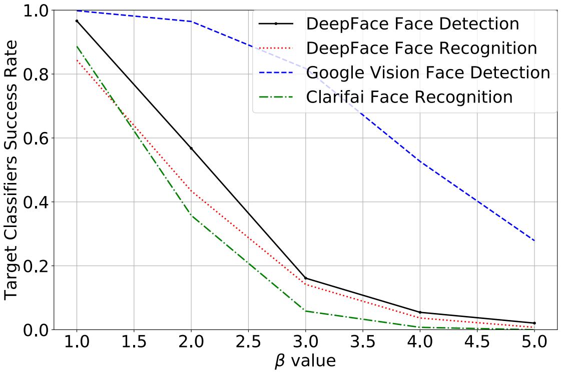


Fig. 8. UEP performance on Google Vision API face detection model, and DeepFace face detector and recognition models.

UEP Effectiveness: We evaluate UEP against `Google Vision API`'s face detection model, and face detection and recognition models of `clarifai.com` and DeepFace [57]. To this end, we selected more than 1000 FaceScrub images in total from all the 530 classes (not just the 10 classes on which local CNNs were trained). Then we cropped (Note: all adversarial perturbation methods learn on a specific image size that depends on the CNNs' input size.) faces with DeepFace face detector model [51] and then applied UEP generated perturbation. Moreover, we can control the transferability of UEP perturbation by using different weighting (β) parameter (see equation 4). We evaluate our method for different values of β parameter.

UEP vs. Google Vision API: Google Vision API was able to detect 98.8% of faces in cropped benign images. As shown in Figure 8 (dashed blue line), for a small value of β, UEP perturbation exhibited low transferability (*i.e.,* success rate) against Google Vision API. However, for larger β values, Google Vision API's face detection rate decreases significantly ($\leq 30\%$ at $\beta = 5$). While larger β values can fool face detection models, they also decrease recognizability for humans. Figure 6 shows the recognizability of the images for different values of β. Thus, there is a trade-off between fooling success rate and human recognizability. Note that, this is a face detection model and not a recognition model.

UEP vs. Clarifai.com: Clarifai.com's face detection could detect more than 99% of faces and recognize 87.25% faces of unmodified images. Clarifai.com was able to detect 98% of perturbed faces with UEP perturbation even with $\beta = 5$. However, its face recognition was able to recognize fewer than 6% of faces for $\beta = 3$ (green dash-and-dot line in Figure 8). In other words, face recognition model is more sensitive to perturbations. While small values of β are not good enough to thwart recognition by unknown CNNs, with $\beta = 2$, face recognition model of clarifai.com was able to recognize only 37% of faces and this comes down to less than 6% for $\beta = 3$. As seen in Figure 6, perturbed faces still remain recognizable for $\beta = 3$.

UEP vs. DeepFace Models: We also evaluated UEP against DeepFace face detection and recognition models. We sampled more than 1000 images of celebrities from FaceScrub dataset that are known to DeepFace face recognition model (about 186 classes). This model was able to detect all faces in unperturbed images and recognize 97.28% of them. We then tested DeepFace with perturbed images with different levels of perturbation (β values). As shown in Figure 8, the face detection and face recognition models have low accuracy, less than 20% (for both detection and recognition) for larger values of $\beta (\geq 3)$.

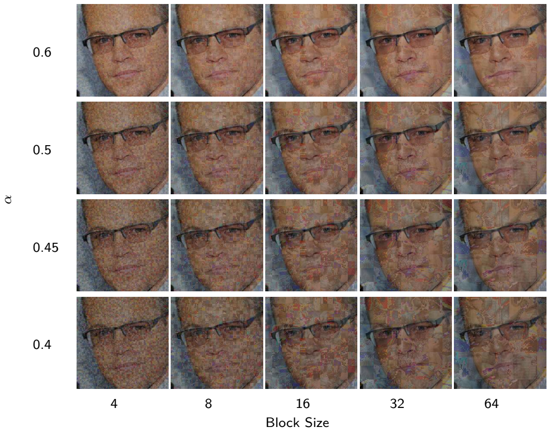


Fig. 9. Different α values vs. size of block for $k = 3$ overlay images. Overlay images were chosen randomly but the same overlay images were used in all cases.

Recoverability/Reversibility: UEP does not add the perturbation directly to pixels but adds it in the arctangent hyperbolic space (see equation 4). When mapping back to the pixel space (integers values in $[0, 255]$), we may need to round the pixel's value. However, this loss is not perceptually recognizable as measured by the structural similarity index (SSIM) [82] between the original and recovered images. SSIM is a popular measure used to compare perceptual image similarity. For two identical images SSIM value will be 1. Our results show that the average SSIM measured over the 1000 original and recovered images is 0.99 ($SD = 0.0038$). A distribution of SSIM values for UEP recovered images is shown in Figure 13.

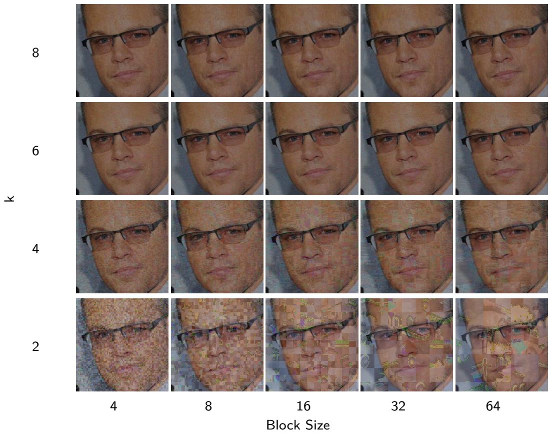


Fig. 10. Different number of overlay images vs. different number of block sizes for $\alpha = 0.6$. Used same overlay images per row.

UEP Computational Cost: Our universal transferable perturbation is learned using only 200 randomly selected images but can be applied to any other image. Compared to previous approaches [42, 48], our hybrid approach requires significantly less computational effort and time and achieves better results. But even training small CNNs is computationally expensive operation. For instance, training three small local CNNs and learning a universal transferable perturbation using 200 images took 13 hours on a standard desktop with 64GB RAM and an Intel Xeon E5-1630v4 3.70GHz quad-core CPU, and a GTX 1080 Ti GPU with 11GB RAM (Appendix C has network details). Training the 3 local CNNs took less than 1 hour and learning the universal transferable perturbation over 200 images took about 12 hours. However, this is a one-time computation cost that can be expended offline. The learned perturbation can be used for perturbing multiple images and hence the learning cost is amortized over all the images.

Since a single perturbation can be applied for a batch of images, the cost for storing the learned perturbations in order to recover the original image is only a small fraction of the storage required for the images. For example, in our evaluation we created a universal perturbation for 200 images and hence the additional cost for local storage is one perturbation ($1/200th$ of total image storage).

