Training with the Invisibles: Obfuscating Images to Share Safely for Learning Visual Recognition Models

Tae-hoon Kim†‡ Dongmin Kang∗†‡ Kari Pulli.§ Jonghyun Choi†‡
†Deeping Source Inc. ‡University of Oulu †§Gwangju Inst. of Science and Tech. (GIST)
pete.kim@deepingsource.io skydmk21@gist.ac.kr kari.pulli@oulu.fi jhc@gist.ac.kr

Abstract

High-performance visual recognition systems generally require a large collection of labeled images to train. The expensive data curation can be an obstacle for improving recognition performance. Sharing more data allows training for better models. But personal and private information in the data prevent such sharing. To promote sharing visual data for learning a recognition model, we propose to obfuscate the images so that humans are not able to recognize their detailed contents, while machines can still utilize them to train new models. We validate our approach by comprehensive experiments on three challenging visual recognition tasks: image classification, attribute classification, and facial landmark detection on several datasets including SVHN, CIFAR10, Pascal VOC 2012, CelebA, and MTFL. Our method successfully obfuscates the images from humans recognition, but a machine model trained with them performs within about 1% margin (up to 0.48%) of the performance of a model trained with the original, non-obfuscated data.

1. Introduction

Large scale datasets [5, 7] are key for the success of modern computer vision and machine learning algorithms. Once trained, the systems can be further improved using continuous streams of data [34]. If we are able to diversify the sources of the data, the performance and robustness of the system can dramatically increase. For continuous (or lifelong) learning [38], data sharing is an important tool to increase the data size by orders of magnitude [31].

However, if the data contains personal and private information, it cannot be readily shared. Before such data can be shared, the sensitive information needs to be de-identified [39]. Common techniques for de-identification include scrambling, masking, blurring, dictionary replace-

* indicates equal contribution. This work is done while D. Kang was an intern at Deeping Source Inc.

Figure 1. We learn an obfuscating network O to allow sharing data securely without exposing private information. Once the obfuscated data is shared, others can train a new model F of their own interest, e.g., for classification or landmark detection, with the obfuscated data. The trained model F can be also shared with the obfuscator O or its light-weight version O∗ (Sec. 5.8) for inference. Inferencing accuracy of F is comparable to a model trained and tested with non-obfuscated data Forig.

ment, and encryption. But when commercial solutions such as surveillance camera systems apply them conservatively, they end up excessively applying masking and blurring to any portion of a captured image that might allow identifying a person. Such anonymization methods degrade not only the quality of the embedded personal information, but also that of the original data, which may make the processed data barely useful for machine learning.
We aim to learn a model to obfuscate the private data so that it can be both shared securely and still used to train new models. We refer to the model as a “recognition-aware obfuscator” and illustrate three usage scenarios we address in this paper in Figure 1.

We use Generative Adversarial Networks (GAN) [11] to generate obfuscated images confusing to human eyes from estimated distribution of original images by setting the target distribution to be in the obfuscated space. Unlike the original GAN formulation, however, we want to generate images as unnatural as possible.

When we distribute the obfuscated images, we need to distribute the obfuscator $O$ as well, in order to handle new test images that are not in the same space of the obfuscated images. However, as the representation of $O$ can be quite large, we learn $O^p$, a light-weight approximation of $O$, and share that instead.

We empirically show that the images obfuscated by our method can be used to train new models that achieve comparable accuracy with the models trained using non-obfuscated data on various challenging datasets and tasks.

2. Related Work

Privacy-Preserving Visual Recognition. Obfuscating data has been widely studied in information security domain [24], but less so in visual recognition domain. With the emergence of the deep neural networks and growing need of large labeled data sets, collecting and securely sharing data is also drawing attention in visual recognition domain. Oh et al. [27] compared different methods for obfuscating faces to recognize the person’s identity by using contextual cues such as clothes and background. Oh et al. [28] show the effectiveness of adversarial image perturbation (AIP) for privacy protection in a game theoretic setting. Jourabloo et al. [18] replace the identity of a face for maintaining privacy while preserving the visual attributes of the faces for facial attribute estimation. Ryoo et al. [32] down-sample images for privacy-preserving action recognition. Wu et al. [40] propose a privacy-preserving action recognition method by degrading the video with a gray scale mask but maintain the action recognition accuracy. Ren et al. [30] anonymize faces while preserving the accuracy of action detection or facial attributes by seamlessly replacing the faces with other faces.

All of these methods, however, have to carefully analyze the task and identify and obfuscate features that are only indirectly related to the recognition task; e.g., obfuscating the faces or scene details in action recognition may not be directly related to the information critical to the recognition performance as the detailed silhouettes of the persons, which are arguably the important information for action recognition, are still preserved. Our more general approach modifies all the information, including information that is directly related to the recognition task, such as the edges in digit recognition or the detailed texture in object recognition, to make the identity of the image perceptually ambiguous to humans. Yet our modifications allow training a model without sacrificing the recognition performance.

Generating Adversarial Examples. Recognition models can be fooled by generating adversarial examples that apply small image changes that are almost invisible to a human observer [37, 43]. We invert this approach to obfuscate images while preserving the classification performance when used in training: we superimpose visible artifacts onto the images while preserving the recognition performance.

One of the earliest and most effective methods of generating an adversarial example is Fast Gradient Sign Method (FGSM) [12]. It uses gradients of a trained model to find the pixels to be modified to fool the model. Generalizing the directly computed gradients or solving an optimization on the image pixels, Baluja and Fischer [2] propose Adversarial Transformation Networks that transform an input image into an adversarial attack on the target network while minimizing the $L_2$ distance between the original and the perturbed image. Both methods use the gradients from the trained model, and therefore need to have such a model to generate adversarial examples. For our problem setup, this constraint hinders easy sharing of visual data (see Sec. 5.7 for more details).

