FaceMix: Privacy-Preserving Facial Attribute Classification on the Cloud

Zhijian Liu[1*] Zhanghao Wu[1*] Ligeng Zhu[1] Chuang Gan[2] Song Han[1]
[1]Massachusetts Institute of Technology [2]MIT-IBM Watson AI Lab
{zhijian, zhwu, ligeng, chuangg, songhan}@mit.edu

Abstract

*Deep neural networks are widely deployed on edge devices (*e.g.*, for facial attribute classification). Users either perform the inference locally (*i.e.*, edge-based) or send the data to the cloud and run inference remotely (*i.e.*, cloud-based). However, both solutions have their limitations: edge devices are heavily constrained by the insufficient hardware resources and cannot afford to run large models; cloud servers, if not trustworthy, will raise serious privacy issues. In this paper, we mediate between the resource-constrained edge devices and the privacy-invasive cloud servers by introducing a novel privacy-preserving edge-cloud inference framework, FaceMix, for the facial attribute classification networks. We offload the majority of the computations to the cloud and leverage a pair of encryption and decryption functions to protect the privacy of the data transmitted to the cloud. Our framework has three advantages. First, it is privacy-preserving as our encryption cannot be inverted without user's private key. Second, our framework is accuracy-preserving because our encryption takes advantage of the linearity, and we train the model in an encryption-aware manner to help maintain the accuracy. Third, our solution is efficient on the edge since the majority of the workload is delegated to the cloud, and our encryption and decryption processes introduce very few extra computations. Also, our framework introduces small communication overhead and maintains high hardware utilization on the cloud. Extensive experiments on multiple facial attribute classification datasets demonstrate that our framework can greatly reduce the local computations on the edge (to fewer than 20% of FLOPs) with negligible loss of accuracy and no leakages of private information.*

1. Introduction

Deep neural networks have significantly improved the performance of human face related tasks and are widely applied in many applications. For example, assisted by the facial attribute classification networks, Google Photos can automatically categorize users' photos by the emotions on their faces. However, the high performance and superior accuracy of deep neural networks always come at the expense of larger model size and more computations. That makes it difficult to be deployed on resource-constrained edge devices: mobile applications require fast reponse and low energy cost, while edge devices have limited hardware resources and tight power budgets. To address these challenges, researchers have proposed to either directly design the compact models [19, 46] or accelerate the existing models by compression [23, 13]. However, a bottleneck is the ceiling of accuracy that small models can achieve. To our best knowledge, it is rather challenging to achieve high accuracy with very compact models: *e.g.*, with roughly 200M FLOPs of computations, the state-of-the-art mobile models [18] can only achieve 75% of top-1 accuracy on ImageNet, which is still 10% lower than the best performances [54].

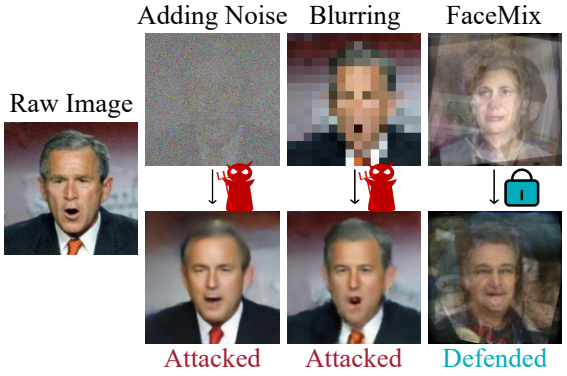
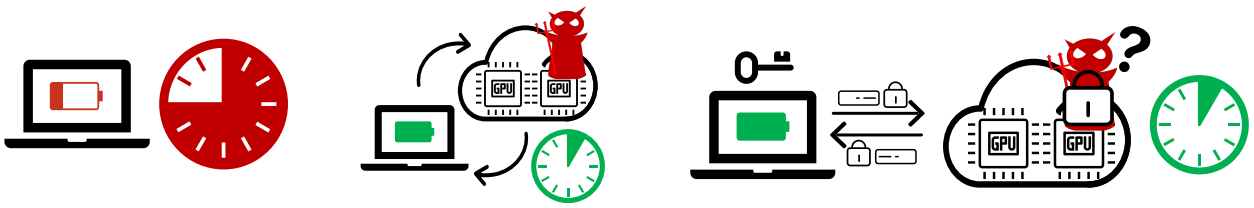


Figure 1: Adding noise to the image and blurring the image are not secure: the original image can be recovered by the GAN-based attack model. Our proposed FaceMix preserves the privacy of facial images better.

In contrast, the cloud servers have much more computation resources and power budgets than edge devices. With the next generation wireless network (*i.e.*, 5G) approaching, the high bandwidth and low latency of the technique will lead to a fundamental change of the way we process information both on the edge devices and cloud servers, which

*indicates equal contributions. The first two authors are listed in the alphabetical order.