Summary: We showed that even with access to a small dataset, 10 classes and at most 200 images per class, one can learn small CNNs and generate a universal per-

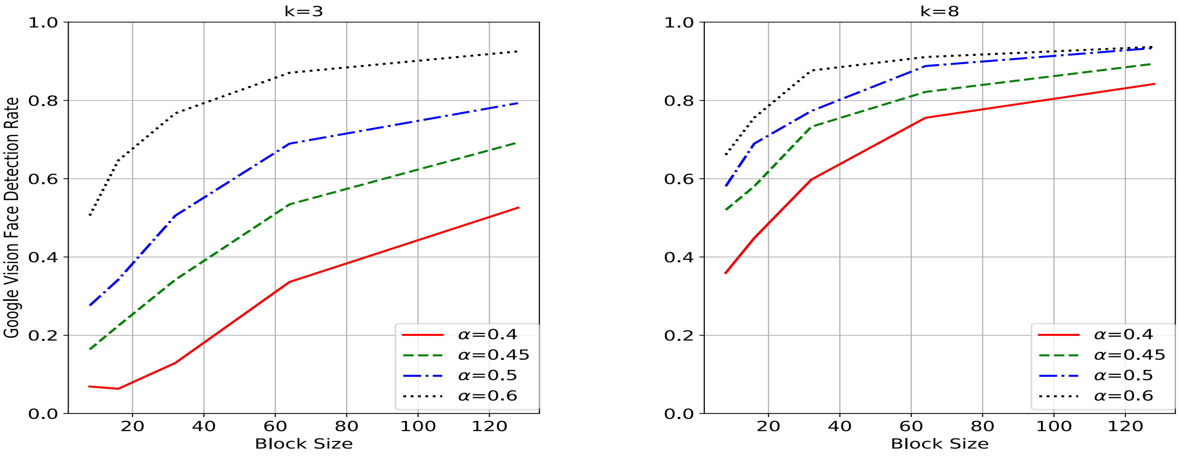


Fig. 11. Google Vision API face detection model performance on kRTIO images for k=3 and k=8.

turbation with high transferability. This is in contrast with Liu *et al.* [42] who train 4 large CNNs (with 1000 classes each) using a training dataset (ILSVRC [61]) of 1.2M images to achieve only 76% transferability (over 200 test images) on Clarifai.com. Similarly, Moosavi-Dezfooli *et al.* achieved only 70% fooling rate even after using 4000 images to learn a universal perturbation. When using only 1000 images to learn a perturbation they achieved less than 40% fooling rate (see Figure 6 in [48]). In contrast, UEP can learn an effective universal perturbation using only 200 images and achieve $\geq$ 94% fooling rate.

Our evaluation of UEP, on more than 1000 images and 3 different face detection and 2 different face recognition models, shows that fooling face detection models is harder than face recognition models. In other words, face recognition models are more sensitive to perturbations. Also, a $\beta = 3$ showed sufficient fooling capability on face recognition models while keeping the images recognizable. Our UEP method also has less loss compared to other transferable perturbation generation approaches because perturbation is added in the arctangent hyperbolic space of the images instead of directly to their pixel values.

6.2 k-RTIO Performance

k-RTIO Simulation Setup: Recall that k-RTIO requires a set S of initial overlay images. We observed empirically that the choice of this set of images did not significantly alter the fooling performance of k-RTIO. However, we felt that including images with human faces may reduce recognizability. So we chose a set of 30 random images from the Internet that did not include faces from our initial set S (this set is shown in Figure 20 in Appendix B). Similar to UEP evaluation, we selected 1000 images from the FaceScrub dataset uniformly at random and applied k-RTIO perturbations that were generated using different values for α (mixing parameter), k (number of overlay source images), and b (number of blocks in an overlay image).

Effectiveness of k-RTIO: Similar to UEP evaluation, to assess k-RTIO, we evaluated it against Google Vision API's face detection model, online face detection and recognition models of Clarifai.com, and face detection and recognition models of DeepFace.

k-RTIO vs. Google Vision API: Google Vision API detected faces in less than 40% of original unperturbed images (here we did not crop the faces but rather submitted the original images). Then we perturbed those images for which Google Vision API correctly detected faces with k-RTIO approach. Note that unlike UEP that needs to resize images to apply perturbation, k-RTIO does not need to resize the main image (just overlay images should be resized to main image's size). As shown in Figure 11, Google Vision API only can detect faces in less than 35% of images when $k = 3$ and $\alpha \leq 0.5$ for small block sizes (less than 10). Increasing α value or number of overlay images (k) leads face detection success rate to increase. This is because a larger α value implies more of the original image is present in the perturbed image and thus can increase recognizability for both humans and automated classifiers. This can be seen in Figure 9 where the image becomes clearer for higher α values for every block size. Increasing the number of overlay images used (k) leads k-RTIO perturbation to look like random noise as the different overlay images are averaged, and therefore the perturbation becomes ineffective. This is evident from Figure 10 where for k values greater than 4 images become easy to perceptually recognize for all block sizes. Interestingly, both Figures 9 and 10 show that smaller block sizes seem to produce less obfuscated images for humans while still thwarting automated classifiers.

k-RTIO vs. Clarifai.com: Clarifai.com's face detection model was able to detect faces in all original images sampled from the FaceScrub dataset. Clarifai.com's face recognition model was able to recognize $\approx 82\%$ of the celebrities in these images. To evaluate the k-RTIO performance on face recognition model, we used only the images that were recognized by Clarifai.com's face recognition model. As shown in the top row of Figure 12, similar to Google Vision API, for small values of $\alpha \leq 0.45$ and small block sizes clarifai.com's face recognition model is able to detect faces in very few images ($< 7\%$) even when face detection model is able to detect faces in more than 80% of images successfully.

k-RTIO vs. DeepFace Models: We also evaluated k-RTIO against DeepFace face detection and recognition models. As in the case of UEP, we selected more than 1000 images from FaceScrub dataset that DeepFace classified correctly. We perturbed the selected images using k-RTIO with different α values and block sizes and tested them against DeepFace. As shown in the bottom row of Figure 12, both models showed low performance for k-RTIO perturbed images with a small block size (even for larger α values).

Recoverability/Reversibility: We can always regenerate the exact overlay image that was added to original images using the secret key. However, k-RTIO still needs to round up the pixel values after adding overlay images leading to some loss. However, this loss is not perceptually recognizable. To measure the perceptual similarity, we again used SSIM to evaluate the similarity between the original and recovered images. Our results show that SSIM measure between original and recovered images averaged over the 1000 tested images is 0.96 (SD=0.07). A distribution of SSIM values k-RTIO recovered images is shown in Figure 13. k-RTIO recovered images tend to exhibit lower SSIM values as the perturbation is applied to the whole image and not just the faces as is done for UEP. For this experiment, we set the k-RTIO parameters as $k = 3$, block-size= 8 and $\alpha = 0.45$ (a sweet spot for fooling unknown face recognition models).

Computational Costs of k-RTIO: Unlike transferable perturbation generation, the *k-RTIO* method is computationally much more efficient. While it does involve $(k \times b) + k$ AES invocations for generating a unique overlay for each image, AES is computationally inexpensive especially with hardware instruction set support. For example, generating a permutation for 500 blocks takes 9×10^{-5} milliseconds and permuting an overlay image with 625 blocks of 40×40 pixels each took less 5×10^{-4} milliseconds (wall-clock time) with no special optimizations or effort. Further, creation of unique overlays can easily be parallelized when perturbing multiple images. End users need only store the secret key (128 bits) and a small image set S used in the creation of unique overlays to recover the original images. Thus, storage costs remain constant irrespective of the number of images that are perturbed.