Kurakin et al. [21] show that even in physical world scenarios, machine learning systems are vulnerable to adversarial examples.

Generative Adversarial Networks (GAN). Generative Adversarial Networks [11, 1] learn a generative model that allows mapping random vectors from a latent space to the distribution of the training set. In principle, our method also maps images into different images while preserving the classification performance when used in training: we superimpose visible artifacts onto the images and set the generator $O(\cdot)$ to obfuscate an input image while preserving the classification performance when using a classification network illustrated in Fig. 2.

One can think of our adversarial image perturbation method as a type of style transfer [9, 46, 45] since we transform an image into one with a different style. However, we aim to remove perceptual visual elements such as edges, outlines, and texture of the objects, while the style transfer methods try to preserve edges and outlines while changing the texture and colors.
3. Recognition-aware Obfuscation

Our goal is to allow sharing images so that people cannot recognize the image contents, yet the images can be used to train a model for a given recognition task. To achieve the goal, we have to address two sub problems: 1) how to transform the image so that it cannot be recognized by others and 2) which parts of the image may be transformed while maintaining the learnability of a visual recognition model.

For the first question, how to transform, there are many choices including blurring, down-sampling [32], pixelation [16], noise super-position [44], elastic distortion [35], or their combinations. We particularly choose to superimpose noise as it destroys local color, edge and texture information. This general obfuscating approach can be used for many applications including classification, local object detection, or fine-grained detection of parts such as facial landmarks.

For the second question, which part to transform while preserving the recognition accuracy, we use adversarial learning (AL) method [11] with a task-specific loss term. It is similar to recent efforts to generate visually less obvious data for privacy [32, 30, 40]. Our objective, however, is exactly the opposite of conventional AL. The conventional AL aims to generate images as natural as possible but which have a high probability of being classified as a different object. On the contrary, we aim to generate images as unnatural (i.e., artifact rich) as possible, yet have a high probability of being classified as the original object. We call this as “recognition-aware obfuscation”.

The generated images can be used for training another neural network for a given visual recognition task such as classification or facial landmark detection so that the models have accuracy comparable to the models trained and tested with non-obfuscated original images. This is a typical scenario where data is shared without private information but for a known use case (e.g., face images for learning a face recognition system) and the data receivers can build a recognition model, with possibly a different architecture, that performs comparably well to the model $F_{\text{orig}}$ trained with the non-obfuscated images.

Ideally, the inference accuracy of the model trained with the obfuscated data but applied to the non-obfuscated data should be comparable to the $F_{\text{orig}}$. However, this is very challenging as the feature distributions of obfuscated and non-obfuscated data may be very different. To reduce the gap between the two spaces, we provide an obfuscator $O$ that can be applied to any new data for inference. But the $O$ is typically large in size, thus computationally not efficient. Instead, we build an economical $O$, called $O^p$ ($O$ for public release), for obfuscating the test data with only a small computational cost. Furthermore, we show a practical solution to allow the model to perform well on the non-obfuscated test data (see Sec. 5.9).

We illustrate the overview of the approach, including all three inference scenarios, in Figure 2.

3.1. Formulation

For the recognition-aware obfuscation, we formulate an objective function to visually degrade the image while maintaining the recognition performance of a chosen task as follows:

$$\min_D \max_O \mathbb{E}_{x \sim P_{\text{data}}}(x) [\log D(x)] + \mathbb{E}_{x \sim P_{\text{data}}}(x) [1 - \log(D(O(x)))] + \lambda \mathcal{L}(\hat{\nu}, \nu),$$

where “target” refers to the distribution of the “target obfuscating type” in the left-most box in Fig. 2 (see Sec. 3.2 for more details). The $D(\cdot)$ is a discriminator to distinguish between the obfuscated image or the target, $O(\cdot)$ is an obfuscating network, $\mathcal{L}(\cdot)$ is a loss for the recognition task that is to be enforced in training phase, $\nu$ is a supervision for the task and $\hat{\nu}$ is a prediction by the model. We refer to the first two terms as GAN loss [11] and the last term as task-dependent recognition loss.

$\lambda$ is a balancing hyper-parameter between the GAN loss and the recognition loss. $\mathcal{L}(\cdot)$ is defined by the choice of target task. In our evaluations, we address two tasks: classification and landmark detection. For the classification task, $\mathcal{L}(\cdot)$ is defined as

$$\mathcal{L}(\hat{y}; y) = \mathbb{E}_{x \sim P_{\text{data}}(x)} [\zeta(F^O(O(x)), y)],$$

where $y$ is a classification label for each sample, $F^O(\cdot)$ is a surrogate network for the target task and $\zeta(\cdot)$ is a cross-entropy (CE) function. For landmark detection preservation, $\mathcal{L}(\cdot)$ is defined as

$$\mathcal{L}(\hat{\eta}; \eta) = \mathbb{E}_{x \sim P_{\text{data}}(x)} [\xi(F^O(O(x)), \eta)],$$

where $\eta$ is a vector of landmark locations and $\xi(\cdot)$ is the mean squared error (MSE) function.

The surrogate network of interest $F^O(\cdot)$ should define the same visual recognition task such as classification or landmark detection. But it does not need to have the same architecture as the network $F(\cdot)$ that is trained with the obfuscated data (see Sec. 5.7 for relevant experiments). Once we train $O(\cdot)$ with the surrogate network $F^O(\cdot)$, we use $O$ to obfuscate the images for training a model of interest, $F$. We illustrate what our objective function performs in the feature space in Figure 3.