(a) Edge-Based Inference (b) Cloud-Based Inference (c) Privacy-Preserving Edge-Cloud Inference

Figure 2: Edge devices are *resource-limited*, and cloud servers are *privacy-invasive*. Our proposed framework takes advantage of both, providing *low-latency*, *privacy-preserving* model inference.

will affect the paradigm of AI computing. The communication latency will be significantly reduced, and the cloud servers can then handle the computation for the edge devices without sacrificing the real-time experience. Taking advantage of the computation power on both cloud and edge will offer new opportunities for efficient AI computing. However, cloud-based solutions raise privacy issues as the cloud servers might be malicious. Users' photos, in many cases, is very privacy-sensitive: *i.e.*, users may not want to disclose their personal information to the the cloud (such as identity, age, gender, and race). Therefore, privacy-preserving inference is of critical importance.

This paper presents a novel perspective to tackle this challenge. We introduce the *privacy-preserving edge-cloud inference* framework, FaceMix, to bring the best of the privacy-preserving edge devices and resource-abundant cloud servers together. We delegate the majority of the computations to the cloud, therefore reducing the local resource requirements. We design a pair of encryption and decryption functions to protect the privacy of the data transmitted to the cloud and train the model in an encryption-aware manner to help maintain the accuracy. Our framework is a general method for cloud-edge inference and can be applied to other modalities, but the privacy issue is more important for the human face. Therefore, we mainly conduct our experiments on the facial attribute classification. We evaluate our FaceMix on two popular benchmarks, CelebA and LFWA, which contains much private information, including personal identity, age, and race. Our framework can greatly improve the efficiency on the edge with negligible loss of accuracy and no leakages of private information, providing a superior trade-off among efficiency, accuracy and privacy compared with previous approaches.

2. Related Work

2.1. Facial Attribute Classification

Facial attribute classification aims to recognize various human attributes, including gender, age and emotions from face appearance [24, 29, 56]. It is widely applied in many computer vision systems, such as person re-identification [30, 49, 34] and smart photo albums. Deep neural networks have been applied to solve the problems and achieve high accuracy [12, 36]. However, when the user has to inference many images, these classification models will consume much time as well as energy.

2.2. Efficient Inference

Considerable efforts have been made to design efficient models under tight resource constraints on the edge devices while maintaining the high performances at the same time, such as SqueezeNets [9, 20], MobileNets [19, 46] and ShuffleNets [55, 37]. Another approach to achieve the efficient inference is to compress and accelerate the existing large models. For instance, some have proposed to prune the separate neurons [14, 13] or the entire channels [17, 35, 16]; others have proposed to quantize the network [7, 57, 28, 52] to accelerate the model inference.

However, it is very challenging to achieve the state-of-the-art performance with these compact models. In this paper, we provide a new solution to the efficient inference, that is to make use of the computing power on the cloud without compromising privacy.

2.3. Privacy-Preserving Inference

There have been extensive investigations on the problem of privacy in the machine learning. Osia *et al*. [41] summarized the previous works into mainly three categories: dataset publishing [3, 2, 22], model sharing [8, 47, 1] and private inference. Our paper falls into the last category, which is to perform the inference on the cloud without leaking any private information.

Researchers proposed different approaches of private inference on specific tasks: *e.g.*, performing activity recognition on extremely low-resolution videos [45, 5, 44, 6]. As for the face-related tasks, researchers have introduced various face de-identification methods to help protect the privacy [38, 4, 25, 39, 33]. However, most of them either require much computation or compromise to the model accuracy degradation.

Inspired by the generative adversarial networks (GANs) [10], researchers proposed to train one neural network to obfuscate the input data and train another neural network to recover the original data in an adversarial man-

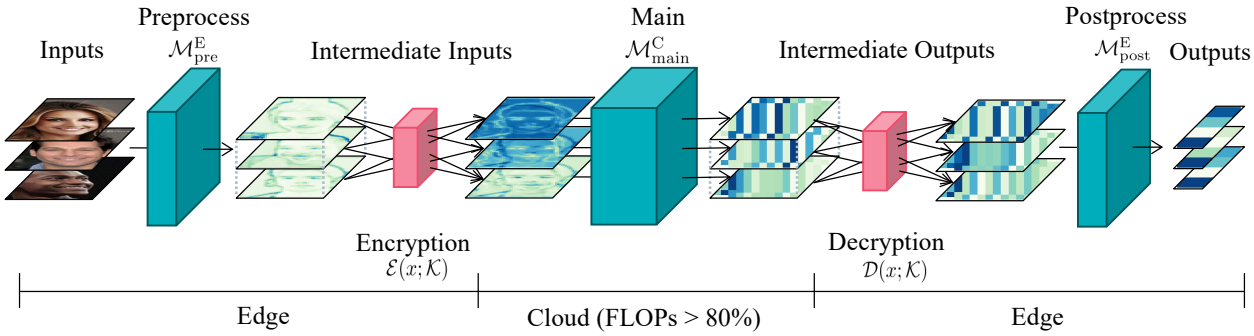


Figure 3: Partitioning the neural network into three parts, our framework leverages the encryption and decryption to protect the privacy of the data transmitted to the cloud.