Summary: Similar to the case of UEP, we find that fooling face detection models is harder than fooling face recognition models. As shown in Figures 11 and 12, face detection models can detect faces better for larger block sizes (*i.e.,* smaller number of blocks (b)) even at lower α values. However, k-RTIO is able to effectively thwart face recognition models. For small block sizes (4×4 or 8×8), face recognition models recognize 15% of faces at best for $\alpha \leq 0.5$. For lower alpha values, say $\alpha = 0.4$, face recognition rate does not go above 15% for any block size. Thus, the mixing parameter α and the number of blocks b are critical factors that impact face recognition. Similarly, as shown in Figures 10 and 11, number of overlay images used is also a critical factor as using more overlays leads to a final perturbation that looks like random noise and therefore ineffective.

6.3 Potential Attacks Against UEP and k-RTIO

We explore different countermeasures that can potentially be deployed against the proposed perturbation approaches. We show that using one perturbation for several images provides adversaries with an opportunity to estimate the perturbation and recover classifiable images (if not original images). This is a downside of using a single semantic perturbation for multiple images and led to the k-RTIO approach which uses a unique perturbation per image. We also explore a countermeasure where the target CNNs learns on perturbed images.

Perturbation Estimation and Removal: Let us assume that the perturbation is added directly to images ($x'_i = x_i + \delta$) and an adversary wants to find δ given only the perturbed images. Adversaries can treat δ as the main signal and the images as noise added to the main signal. In this case, the noise signal is not independent or zero mean (but rather images of faces). However, target CNNs do not need to recover the exact pertur-

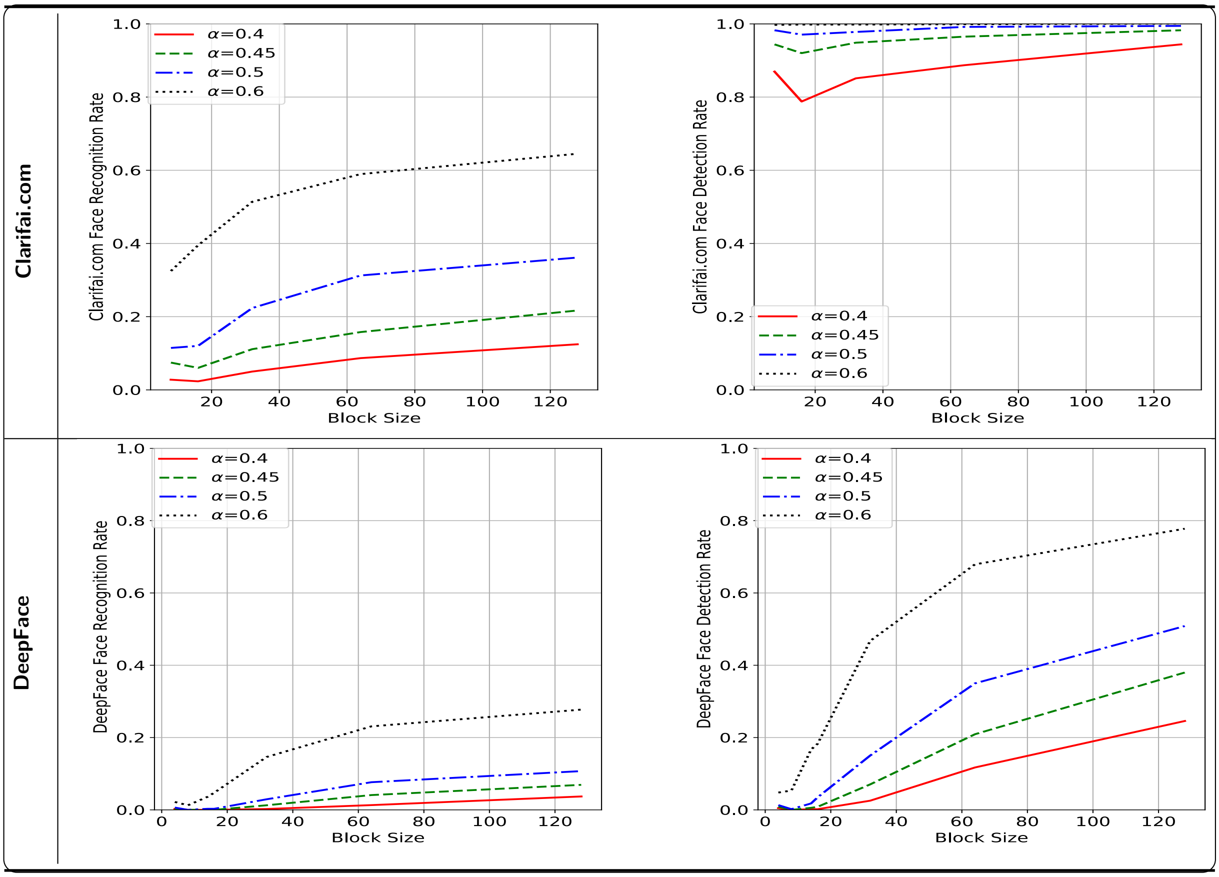


Fig. 12. Clarifai.com (online) and DeepFace (offline) face recognition and detection models on k-RTIO perturbed images for $k = 3$.

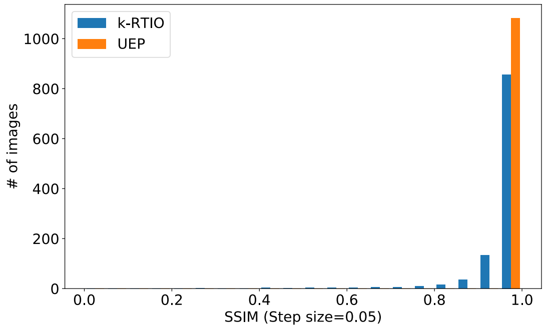


Fig. 13. SSIM histogram for recovered UEP and k-RTIO images. In contrast to k-RTIO, UEP only perturbs faces in an image, therefore it has larger SSIM on average.

bation as they do not have to recover the exact original image. They only need to recover enough of a likeness to the original image to be able to classify the image correctly. To get a rough estimate of the perturbation, a target CNNs can compute the median of pixels in all perturbed images to approximate the δ. It can then add the inverted median to a perturbed image as a new layer ($x = \lambda \times x'_i + (1 - \lambda) \times inv(\delta')$).

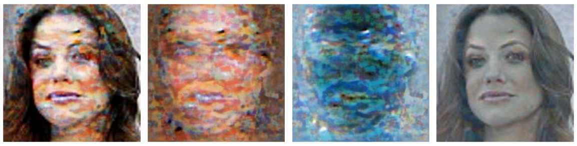


Fig. 14. UEP perturbation estimation and removal. The first image is the perturbed image, the second one is median of 200 perturbed images. The third image is inverted median and the final image is the recovered image by an adversary ($\lambda = 0.42$).