Our formulation shares similarity to privacy-preserving video degradation methods [40]. However, we address more difficult problems than [40] who only minimize a similar objective but degrade the information that is not directly relevant to the tasks, whereas our problem requires degrading information that is directly relevant to the task, such as edges for digit recognition.
3.2. Type of Obfuscation

We use pixel-wise color perturbation (noise) as a type of obfuscation. Although we also considered blur [16], down-sampling, and black-out [26], we chose pixel-wise color perturbation [44] as it provides obfuscation for a wide range of applications, including image and attribute classification, landmark detection, and object detection. Differently from [8] and [40] that remove features of privacy information by maximizing prediction error of the given private labels, we degrade the input image by explicit obfuscation using GAN loss without necessity of specifying the target private labels. Since pixel-wise color perturbation is known to negatively affect the accuracy of deep neural networks [44], competition between the GAN and the recognizer in our model tries to obfuscate the input image to be unrecognizable for both human and machines, while minimally preserving features necessary for the target task.

In particular, blur is not very useful in obfuscating attributes as many attribute types are related to the local color information that the blur operation preserves. While down-sampling is effective in action recognition [32], it is not effective for image classification since the human visual system is good at guessing the details by looking at the object outlines, which the down-sampling preserve [29]. Additionally, blurring or down-sampling do not adaptively handle various scales of the objects in the scene. A small object may be averaged into a single pixel while a big object may preserve visual details still perceptible to human eyes.

Specifically, we set the obfuscating targets to be a blend of $x$ and Gaussian noise of unit variance $\mathcal{N}(0, 1)$ as follows:

$$r \leftarrow \alpha x + (1 - \alpha) \mathcal{N}(0, 1),$$  \hspace{1cm} (2)

where $\alpha \in [\theta_1, \theta_2]$ is a blending factor randomly sampled from a uniform distribution $\mathcal{U}(\theta_1, \theta_2)$, where $0 < \theta_1 \leq \theta_2 < 1$. In our experiments, we set $\theta_1 = 0.1$ and $\theta_2 = 0.4$. Higher values of $\alpha$ undesirably reduce the level of obfuscation, though making the training faster.

4. Training

Training the obfuscation network, $O$, is not trivial as training a GAN loss is not stable, and additional recognition loss makes training further unstable. Thus, balancing the terms with the hyper-parameters in Eq. 1 is also not straightforward. To stabilize the training, we use stochastic block coordinate descent method; descending with the gradient of entire formulation and descending with the gradient of the recognition loss only.

As shown in Algorithm 1, we iteratively update the discriminator $D(\cdot)$ and the obfuscating network $O(\cdot)$ using adversarial training. To improve convergence, we adopt GAN loss optimization practices\(^1\) such as adding noise to the input to the discriminator $D(\cdot)$ and training the discriminator more for each iteration.

\(^1\)https://github.com/soumith/ganhacks
### Algorithm 1: Update GAN: Discriminator and Obfuscator

1. **procedure** UPDATEGAN
2. Given:
3. \( m \leftarrow \) batch size
4. **Update D:**
5. \( \text{Sample}\ \{x(i)\}_{i=1}^{m} \sim P_{\text{data}}(x) \)
6. \( x'(i) \leftarrow O(x(i)) \) \( \triangleright \) Compute \( x' \) from \( x \)
7. \( r(i) \leftarrow \text{Obfuscating target by Eq. 2} \)
8. Maximize \( E[\log D(r(i))] + E[1 - \log(D(x'(i)))] \)
9. **Update O:**
10. \( \text{Sample}\ \{x(i)\}_{i=1}^{m} \sim P_{\text{data}}(x) \)
11. \( x'(i) \leftarrow O(x(i)) \)
12. Minimize \( \lambda L(\tilde{\nu}, \nu) + E[1 - \log(D(x'(i)))] \)

### Algorithm 2: Update Recognition Loss: Task \( F^{O} \)

1. **procedure** UPDATERECOGLoss
2. Given:
3. \( m \leftarrow \) batch size
4. \( n_{O} \leftarrow \) number of iterations of updating \( O \)
5. **Update O:**
6. \( \text{for } i = 1, \ldots, n_{O} \text{ do} \)
7. \( \text{Sample}\ \{x(i)\}_{i=1}^{m} \sim P_{\text{data}}(x) \)
8. \( x'(i) \leftarrow O(x(i)) \)
9. Minimize \( L(\tilde{\nu}, \nu) \) \( \triangleright \) Task loss

Both the GAN loss and the recognition loss compete each other to obfuscate \( x \) while minimizing the visual distortion for the classification. We can prioritize the classification task by setting \( n_{O} \) be higher than 0, which updates the obfuscation network \( O(\cdot) \) more focusing on minimizing the recognition loss \( L(\tilde{\nu}, \nu) \) in Equation 1 (see Algorithm 2). We can also optionally update the surrogate network \( F^{O}(\cdot) \) at each iteration similar to [40].

At line 7 in Algorithm 1, to make the obfuscating targets described in Section 3.2, we can also set a random subset of noise-blended images to be additional obfuscating targets in a manner similar to instance noise [36].

### 5. Experiments

#### 5.1. Datasets

We have evaluated our obfuscation method with three vision tasks using multiple datasets: CIFAR10 [19], SVHN [25], Pascal VOC 2012 [6] for image classification tasks with either single label or multi-label set-up; CelebA dataset [23] for classifying facial attributes, a challenging multi-label classification task; and MTFL dataset [42] for facial landmark detection.