As shown in Figure 14, using this approach a target CNN can obtain an image close to the original and that seems to be sufficient for recognition. We have assessed 200 images after removing the perturbation with median estimation method and submitted it to the `clarifai.com`'s celebrity face detection model. As shown in Figure 15, `clarifai.com`'s success rate for different values of λ (0.40 to 0.56) is as high as 70% (for $\lambda \in [0.44, 0.48]$) - more than tenfold improvement! In the worst case scenario, if we assume that the target CNN

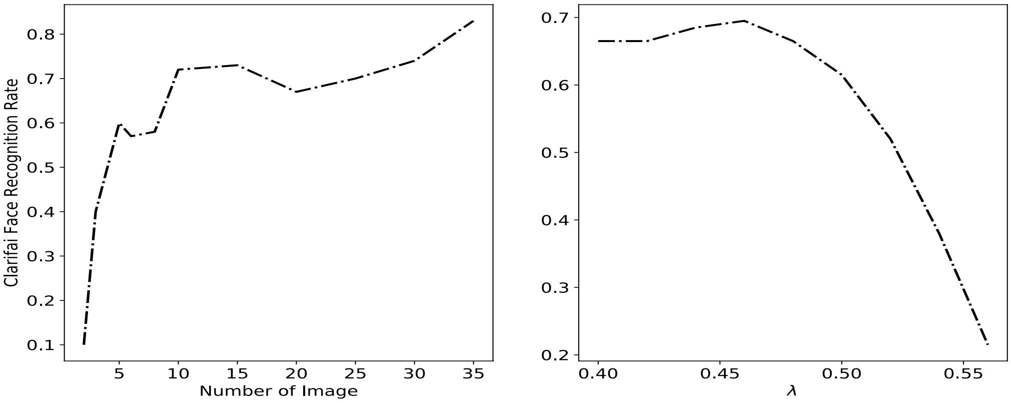


Fig. 15. Target CNN can recover 60% of the images by using Perturbation Estimation and Removal method just with 5 images using the same perturbation. By setting λ value to 0.48, an adversary can improves its success rate to 70%.

can know when it classified correctly then it can vary λ for every image and reach an accuracy as high as 80.5% with access to only 35 images modified using the same perturbation. Thus, using a single UEP perturbation for several images may allow target classifiers to estimate the perturbation and recover classifiable images. As shown in the Figure 15, an automated classifier can improve their classification accuracy to 70% by estimating perturbation using as few as 10 perturbed images.

Unlike UEP that uses a single perturbation for several images, k-RTIO generates a unique perturbation for each image, hence this perturbation estimation and removal approach is not effective against k-RTIO.

Robust CNNs: Many methods have been proposed to detect and reject adversarial examples (*e.g.,* [3, 4, 47, 67, 80]). These methods are suitable for safety sensitive application such as disease recognition and autonomous cars in which making wrong decision may cause huge consequences. However, such approaches are not useful as defenses against UEP or k-RTIO as those methods want to detect and reject perturbed images as much as possible which aligns with our goal of preventing automated classifiers from correct classification. Other proposed methods tend to make learning white-box attacks harder for an adversary [5, 56]. However, such defenses are still vulnerable to black-box attacks in which an adversary learns highly transferable perturbations on using local CNNs. Another approach is to train robust CNNs that are able to classify adversarial images correctly [43] by re-training the network on both benign and adversarial images. However, it has been shown that this approach does not make CNNs robust against all kinds of adversarial examples even for small CNNs [64]. Here we aim to assess the *classification accuracy improvement* on k-RTIO perturbed images when one trains a CNN both on benign and k-RTIO perturbed images.

For this, we selected TinyImageNet [39] with 200 classes, 100000 training-set images, and 10000 test-set images as the target CNN. We trained an Xception Convolutional Neural Network [14] on this dataset. In 100 training epochs, this CNN achieved 65.42% accuracy on the test-set images. We then added 100000 and 10000 k-RTIO perturbed images to training and test respectively, and retrained the Xception CNN. We used the same learning rate and same random seed number to initialize the CNN weights. The robust Xception CNN achieved 68.50% accuracy on benign test-set images and 32.1% accuracy on average on k-RTIO images ($\alpha \in \{0.45, 0.5\}, block - size \in \{4, 8, 16, 64\}$).

As shown in Figure 16, learning on k-RTIO images can improve accuracy. However, the accuracy on images with low alpha values $\alpha = \{0.4, 0.45\}$ was still less than 29% and 39% respectively. In other words, robust CNNs can improve their accuracy on k-RTIO perturbed images but not significantly for small α values. Note that we used the same overlay set for perturbing images in the training and test sets to evaluate the robust CNN. This simulates a situation when the target CNN has access to a user's overlay set S.

Noise Reduction Techniques: Adversarial training showed good performance against adversarial examples for small CNNs, but this approach has been shown to be computationally expensive at large scale [5] and not a desirable solution for dealing with adversarial samples.

Approaches to recover classifiable images from perturbed ones using different filtering techniques have been explored [15]. Note that CNNs do not need to recover original images to classify them correctly. They only need to reduce the impact of the perturbation to

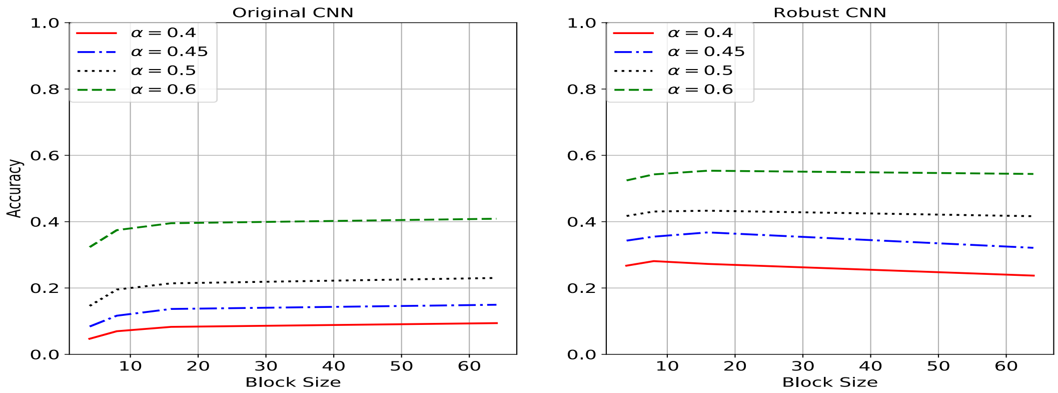


Fig. 16. The accuracy of the CNN of TinyImageNet on perturbed images by k-RTIO images for $k = 3$ (left) for the original CNN learned on benign images, and (right) robust CNN which is learned on the k-RTIO images.

obtain a correct classification. Here, we explore filtering approaches that can reduce perturbation noise sufficiently to make them recognizable for CNNs. To evaluate robustness of k-RTIO, we applied different filters such as Blurring [60], Median Filter [30], Grayscale Filter and Gaussian Filter to reduce noise or distortion. As shown in Figure 17, none of these methods could improve image quality for DeepFace face recognition classifier significantly.

7 Discussion and Future Work

Adversarial perturbation is being studied as a way to address privacy concerns with scalable automated face recognition by CNNs [20, 31, 63]. In this work, we proposed some practical requirements for such defenses and proposed two perturbation schemes, UEP and k-RTIO, that meet those requirements. However, these schemes have their limitations.

Downside of Universal Perturbation: UEP generates a single transferable perturbation that can be used on multiple image to successfully fool an unknown CNN. The rationale for designing UEP, a universal transferable perturbation scheme, was to amortize the cost of learning a perturbation across multiple images, and to be able to reverse the perturbation with a low storage cost of just storing one perturbation. However, the use of a single perturbation across multiple images opens up the possibility of a target CNN estimating or learning this perturbation by observing multiple perturbed images. Indeed, as described in Section 6.3, we show that it is possible to filter the perturbation sufficiently enough to allow correct classification. This leads us to observe that cost-efficient approaches to learn and store image-specific perturbations need to be developed before *learning-based* perturbation approaches can become practical defense tools against automated classification. Alternatively, if multiple independent universal perturbations can be computed and used, it may be possible to leverage them to thwart the perturbation estimation approach discussed in Section 6.3. Both these remain open problems.