CIFAR10 image size is \( 32 \times 32 \); it has 50, 000 images in training and 10,000 images in testing set. SVHN also has \( 32 \times 32 \) color images; 73, 257 images in training and 26, 032 images in testing set. We prefer SVHN over MNIST since colors and cluttered background present a more challenging scenario than MNIST. Pascal VOC 2012 was created for multi-label classification and has larger images. It has 5, 717 images in the training and 5, 823 images in the validation set. Pascal VOC 2012 is one of the most popular visual recognition benchmarks, along with ImageNet.

CelebA dataset is for a challenging multi-label classification set-up as the facial attributes are multi-faceted visual information that the obfuscation process might easily destroy. It has 40 attributes for the facial attribute classification task. We choose 8 attributes that are hard to recognize after obfuscation; e.g., slightly smiling, mouth slightly open, no beard, (wearing) eye glasses, (wearing) heavy makeup, male, wearing lipstick, and young. These are either holistic (e.g., male, young) or local-detail-oriented (e.g., mouth slightly open, wearing lipstick), which make the obfuscation challenging. We call this dataset for classifying eight challenging attributes CelebA-8.

MTFL dataset is a recent dataset for facial landmark detection, a fine-grained localization task. From the obfuscation perspective, fine-grained localization is extremely challenging as the localization should be pixel-accurate.

#### 5.2. Experimental Set-up

The architecture of the obfuscating network \( O(\cdot) \) is a simple convolutional network; details can be found in the supplementary material. We tried a set of popular architectures for the surrogate classification network \( F^{O} \), and chose the ones that performed well for a given task. Specifically, we chose AlexNet [20], MobileNet v2 [33], DenseNet [15], ResNet-50 [13] for the experiments on CIFAR10, SVHN, Pascal VOC 2012 datasets, respectively, if not specified. For the experiments on CelebA-8 and MTFL dataset, we use simple convolutional network architectures, each. Please refer to the supplementary material for details.

#### 5.3. Baselines

For each experiment, we compare our method with well-known baseline obfuscation methods that are not recognition-aware, such as Blurring, Gaussian noise, and Pixelation. We tried to minimally apply each method such that it degrades images until the images become just unrecognizable by humans for better accuracy. For the detailed configuration, please refer to the supplementary materials.

#### 5.4. Image Classification

We first validate our approach on image classification, both on the single-label and multi-label classification tasks. Single label classification is the most widely-used classification set-up where each image is associated with only one label. We use CIFAR10 and SVHN datasets for this
Figure 4. Original images and their baseline obfuscations. Each block has four columns: original, blurred, Gaussian noised, and pixelated (from left to right).

set-up. We use the obfuscation inference scenario 1 depicted in Figure 1: train a model with obfuscated images and test it by the obfuscated images with $O$. For the experiments on CIFAR10 dataset, we tried two architectures for the newly trained model $F$: AlexNet and MobileNet. Note that the performance gap between $F^{\text{orig}}$ to the $F$ is smaller in AlexNet, while the MobileNet achieves higher accuracy overall. SVHN is challenging for obfuscation in particular since the important privacy information is on the outlines of digits, so if the outlines are retained, humans can easily recognize the objects. Obfuscation then requires removing the edge information while still preserving recognition accuracy, a challenging task.

Obfuscating images in multi-label classification is more challenging than in single-label classification as the models must retain information about all labels in the image. We use PASCAL VOC 2012 dataset for the evaluation in multi-label tasks. Here, we fine-tuned ResNet50-based $F$ that had been pretrained with ImageNet dataset, using the obfuscated data $O(x)$ due to the small scale of Pascal VOC 2012 dataset (5, 717 training images). We prevent $F$ from overfitting to $O(x)$ using small learning rate of $10^{-6}$, thus reducing the gap between the underlying distribution of the obfuscated images and the distribution of the non-obfuscated (original) images. So we use the scenario 3 (train a model with the obfuscated images and test it without $O$) that makes more sense in the experiments with Pascal VOC 2012. If we use scenario 1 using $O$, the accuracy drops a bit to 78.36%.

We also compare with the baseline obfuscating methods that are not recognition-aware. The methods include blur, noise, and pixelation in Table 1.

| Dataset       | $F^{\text{orig}}$ | Baselines       | Ours ($F$) |
|---------------|-------------------|-----------------|------------|
| CIFAR10 (AlexNet) | 83.84             | 69.04 70.45 70.08 | 82.81      |
| CIFAR10 (MobileNet) | 90.60             | 73.46 72.50 73.11 | 88.35      |
| SVHN          | 96.6 (98.11)      | 82.14 67.62 52.62 | 80.01      |

Table 1. Accuracy (%) in image classification task. The Baseline refers to non-recognition-aware baselines. Numbers in () denotes the accuracy of the model trained further with extra datasets. Note that performance of our method in VOC2012 (denoted by *) uses scenario 3 (others use scenario 1).

We present some examples that our obfuscator generates on three datasets for the classification task in Fig. 5 for qualitative analysis. The detailed contents of each image are hardly recognizable as the obfuscator removes the textures (CIFAR10, SVHN), edges (SVHN) and privacy-related details (CIFAR10, VOC2012). VOC2012 contains objects of various scales in the scene, which makes obfuscation challenging and requires adaptively handling multiple scales of the input images. Note that the images in the second rows are used to train a visual recognition model that performs comparably to the ones trained with non-obfuscated images (Table 1).

**Peak Signal to Noise Ratio (PSNR).** PSNR is used as one of the direct measurements of image quality or its degradation. We employ it as a proxy measure of obfuscation in human perception. Table 2 shows the average PSNR of all images in each dataset. Note that PSNR less than 10 dB generally implies really low quality.

| Unit (dB) | CIFAR10 | SVHN | VOC2012 |
|-----------|---------|------|---------|
| Mean PSNR | 3.45    | 6.32 | 8.80    |

Table 2. Mean PSNR (dB). The lower, the noisier.

**Human Perceptual Study.** Since the PSNR can only serve as a proxy measure of how much information was lost for identifying the image class label by human, we also directly study the human classification accuracy with the images obfuscated by our networks, since human is performance is more robust to noise than neural networks on object recognition tasks [10]. For the human study, we asked 21 human subjects (an odd number for tie breaking) to classify 100 obfuscated images of CIFAR10 and SVHN. As shown in Table 3, humans perform poorly with the obfuscated images.
due to the missing details.

We omit VOC 2012 dataset from this study as the number of ground truth multi-labels are subjectively chosen, thus the number of objects to label is not obvious for the human subjects.

| Unit (%) | Chance | Human |
|----------|--------|-------|
| CIFAR10  | 10.00  | 13.49 |
| SVHN     | 10.00  | 7.43  |

Table 3. Image Classification Accuracy (%) by Humans for 100 randomly chosen obfuscated images.

5.5. Attribute Classification

The second task we evaluated was classifying facial attributes using CelebA-8 dataset. We again compare with the same set of baseline obfuscating methods that are not recognition-aware. The classwise accuracy is compared in Fig. 6. Preserving multi-faceted visual attributes under obfuscation is quite challenging since for a visual element (e.g., edge), the multiple attributes disagree whether a detail can be removed (‘WearingLipstick’ attribute) or not (‘Eye-glasses’ attribute). Interestingly, our method allows learning a model that performs comparably to $F_{\text{orig}}$ with only a small margin of 0.48%.

| Dataset | $F_{\text{orig}}$ | Baselines | Ours ($F$) |
|---------|------------------|-----------|------------|
| CelebA-8| 96.98            | Blur      | Noise      | Pixelation | Ours |
|         | 94.30            | 84.84     | 91.82      | 96.50      |

Table 4. Accuracy (%) in attribute classification task (average over all classes).

Fig. 7 presents obfuscated examples that generated on the CelebA-8 dataset. The detailed contents of each image are hardly recognizable as the obfuscator removes textures (classification on CIFAR10, SVHN) and edges (SVHN for classification, MTFL for landmark detection), attribute-related information, e.g., wearing sunglasses, smiling, beard (attribute classification on CelebA-8) and privacy related details (VOC2012). Note that these images are used to train a visual recognition model that performs comparable to the ones trained with non-obfuscated images.

Figure 7. Qualitative results of CelebA dataset. First row: Original, Second row: Obfuscated version.

The PSNR of the images obfuscated by our method in the CelebA-8 dataset is 4.45 dB (< 10 dB). We skip human perceptual study for this task as the goal of obfuscation in this case is to remove identity of the face, not the attributes.

5.6. Facial Landmark Detection

Landmark detection estimates locations of two eyes, nose, and two mouth end-points. It is a challenging task for privacy preserving obfuscation as the task requires preserving the details of facial components, but the same locations contain identity-related information which the obfuscation algorithm tries to remove. We used MTFL dataset which was originally created for multi-task Facial Landmark Detection [41]. Table 5 shows landmark detection mean error, a widely used metric for detection [3, 4]. We compared with the baseline obfuscation methods: blur, noise, and pixelation (4 × 4 pixel block). The $F$ trained with the data obfuscated by our method perform comparably to the $F_{\text{orig}}$ by outperforming all other baselines by large margins. The PSNR of the obfuscated images from MTFL dataset is 4.31 dB (< 10 dB).

| Dataset | $F_{\text{orig}}$ | Baselines | Ours ($F$) |
|---------|------------------|-----------|------------|
| MTFL    | 0.0796           | Blur      | Noise      | Pixelation | Ours |
|         | 0.2707           | 0.2005    | 0.1455     | 0.0936     |

Table 5. Mean relative error in facial landmark detection (lower is better). The mean error is measured by the distances between estimated landmarks and the ground truths, normalized by the interocular distance.

Fig. 8 presents obfuscated examples that generated on the CelebA-8 dataset. The detailed contents of each image are hardly recognizable as the obfuscator removes textures (classification on CIFAR10, SVHN) and edges (SVHN for classification, MTFL for landmark detection), attribute-related information, e.g., wearing sunglasses, smiling, beard (attribute classification on CelebA-8) and privacy related details (VOC2012). Note that these images are used to train a visual recognition model that performs comparable to the ones trained with non-obfuscated images.

Figure 8. Qualitative results of facial landmark detection with MTFL dataset. First row: Original; Second row: Obfuscated images; Third row: Landmark detection results, white dot indicates ground-truth locations and blue dot marks our output.
5.7. Learnability using Obfuscated Images

We compare learnability of the obfuscated data $O(X)$ with a baseline of FGSM [12], the most successful method for generating adversarial examples by computing the accuracy of a new model $F$ trained with the data generated by the FGSM and the data generated with our $O(X)$. FGSM uses gradients of the target network to generate the adversarial samples. Using FGSM in reverse is the simplest way to generate a differently looking image that maintains the classification accuracy.

FGSM is, however, not suitable when sharing data for training new models since the gradients must be generated using the same model that was used in original training. Even if we train a second model with the same architecture, but use the obfuscated data $O(X)$, the gradient from the newly trained model will be very different from the original FGSM model. In contrast, our $O()$ model which also utilizes the gradient information to train, allows the obfuscated data to be used for training a new model regardless of the architecture. Note that the classification accuracy of the model trained with our obfuscated images is significantly better than the accuracy of the model trained with FGSM-generated data.