Efficiency of Semantic Perturbations: In contrast to adversarial-learning approaches such as UEP, our semantic approach *k-RTIO* reduces the efficiency of a CNN in recognizing features, by reducing the similarity of adjacent pixels. *k-RTIO* is as a better choice than UEP, given that k-RTIO is both computationally cheaper and found to be more effective. We are encouraged by the feasibility to find cheaper techniques to thwart recognition by existing classifiers. While our initial evaluation shows *k-RTIO* is robust against common filtering techniques, it is not an exhaustive list and further work is needed in this direction.

Re-training a CNN on adversarial examples has been shown to improve the CNN's robustness against such examples [43] and is a potential defense against k-RTIO. Although this approach is not computationally feasible for large scale datasets [5], we did evaluate its effectiveness against k-RTIO using TinyImagenet dataset to train a CNN on both clean and k-RTIO perturbed images. Our evaluation showed that for small values of α, this robust CNN's accuracy on perturbed images has not improved significantly ($< 30\%$ and 40% for $\alpha = 0.4$ and 0.45, respectively).

Recognizability: Our findings indicate a sweet spot for k-RTIO parameters ($k = 3$, $\alpha = 0.45$, and $b < 20$) where classifiers have low success while the images remain visually recognizable (see Figure 9). However, it

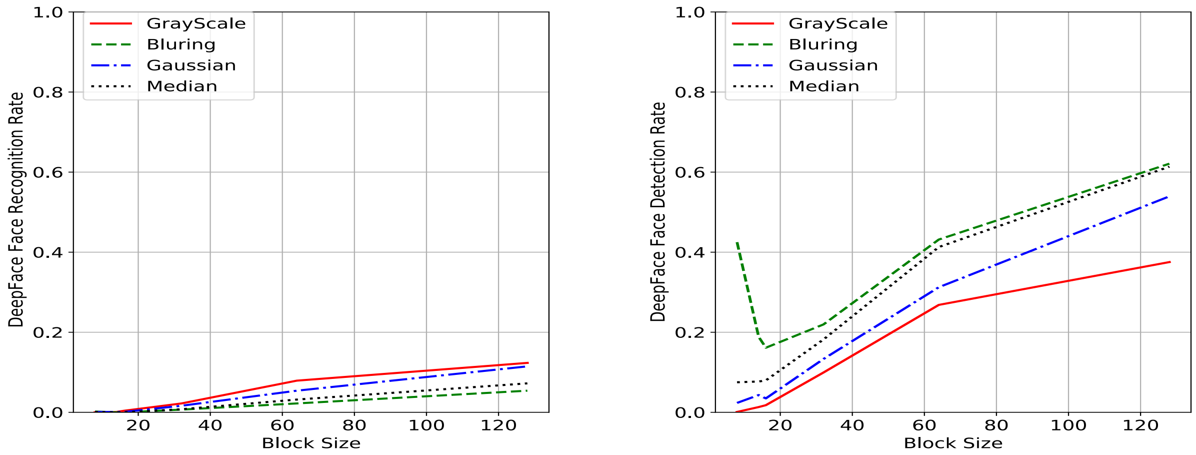


Fig. 17. Filtering results on DeepFace face detection and celebrity recognition models for $\alpha = 0.45$ and $k = 3$.

would be good to empirically evaluate the recognizability of *k-RTIO* perturbed images through user studies and to characterize the boundaries for human recognition in terms of parameter values for k, α, and b.

Lack of Strong Guarantees: Besides learning a robust CNN, online service providers may also be able to rely on other channels of information other than the image itself to infer information that can help them classify correctly, such as where, when, and with whom the image was shared. Further, an adversary may be able to use the structure present in overlay images to reduce their effort required to recreate the unique overlay image. While this can be done using visual clues manually, it may also be feasible to automate this although we are not aware of a specific way to efficiently achieve that. Similarly, it may be possible to find a smoothing filter that allows an adversary to correctly classify a k-RTIO image without reversing the perturbation. In other words, unlike traditional privacy mechanisms like encryption, perturbation-based defenses are not yet able to provide provable guarantees. This is especially the case as it is hard to quantify the information leaked by perturbed images (especially since we require it to be sufficient enough for human recognition). At least in the case of ideal TPE schemes [74] the leaked bits can be succinctly characterized. While differential-privacy based perturbation approaches are emerging (*e.g.,* [17, 18]) they are currently not reversible. Perturbation schemes that thwart one pre-specified classifier while allowing another pre-specified classifier to classify correctly (*e.g.,* [31, 79]) have also been proposed. However, those approaches are also not reversible. Further, it is not clear how they fare against an unknown classifier. This lack of defined guarantees is not specific to just UEP or k-RTIO but seems to be true for adversarial perturbation based defenses in general. This is an open question that needs to be explored for image perturbation based privacy defenses to become practical.

Finally, while problems that are hard for AI and easier for humans have been explored in the context of distinguishing between humans and robots on the web (*e.g.,* CAPTCHAs), there seems to be a need to revisit such a paradigm for privacy against automated classifiers.

8 Conclusion

Given that users tend to store and share photos with friends and family using cloud-based platforms with weak access control options, the emergence of scalable and automated image classification is a serious threat to user privacy. In this work we explored two reversible image perturbation techniques that can thwart classification and tracking by automated classifiers while at the same time allowing recognition by humans. We showed that our k-Randomized Transparent Image Overlays (k-RTIO) approach is not only very effective but also computationally cheaper in contrast to learning-based perturbation methods. However, more work is needed before perturbation based approaches can truly become practical.

Acknowledgements. The authors would like to thank the anonymous reviewers and our shepherd for their constructive feedback. This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

References

[1] An app that encrypts your photos from camera to cloud. https://www.wired.com/story/pixek-app-encrypts-photos-from-camera-to-cloud/. Accessed: 2019-11-30.

[2] Planet selfie. https://www.businessinsider.com/were-now-posting-a-staggering-18-billion-photos-to-social-media-every-day-2014-5. Accessed: 2019-11-30.

[3] M. Abbasi, A. Rajabi, C. Gagné, and R. B. Bobba. Towards Dependable Deep Convolutional Neural Networks (CNNs) with Out-distribution Learning. *Workshop on Dependable and Secure Machine Learning*, 2018.

[4] N. Akhtar and A. Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access*, 6:14410–14430, 2018.

[5] A. Athalye, N. Carlini, and D. A. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In J. G. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 274–283. PMLR, 2018.

[6] S. Baluja and I. Fischer. Adversarial transformation networks: Learning to generate adversarial examples. *arXiv preprint arXiv:1703.09387*, 2017.

[7] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 417–424, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[8] D. Bitouk, N. Kumar, S. Dhillon, P. Belhumeur, and S. K. Nayar. Face swapping: Automatically replacing faces in photographs. *ACM Trans. Graph.*, 27(3):39:1–39:8, Aug. 2008.

[9] C. Bourez. *Course 2: build deep learning neural networks in 5 days only*, 2018. http://christopher5106.github.io/deep/learning/2018/10/20/course-two-build-deep-learning-networks.html".

[10] J. F. Cardoso. Blind signal separation: statistical principles. *Proceedings of the IEEE*, 86(10):2009–2025, Oct 1998.

[11] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pages 39–57. IEEE, 2017.

[12] F. Carmo, J. Assis, V. Estrela, and A. Coelho. Blind signal separation and identification of mixtures of images. pages 337 – 342, 12 2009.