Table 6 compares the accuracy of models trained with FGSM data and our $O()$ model with different combinations of $F^O$ and $F$. The model trained with the data generated by FGSM shows accuracy close to random chance (10%), which is consistent with the results in [22]. The authors empirically showed that the gradient-based method was not transferable to a newly trained network as an attack even when the architectures are homogeneous [22]. The results in Table 6 imply that the non-transferable data are also not learnable by another network.

5.8. Obfuscator for Public Sharing ($O^p$)

Obfuscator $O$ needs to be shared so new data can be preprocessed before classification. However, sharing and continuously running a large $O$ is inconvenient. If we can provide a more compact version $O^p$, the new model can be run more efficiently. We use knowledge distillation method [14] to learn $O^p$. Table 7 shows the classification accuracy of the classification models trained on $X$, $O(X)$ and $O^p(X)$ on CIFAR10, SVHN, and VOC2012 datasets. Despite the huge efficiency gain in the number of parameters, models trained with $O^p(X)$ show very comparable accuracy to using $O(x)$.

5.9. Enabling Inference without $O$ or $O^p$

Inference using non-obfuscated data by models trained with obfuscated data is challenging as the feature distributions of obfuscated and non-obfuscated data may be very different. To reduce the gap between the two underlying data distributions, we use some non-obfuscated data that may not be privacy sensitive when training a new model. If the model gets a glimpse of the non-obfuscated data distribution, it can infer the non-obfuscated data better.

To simulate this scenario, we plot the accuracy curve as a function of ratio between non-obfuscated data and the obfuscated data in training phase in Fig. 9 using CIFAR10 and SVHN dataset. With only a small fraction (5~10%) of non-obfuscated data in training, we already achieved comparable accuracy to the $F^{orig}$, which is the model trained with 100% non-obfuscated data.

| CIFAR10 | SVHN |CelebA-8 |
|---------|------|---------|
| X | $O(X)$ | $O^p(X)$ | #Param | $O^p$ |
| CIFAR10 (AlexNet) | 83.84 | 82.81 | 81.87 | 20,451 | 20,451 |
| CIFAR10 (MobileNet) | 90.60 | 88.35 | 87.67 | 20,451 | 20,451 |
| SVHN | 96.6 | 92.1 | 90.7 | 30,051 | 30,051 |
| CelebA-8 | 96.98 | 96.50 | 94.32 | 20,451 | 20,451 |

Table 7. Classification accuracy (%) of the models trained with $X$, $O(X)$, and $O^p(X)$.

Figure 9. Classification accuracy (%) of the model trained with the mix of the obfuscated and non-obfuscated data and tested with the non-obfuscated data. ‘Portion of X’ refers to the percentage of the non-obfuscated data used in training. 0% means that the model is trained only with the obfuscated data. 100% is identical to $F^{orig}$.

6. Conclusion

We propose to learn to obfuscate training images so that humans cannot recognize the contents or privacy-related information, while machines can still utilize them to train a new model. We demonstrate that the obfuscation does...
confound humans using a human study, and also demon-
strate that the images still allow successful training of new
networks for image classification, attribute classification,
and landmark detection tasks, so that the networks per-
form almost as well as those trained with the original, non-
obfuscated data.

As a future work, we want to combine different obfuscat-
ing targets including blur, pixelation, black-out, and noise
to provide additional versatility to recognition-aware image
obfuscation.

Appendix

A. Architectures of $O(\cdot)$, $F^O(\cdot)$ and $O^p(\cdot)$

A.1. Notation

We present the architectures of $O(\cdot)$, $F^O(\cdot)$, and $O^p(\cdot)$
in tables. Each row of the table shows a layer-group con-
sisting of two or three layers in order. The layer group is
denoted by a combinations of symbols. The symbols are
C, B, R, L, E, M, and D denoting convolution, batch nor-
malization, ReLU, LeakyReLU ($\alpha = 0.2$ for all experi-
ments), ELU ($\alpha = 1.0$ for all experiments), Max-pooling
and Drop-out ($p = 0.5$ for all experiments) layer, respec-
tively. The last symbol at the final layers, namely S or T, de-
notes sigmoid or tanh activation functions, respectively.

Each column of the table specifies the hyperparameters
of each layer group; a width (or height) of a square kernel
(Kernel), stride (Stride), size of zero padding (Padding), and
output dimensions (Out).

For all experiments, the input of the first layer has three
channels.

A.2. Architectures of $O(\cdot)$

Table 8-11 show the detailed architectures of $O(\cdot)$ used
in the experiments with the CIFAR10, SVHN and CelebA-
8, MTFL, and VOC2012 dataset, respectively. Note that it
is not trivial to obtain the inverse of $O(\cdot)$ as it has multiple
ReLU layers which degenerate the signal less than zero to
zero.

| Layer | Kernel | Stride | Padding | Dilation | Out |
|-------|--------|--------|---------|----------|-----|
| CBR   | 3      | 1      | 1       | 1        | 32  |
| CBR   | 3      | 1      | 1       | 1        | 32  |
| CBR   | 3      | 1      | 1       | 1        | 32  |
| CS    | 3      | 1      | 1       | 1        | 3   |

Table 8. Architecture of $O(\cdot)$ for CIFAR10

A.3. Architectures of $F^O(\cdot)$

CelebA-8. Table 12 shows the detailed architecture of
$F^O(\cdot)$ used in the experiments with the CelebA-8 dataset.

| Layer | Kernel | Stride | Padding | Dilation | Out |
|-------|--------|--------|---------|----------|-----|
| CBR   | 3      | 1      | 1       | 1        | 32  |
| CBR   | 3      | 1      | 2       | 2        | 32  |
| CBR   | 3      | 1      | 1       | 1        | 32  |
| CBR   | 3      | 1      | 4       | 4        | 32  |
| CT    | 3      | 1      | 1       | 1        | 3   |