[13] K. Chinomi, N. Nitta, Y. Ito, and N. Babaguchi. Prisurv: Privacy protected video surveillance system using adaptive visual abstraction. In *Proceedings of the 14th International Conference on Advances in Multimedia Modeling*, MMM'08, pages 144–154, Berlin, Heidelberg, 2008. Springer-Verlag.

[14] F. Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

[15] N. Das, M. Shanbhogue, S.-T. Chen, F. Hohman, S. Li, L. Chen, M. E. Kounavis, and D. H. Chau. Shield: Fast, practical defense and vaccination for deep learning using jpeg compression. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 196–204. ACM, 2018.

[16] B. Driessen and M. Dürmuth. *Achieving Anonymity against Major Face Recognition Algorithms*, pages 18–33. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[17] L. Fan. Image pixelization with differential privacy. In *Data and Applications Security and Privacy XXXII - 32nd Annual IFIP WG 11.3 Conference, DBSec 2018, Bergamo, Italy, July 16-18, 2018, Proceedings*, pages 148–162, 2018.

[18] L. Fan. Practical image obfuscation with provable privacy. In *Proceedings of IEEE International Conference on Multimedia and Expo (ICME), 2019*, 2019.

[19] R. A. Fisher, F. Yates, et al. Statistical tables for biological, agricultural and medical research. *Statistical tables for biological, agricultural and medical research.*, (6th ed), 1963.

[20] C. Gao, V. Chandrasekaran, K. Fawaz, and S. Jha. Face-off: Adversarial face obfuscation. *arXiv preprint arXiv:2003.08861*, 2020.

[21] E.-J. Goh, H. Shacham, N. Modadugu, and D. Boneh. Sirius: Securing remote untrusted storage. In *NDSS*, volume 3, pages 131–145, 2003.

[22] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.

[23] I. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[24] R. Gross, E. Airoldi, B. Malin, and L. Sweeney. Integrating utility into face de-identification. In *Proceedings of the 5th International Conference on Privacy Enhancing Technologies*, PET'05, pages 227–242, Berlin, Heidelberg, 2006. Springer-Verlag.

[25] R. Gross, L. Sweeney, J. Cohn, F. Torre, and S. Baker. Face de-identification. *Protecting Privacy in Video Surveillance*, pages 129–146, 2009.

[26] K. Hill. *The Secretive Company That Might End Privacy as We Know It*, 1/182020.

[27] S. Hill, Z. Zhou, L. Saul, and H. Shacham. On the (in)effectiveness of mosaicing and blurring as tools for document redaction. *Proc. Privacy Enhancing Technologies*, 2016(4):403–17, Oct. 2016.

[28] H. Hosseini and R. Poovendran. Semantic adversarial examples. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2018.

[29] H. Hosseini, B. Xiao, A. Clark, and R. Poovendran. Attacking automatic video analysis algorithms: A case study of google cloud video intelligence API. In R. A. Hallman, K. Rohloff, and V. I. Chang, editors, *Proceedings of the 2017 on Multimedia Privacy and Security, MPS@CCS 2017, Dallas, TX, USA, October 30, 2017*, pages 21–32. ACM, 2017.

[30] T. Huang, G. Yang, and G. Tang. A fast two-dimensional median filtering algorithm. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 27(1):13–18, 1979.

[31] S. Joon Oh, M. Fritz, and B. Schiele. Adversarial image perturbation for privacy protection – a game theory perspective. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[32] A. Jourabloo, X. Yin, and X. Liu. Attribute preserved face de-identification. In *2015 International Conference on Biometrics (ICB)*, pages 278–285, May 2015.

[33] M. Kallahalla, E. Riedel, R. Swaminathan, Q. Wang, and K. Fu. Plutus: Scalable secure file sharing on untrusted storage. In *Fast*, volume 3, pages 29–42, 2003.

[34] A. C. Kokaram, R. D. Morris, W. J. Fitzgerald, and P. J. W. Rayner. Interpolation of missing data in image sequences. *IEEE Transactions on Image Processing*, 4(11):1509–1519, Nov 1995.

[35] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[36] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial examples in the physical world. *Workshop Track of the International Conference on Learning Representations (ICLR)*, 2017.

[37] K. Lander, V. Bruce, and H. Hill. Evaluating the effectiveness of pixelation and blurring on masking the identity of familiar faces. *Applied Cognitive Psychology*, 15(1):101–116, 2001.

[38] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back. Face recognition: A convolutional neural-network approach. *IEEE transactions on neural networks*, 8(1):98–113, 1997.

[39] Y. Le and X. Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 2015.

[40] T. Li and L. Lin. Anonymousnet: Natural face deidentification with measurable privacy. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.

[41] Y. Lin, S. Wang, Q. Lin, and F. Tang. Face swapping under large pose variations: A 3d model based approach. In *2012 IEEE International Conference on Multimedia and Expo*, pages 333–338, July 2012.

[42] Y. Liu, X. Chen, C. Liu, and D. Song. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*, 2016.

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

[44] B. Marohn, C. V. Wright, W.-c. Feng, M. Rosulek, and R. B. Bobba. Approximate thumbnail preserving encryption. 2017.

[45] M. Mathias, R. Benenson, M. Pedersoli, and L. Van Gool. Face detection without bells and whistles. In *European Conference on Computer Vision*, pages 720–735. Springer, 2014.

[46] R. McPherson, R. Shokri, and V. Shmatikov. Defeating image obfuscation with deep learning. *arXiv preprint arXiv:1609.00408*, 2016.

[47] D. Meng and H. Chen. Magnet: a two-pronged defense against adversarial examples. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 135–147. ACM, 2017.

[48] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard. Universal Adversarial Perturbations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1765–1773, 2017.

[49] S. M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. Deepfool: A simple and accurate method to fool deep neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2574–2582, June 2016.

[50] E. M. Newton, L. Sweeney, and B. Malin. Preserving privacy by de-identifying face images. *IEEE transactions on Knowledge and Data Engineering*, 17(2):232–243, 2005.

[51] H.-W. Ng and S. Winkler. A data-driven approach to cleaning large face datasets. In *2014 IEEE international conference on image processing (ICIP)*, pages 343–347. IEEE, 2014.

[52] G. Oded. *Foundations of Cryptography: Volume 2, Basic Applications*. Cambridge University Press, USA, 1st edition, 2009.

[53] S. J. Oh, R. Benenson, M. Fritz, and B. Schiele. *Faceless Person Recognition: Privacy Implications in Social Media*, pages 19–35. Springer International Publishing, Cham, 2016.

[54] T. Orekondy, B. Schiele, and M. Fritz. Towards a visual privacy advisor: Understanding and predicting privacy risks in images. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 3706–3715. IEEE Computer Society, 2017.

[55] A. B. Pande and S. Prabha. Image blind signal separation algorithm based on fast ica. In *IJCA Proceedings on International Conference on Advances in Communication and Computing Technologies 2012*, number 3, pages 21–24. Citeseer, 2012.

[56] 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*, pages 506–519. ACM, 2017.

[57] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. In *British Machine Vision Conference*, 2015.

[58] M.-R. Ra, R. Govindan, and A. Ortega. P3: Toward privacy-preserving photo sharing. In *Presented as part of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*, pages 515–528, Lombard, IL, 2013. USENIX.

[59] A. RachelAbraham, A. Kethsy Prabhavathy, and J. Devi Shree. A Survey on Video Inpainting. *International Journal of Computer Applications*, 56(9):43–47, Oct. 2012.