Table 9. Architecture of $O(\cdot)$ for SVHN and CelebA-8

| Layer | Kernel | Stride | Padding | Dilation | Out |
|-------|--------|--------|---------|----------|-----|
| CBR   | 3      | 1      | 1       | 1        | 32  |
| CBR   | 3      | 1      | 2       | 2        | 32  |
| CBR   | 3      | 1      | 1       | 1        | 32  |
| CBR   | 3      | 1      | 4       | 4        | 32  |
| CS    | 3      | 1      | 1       | 1        | 3   |

Table 10. Architecture of $O(\cdot)$ for MTFL

MTFL. Table 13 shows the detailed architecture of $F^O(\cdot)$
used in the experiments with the MTFL dataset.

| Layer | Kernel | Stride | Padding | Dilation | Out |
|-------|--------|--------|---------|----------|-----|
| CBR   | 3      | 1      | 0       | 1        | 8   |
| M     | 2      | 2      | -       | -        | 8   |
| CBR   | 3      | 1      | 0       | 1        | 16  |
| M     | 2      | 2      | -       | -        | 16  |
| CBR   | 3      | 1      | 0       | 1        | 32  |
| M     | 2      | 2      | -       | -        | 32  |
| CBR   | 3      | 1      | 1       | 1        | 64  |
| M     | 2      | 2      | -       | -        | 64  |
| C     | 3      | 1      | 0       | 1        | 8   |

Table 11. Architecture of $O(\cdot)$ for VOC 2012 experiments. It is
based on U-Net architecture [17], which has skip connections be-
tween $i$th layer in the encoder (five layers above the dashed line)
and $(6 - i)$th layer in the decoder (five layers below the dashed
line), where $i = 1, \ldots, 5$.

| Layer | Kernel | Stride | Padding | Dilation | Out |
|-------|--------|--------|---------|----------|-----|
| CBL   | 4      | 2      | 1       | 1        | 16  |
| CBL   | 4      | 2      | 1       | 1        | 32  |
| CBL   | 4      | 2      | 1       | 1        | 64  |
| CBL   | 4      | 2      | 1       | 1        | 128 |
| CBR   | 4      | 2      | 1       | 1        | 128 |
| CBDR  | 4      | 2      | 1       | 1        | 256 |
| CBDR  | 4      | 2      | 1       | 1        | 128 |
| CBDR  | 4      | 2      | 1       | 1        | 64  |
| CBR   | 4      | 2      | 1       | 1        | 32  |
| CS    | 4      | 2      | 1       | 1        | 3   |

Table 12. Architecture of $F^O(\cdot)$ for CelebA-8

A.4. Architectures of $O^p(\cdot)$ for Knowledge Distilla-
tion

Table 14-16 show the detailed architecture of $O^p(\cdot)$
which is a compact version of $O(\cdot)$ and learned by the
knowledge distillation [14] used in the experiments with CI-
| Layer | Kernel | Stride | Padding | Dilation | Out |
|-------|--------|--------|---------|----------|-----|
| CR    | 3      | 1      | 1       | 1        | 32  |
| CR    | 3      | 1      | 1       | 1        | 64  |
| M     | 2      | 2      | -       | -        | 64  |
| CR    | 3      | 1      | 1       | 1        | 128 |
| CR    | 3      | 1      | 1       | 1        | 192 |
| M     | 2      | 2      | -       | -        | 192 |
| CR    | 3      | 1      | 1       | 1        | 384 |
| CR    | 3      | 1      | 1       | 1        | 256 |
| M     | 2      | 2      | -       | -        | 256 |
| CR    | 3      | 1      | 1       | 1        | 128 |
| CR    | 3      | 1      | 1       | 1        | 192 |
| M     | 2      | 2      | -       | -        | 192 |
| CR    | 3      | 1      | 1       | 1        | 384 |
| CR    | 3      | 1      | 1       | 1        | 256 |
| M     | 2      | 2      | -       | -        | 256 |
| CBR   | 3      | 1      | 1       | 1        | 128 |
| CBR   | 3      | 1      | 1       | 1        | 192 |
| M     | 2      | 2      | -       | -        | 192 |
| CBR   | 3      | 1      | 1       | 1        | 384 |
| CBR   | 3      | 1      | 1       | 1        | 256 |
| MT    | 2      | 2      | -       | -        | 256 |

Table 13. Architecture of \( F^O (\cdot) \) for MTFL

FAR10, SVHN and CelebA-8 dataset, respectively.

| Layer | Kernel | Stride | Padding | Dilation | Out |
|-------|--------|--------|---------|----------|-----|
| CBE   | 3      | 1      | 1       | 1        | 8   |
| CBE   | 3      | 1      | 1       | 1        | 8   |
| CS    | 3      | 1      | 1       | 1        | 3   |

Table 14. Architecture of \( O^P (\cdot) \) for CIFAR10

| Layer | Kernel | Stride | Padding | Dilation | Out |
|-------|--------|--------|---------|----------|-----|
| CBR   | 3      | 1      | 1       | 1        | 16  |
| CBR   | 3      | 1      | 2       | 2        | 16  |
| CBR   | 3      | 1      | 1       | 1        | 16  |
| CT    | 3      | 1      | 1       | 1        | 3   |

Table 15. Architecture of \( O^P (\cdot) \) for SVHN

| Layer | Kernel | Stride | Padding | Dilation | Out |
|-------|--------|--------|---------|----------|-----|
| CBR   | 3      | 1      | 1       | 1        | 32  |
| CBR   | 3      | 1      | 2       | 2        | 32  |
| CBR   | 3      | 1      | 1       | 1        | 32  |
| CT    | 3      | 1      | 1       | 1        | 3   |

Table 16. Architecture of \( O^P (\cdot) \) for CelebA-8

### B. Hyperparameters of Baselines

We present the hyperparameters of non recognition-aware baselines; Blur, Noise and Pixelate.