[60] E. Reinhard, W. Heidrich, P. Debevec, S. Pattanaik, G. Ward, and K. Myszkowski. *High dynamic range imaging: acquisition, display, and image-based lighting*. Morgan Kaufmann, 2010.

[61] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[62] S. Sengupta and K. J. O'Brien. Facebook can id faces, but using them grows tricky. *The New York Times*, September 2012.

[63] S. Shan, E. Wenger, J. Zhang, H. Li, H. Zheng, and B. Y. Zhao. Fawkes: Protecting personal privacy against unauthorized deep learning models. *arXiv preprint arXiv:2002.08327*, 2020.

[64] Y. Sharma and P.-Y. Chen. Attacking the madry defense model with l_1-based adversarial examples. *arXiv preprint arXiv:1710.10733*, 2017.

[65] Y. Shoshitaishvili, C. Kruegel, and G. Vigna. Portrait of a privacy invasion. *Proceedings on Privacy Enhancing Technologies*, 2015(1):41–60, 2015.

[66] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *International Conference on Learning Representations(ICLR)*, 2015.

[67] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[68] Z. Stone, T. Zickler, and T. Darrell. Autotagging facebook: Social network context improves photo annotation. In *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–8, June 2008.

[69] Q. Sun, A. Tewari, W. Xu, M. Fritz, C. Theobalt, and B. Schiele. A hybrid model for identity obfuscation by face replacement. In V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, editors, *Computer Vision – ECCV 2018*, pages 570–586, Cham, 2018. Springer International Publishing.

[70] Y. Sun, X. Wang, and X. Tang. Deep convolutional network cascade for facial point detection. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 3476–3483. IEEE, 2013.

[71] Z. Sun, L. Meng, and A. Ariyaeeinia. Distinguishable de-identified faces. In *Automatic Face and Gesture Recognition (FG), 2015 11th IEEE International Conference and Workshops on*, volume 04, pages 1–6, May 2015.

[72] L. Sweeney. K-anonymity: A model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10(5):557–570, Oct. 2002.

[73] 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*, 2014.

[74] K. Tajik, A. Gunasekaran, R. Dutta, B. Ellis, R. B. Bobba, M. Rosulek, C. V. Wright, and W. Feng. Balancing image privacy and usability with thumbnail-preserving encryption. In *26th Annual Network and Distributed System Security Symposium, NDSS 2019, San Diego, California, USA, February 24-27, 2019*, 2019.

[75] S. Tansuriyavong and S.-i. Hanaki. Privacy protection by concealing persons in circumstantial video image. In *Proceedings of the 2001 Workshop on Perceptive User Interfaces*, PUI '01, pages 1–4, New York, NY, USA, 2001. ACM.

[76] M. Tierney, I. Spiro, C. Bregler, and L. Subramanian. Cryptagram: Photo privacy for online social media. In *Proceedings of the First ACM Conference on Online Social Networks*, COSN '13, pages 75–88, New York, NY, USA, 2013. ACM.

[77] L. Wolf, T. Hassner, and I. Maoz. Face recognition in unconstrained videos with matched background similarity. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 529–534. IEEE, 2011.

[78] C. V. Wright, W.-c. Feng, and F. Liu. Thumbnail-preserving encryption for jpeg. In *Proceedings of the 3rd ACM Workshop on Information Hiding and Multimedia Security*, IH&#38;MMSec '15, pages 141–146, New York, NY, USA, 2015. ACM.

[79] Z. Wu, Z. Wang, Z. Wang, and H. Jin. Towards privacy-preserving visual recognition via adversarial training: A pilot study. In V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, editors, *Computer Vision – ECCV 2018*, pages 627–645, Cham, 2018. Springer International Publishing.

[80] W. Xu, D. Evans, and Y. Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. *Network and Distributed System Security Symposium*, 2018.

[81] Q. A. Zhao and J. T. Stasko. The awareness-privacy tradeoff in video supported informal awareness: A study of image-filtering based techniques. Technical report, Goergia Tech, http://hdl.handle.net/1853/3452, 1998.

[82] Zhou Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

A Adversarial Attacks

Here, we introduce the well-known methods for generating adversarial examples for a given image for a known CNNs.

Fast Gradient Sign [23]: This method aims to minimize the maximum changes in each pixel for a target attack. More precisely, it tends to minimize L_∞. Therefore, it uses the sign of gradient of loss function as follows:

$$x_{new} = x - \epsilon.sign(\nabla J(F(x,\theta), y')) \tag{6}$$

In that, y' is the target label and ϵ is a constant small coefficient which controls the maximum changes of pixels. This method is fast but not efficient. Also, in [36], an iterative variants of FGS, called IFGS, has been proposed to increase its efficiency.

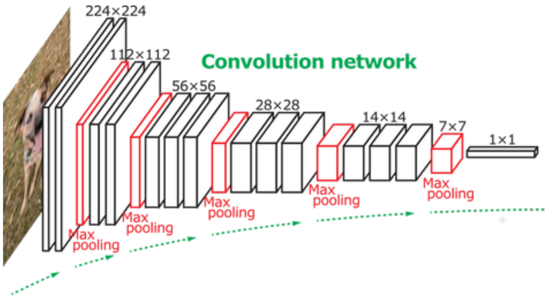


Fig. 18. VGG-16 Convolutional Neural Network [9].

L-BFGS [73]: This method is a fast method to find a targeted or untargeted attack. This method minimizes following equation by L-BFGS method.

$$\min_{\delta} c.\|\delta\|_2 + J(F(x,\theta)), \;\; x + \delta \in [0,1]^d \tag{7}$$

where c is constant coefficient which leads δ has small value if it has the large value. This parameter controls the amount of changes in a given image.

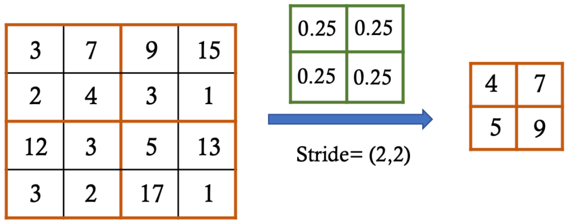


Fig. 19. Using a convolutional layer with stride value of $(2, 2)$ and kernel values of 0.25, one can implement a thumbnail function to reduce the size of an image to 25% of original size.

Deepfooling [49]: Unlike the fast gradient sign method, this method is for untargeted attack. This method is a fast and efficient method to find a small perturbation for a given image. Similar to L-BFGS, it minimizes $||\delta||_2$. This method is based on an iterative linearization of the loss function.

Table 1. Table of Notations.

Notation	Description
CNN	Convolutional Neural Network
UEP	Universal Ensemble Perturbation
k-RTIO	k Randomized Transparent Image Overlays
White-Box Attack	Learning adversarial perturbation on known and accessible CNNs
Black-Box Attack	Learning adversarial perturbation locally for unknown CNNs
Universal perturbation	A single perturbation which works on several images
Transferability	Ability of a perturbation to fool unknown CNNs
BSS	Blind Signal Separation
$F(x, \theta)$	A function representing a CNN that returns a probability vector. x is the input image and θ are weights of the CNN.
$Y(x)$	Probability vector returned by a CNN $(F(x, \theta) = Y(x),\ \sum_i Y(x)_i = 1)$
$J(.)$	Loss function ($e.g.,$ entropy)
δ	Adversarial perturbation
α	Mixing parameter in kRTIO
k	Number of overlay images in k-RTIO
b	Number of blocks in k-RTIO
κ	Misclassification confidence: minimum difference between confidence of the target class and the true class.
β	UEP perturbation weighting factor
y^*	True label of a given input
$\sigma(t)$	$\sigma(t) = 1$ if t is true otherwise $\sigma(t) = 0$
p	Universal perturbation misclassification rate
λ	Mixing parameter in perturbation and estimation removal
$inv(\delta)$	$1 - \delta$

Carlini Attack [11]: In contrast to previous methods, this method does not use loss function to find a perturbation. Authors of [11] introduced a new objective function based on minimum difference between the probabilities assigned to the true class and other classes such that the objective function has lower value if a CNN misclassifies the image with higher confidence as follows:

$$f(x) = \max(\max\{Y(x)_j : j \neq y^*\} - Y(x)_{y^*}, -\kappa) \quad (8)$$

where $Y(x)$ is the probability vector assigned by CNN to the input(x). Also, y^* and κ denote the real label of the input and confidence parameter, respectively. Larger value for confidence parameter leads the classifiers to assign the image to a new class different from the true one with higher probability such that the difference between the probability of the new class and others is κ at least. Moreover, previous methods add perturbation directly to images. Therefore, finding a perturbation that keeps the final value of images in acceptable rage $[0, 1]$ is difficult. To simplify the problem, they suggested using the alternative term of $\frac{1}{2}(\tanh(z) + 1)$ whose value is always in $[0, 1]$. More precisely, the perturbation δ is added to arc-tangent hyperbolic value of an image instead of added directly images. As one can see in equation 4, the user does not add the noise (δ) to her images directly. The variable δ can get any values from $[-\infty, \infty]$.

B Overlay Images

We used different images of objects, landscape etc. in our set S (initial set of overlay images). We did not have any constraints on selecting the images except not having human faces in there. Because using an image with faces in there may reduce the recognizability of the images by end users.

Set of overlay images used in our experiments are shown in Figure 20.

C Neural Networks

We trained small convolutional neural networks on limited faces (10 classes of faces). To minimize the network dependencies, we utilized two different CNNs structures trained on different images sizes.

VGG16:

```
model = Sequential()
```

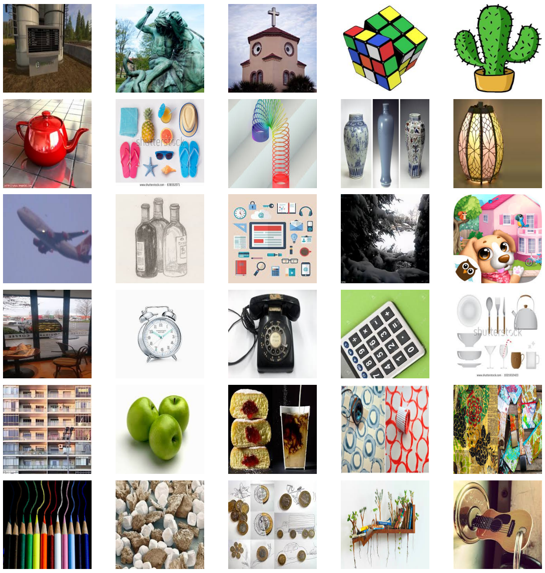


Fig. 20. Set S of the overlay images used in generating k-RTIO images

```
model.add(Conv2D(128, (5, 5),
    input_shape=shape_image))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2),
    strides=(2,2), padding='valid'))
model.add(Conv2D(256, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),
    strides=(2,2), padding='valid'))
model.add(Conv2D(512, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),
    strides=(2,2), padding='valid'))
model.add(Conv2D(512, (2, 2)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(512, (2, 2)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2),
    strides=(2,2), padding='valid'))
model.add(Flatten())
model.add(Dropout(0.5))
model.add(Dense(200))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(self.num_classes))
```

FaceScrub:

```
model = Sequential()
model.add(Conv2D(32, (3, 3),
    input_shape=shape_image))
model.add(LeakyReLU(alpha=.01))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Conv2D(64, (3, 3)))
model.add(LeakyReLU(alpha=.01))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.5))
model.add(Conv2D(128, (3, 3)))
model.add(LeakyReLU(alpha=.01))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.5))
model.add(Flatten())
model.add(Dense(200))
model.add(LeakyReLU(alpha=.01))
model.add(Dropout(0.5))
model.add(Dense(self.num_classes))
```

D Blind Signal Separation

Blind signal separation (BSS) is used to extract independent signals $X = [x_1^T, \cdots, x_n^T]$ which are mixed together by a mixing matrix $(A_{m\times n})$ from the resulting signals $B = A \times X$. The solution would be easy if the adversary knows the mixing matrix $(A_{m\times n})$. However, this matrix is unknown. The adversary can use blind signal separation method to recover the original signal only by considering the correlation between the signals B.

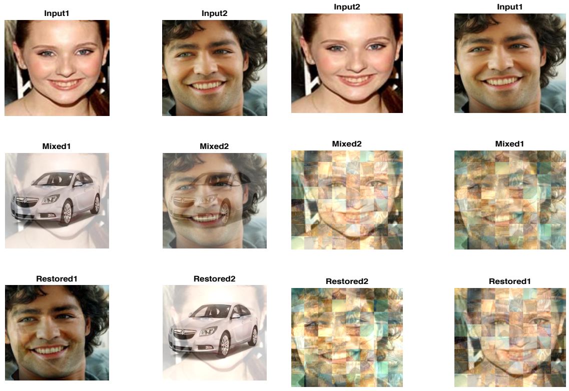
Let, $I = [I_1, \cdots, I_m]$ be a set of original images where we want to add the image I' as a new layer with the weight of α $(b_j = \alpha I_j + (1 - \alpha) I'_j)$. The user first converts the images to raw data in the form that a signal is represented by as follows:

$$X = \begin{bmatrix} I_{11} & \dots & I_{1n} \\ I_{21} & \dots & I_{2n} \\ \vdots & \vdots & \vdots \\ I_{m1} & \dots & I_{mn} \\ I'_{11} & \dots & I'_{1n} \end{bmatrix}, A = \begin{bmatrix} \alpha & 0 & 0 & \dots & 1-\alpha \\ 0 & \alpha & 0 & \dots & 1-\alpha \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & \alpha & 1-\alpha \end{bmatrix} \quad (9)$$

and then creates matrix A. The user obtains the perturbed images by multiplying matrix X by A.

The adversary only has access to the mixed signal B and wants to recover the original images.

As shown in Figure 21a, we used FastICA [12, 55] method (variant of BSS for images) to recover the I' or the images. As shown in this Figure, the adversary can easily recover the original images from the mixed images. Because this method initializes the correlation matrix randomly, the final recovered images are different. But it is possible to recover one of the pictures well. It is obvious that just by having one image, we can recover the other images easily. However, in k-RTIO we apply different overlay images for each original image. Further, we also select uniformly at random k overlay images used for any given image from a larger pool of S overlay images which makes finding blocks overlaid with same k blocks difficult. Therefore, BSS is not effective against k-RTIO.



(a) Naive TOT **(b)** k-RTIO

Fig. 21. Blind Signal Separation can recover perturbed images by transparent overlays technique. Generating overlay images by permuting the original overlay images' blocks randomly leads BSS to not be able to recover images.