#### B.1. Blur

For the blur baseline, we use Gaussian blurring with kernel size \( k \) and standard deviation \( \sigma \). Table 17 shows hyperparameters we used in the experiments for the blur baseline.

#### B.2. Pixel-wise Color Perturbation - Noise

For the noise baseline, we use Gaussian noise with mean \( \mu \) and standard deviation \( \sigma \). Table 18 shows hyperparameters we used in the experiments for the noise baseline.

| Dataset | kernel size \( (k) \) | std-dev \( (\sigma) \) |
|---------|----------------------|---------------------|
| CIFAR10 | 11                   | 3.0                 |
| SVHN    | 13                   | 8.0                 |
| CelebA-8| 15                   | 5.0                 |
| MTFL    | 15                   | 5.0                 |
| VOC2012 | 15                   | 5.0                 |

Table 17. Hyperparameters of Blur Baseline

| Dataset | mean \( (\mu) \) | std-dev \( (\sigma) \) |
|---------|----------------|---------------------|
| CIFAR10 | 0             | 1.0                 |
| SVHN    | 0             | 1.0                 |
| CelebA-8| 0             | 2.0                 |
| MTFL    | 0             | 1.0                 |
| VOC2012 | 0             | 1.0                 |

Table 18. Hyperparameters of Noise Baseline

#### B.3. Pixelation

For the pixelation baseline, we use pixelation by block-wise color averaging. Table 19 shows hyperparameters we used in the experiments for the pixelation baseline.

| Dataset | block-size \( (n) \) |
|---------|---------------------|
| CIFAR10 | 4                   |
| SVHN    | 8                   |
| CelebA-8| 4                   |
| MTFL    | 4                   |
| VOC2012 | 8                   |

Table 19. Hyperparameters of Pixelation. Block-size \( n \) denotes \( n \times n \) pixel-blocks are clustered into a super-pixel

### References

[1] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein GAN. In ICML, 2017. 2
[2] S. Baluja and I. Fischer. Learning to Attack: Adversarial Transformation Networks. In AAAI, 2018. 2
[3] X. P. Burgos-Artizzu, P. Perona, and P. Dollár. Robust face landmark estimation under occlusion. In Proceedings of the IEEE International Conference on Computer Vision, pages 1513–1520, 2013. 7
[4] X. Cao, Y. Wei, F. Wen, and J. Sun. Face alignment by explicit shape regression. International Journal of Computer Vision, 107(2):177–190, 2014. 7
[5] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In CVPR, 2009. 1
[6] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html. 5
[7] B. G. Fabian Caba Heilbron and J. C. Niebles. ActivityNet: A Large-Scale Video Benchmark for Human Activity Understanding. In CVPR, pages 961–970, 2015.

[8] C. Feutry, P. Piantanida, Y. Bengio, and P. Duhamel. Learning anonymized representations with adversarial neural networks. CoRR, abs/1802.09386, 2018.

[9] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, R. Geirhos, D. H. J. Janssen, H. H. Schütt, J. Rauber, G. Hinton, O. Vinyals, and J. Dean. Distilling the

[10] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017.

[11] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. arXiv, 2015.

[12] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger. Densely Connected Convolutional Networks. In CVPR, 2017.

[13] J. F. Hughes, A. van Dam, M. McGuire, D. F. Sklar, J. D. Foley, S. Feiner, and K. Akeley. Computer Graphics: Principles and Practice. Addison-Wesley, 3 edition, 2013.

[14] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017.

[15] A. Jourabloo, X. Yin, and X. Liu. Attribute Preserved Face De-identification. In International Conference on Biometrics (ICB), 2015.

[16] A. Krizhevsky. Learning Multiple Layers of Features from Tiny Images. Technical report, 05 2012.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In NIPS, 2012.

[18] A. Kurakin, I. J. Goodfellow, and S. Bengio. Adversarial examples in the physical world. ICLR, 2017.

[19] Y. Liu, X. Chen, C. Liu, and D. Song. Delving into transferable adversarial examples and black-box attacks. In ICLR, 2017.
[38] S. Thrun. A Lifelong Learning Perspective for Mobile Robot Control. In V. Graefe, editor, *Intelligent Robots and Systems*. 1995. 1

[39] Verisec. GDPR: White paper on General Data Protection Regulation. Technical report, 2016. 1

[40] Z. Wu, Z. Wang, Z. Wang, and H. Jin. Towards Privacy-Preserving Visual Recognition via Adversarial Training: A Pilot Study. In *ECCV*, 2018. 2, 3, 4, 5

[41] Z. Zhang, P. Luo, C. C. Loy, and X. Tang. Facial landmark detection by deep multi-task learning. In *European Conference on Computer Vision*, pages 94–108. Springer, 2014. 7

[42] Z. Zhang, P. Luo, C. C. Loy, and X. Tang. Learning Deep Representation for Face Alignment with Auxiliary Attributes. *IEEE Trans. PAMI*, 38(5):918–930, 2016. 5

[43] Z. Zhao, D. Dua, and S. Singh. Generating Natural Adversarial Examples. In *ICLR*, 2018. 2

[44] Y. Zhou, S. Song, and N.-M. Cheung. On classification of distorted images with deep convolutional neural networks. *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1213–1217, 2017. 3, 4

[45] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. In *ICCV*, 2017. 2

[46] J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, and E. Shechtman. Toward Multimodal Image-to-Image Translation. In *NIPS*, 2017. 2