ner [40, 43, 31, 53, 26]. However, these obfuscators are dedicated to one particular adversarial attacker and might not be able to generalize to other attack methods. Furthermore, these works do not take the efficiency into consideration, and some obfuscators are very computationally expensive: *e.g.*, 30× more FLOPs than ResNet-18 [43].

2.4. Hybrid Edge-Cloud Inference

There are several preliminary attempts [41, 42, 32, 51] to leverage both the edge and the cloud during the model inference. However, in order to maintain the accuracy, these frameworks need to send very deep layers to the cloud (*e.g.*, after `conv5-1` for VGG-16), which means that more than 90% of the computation still needs to be performed on the edge. Besides, they only consider the input privacy; however, the outputs (*i.e.*, prediction result) might also be very sensitive: *e.g.*, for facial attribute classification, the prediction of the user's race can lead to some racial behaviors, and it therefore should be equally important as the input.

Recently, Tramer *et al.* [50] proposed to delegate the executions of all linear layers in a DNN from a trusted processor (TEEs) to another untrusted processor. However, it requires two processors to be co-located as it requires transmitting the activations of *all* the linear layers, which brings a large communication overhead. Therefore, the approach is not suitable for the edge-cloud inference setting where the transmission cost between the edge (trusted processor) and the cloud (untrusted processor) is expensive. We highlight the differences (w.r.t. efficiency, privacy and communication overhead) between these previous frameworks and our proposed framework in Table 1.

3. Method

In this section, we first describe the problem setting; we then propose our *privacy-preserving edge-cloud inference* framework; we finally design a pair of encryption and decryption functions.

3.1. Problem Setting

In this paper, we focus on the *model inference on the edge devices*. As these edge devices are tightly resource-constrained, it is beneficial for them to offload the computations to the cloud for fast and efficient inference. We assume that *the users' data is privacy-sensitive* and *the cloud is malicious*. In our setting, the cloud has the weights of the neural network model and wants to recover user's private information from each request. Our goal is to develop effective and efficient encryption and decryption functions so that the cloud cannot recover user's data from user's request.

3.2. Privacy-Preserving Edge-Cloud Inference

As illustrated in Figure 3, we partition the neural network model $\mathcal{M}$ sequentially into three parts:

$$\mathcal{M} = \mathcal{M}^{\mathrm{E}}_{\mathrm{post}} \circ \mathcal{M}^{\mathrm{C}}_{\mathrm{main}} \circ \mathcal{M}^{\mathrm{E}}_{\mathrm{pre}}, \tag{1}$$

where $\mathcal{M}^{\mathrm{E}}_{\mathrm{pre}}$, $\mathcal{M}^{\mathrm{C}}_{\mathrm{main}}$ and $\mathcal{M}^{\mathrm{E}}_{\mathrm{post}}$ represent the *preprocess*, *main* and *postprocess* models, respectively.

During the inference, the edge first runs the preprocess model on the *raw input* $\boldsymbol{x}$ and sends its output (*i.e.*, *intermediate input*) $\tilde{\boldsymbol{x}}$ to the cloud; then, the cloud runs the main model on $\tilde{\boldsymbol{x}}$ and transmits its output (*i.e.*, *intermediate output*) $\tilde{\boldsymbol{y}}$ back to the edge; finally, the edge executes the postprocess model on $\tilde{\boldsymbol{y}}$ to obtain the *final output* $\boldsymbol{y}$. Throughout the whole procedure, the cloud only has access to the intermediate input $\tilde{\boldsymbol{x}}$ instead of the raw input data $\boldsymbol{x}$.

However, without encrypting the transmissions, the framework is not yet secure for both input and output privacy. For the input data, it is possible to train a neural network to approximate the inverse function of the preprocess model if it is relatively shallow, and the cloud can then recover the input by running the inference of the *inverse* preprocess model on $\tilde{\boldsymbol{x}}$. For the output data, the cloud can simply rerun the inference of the postprocess model on $\tilde{\boldsymbol{y}}$ since the cloud has access to the weights.

To solve these issues, we extend our framework with a private key $\mathcal{K}$ and a pair of encryption function $\mathcal{E}(\boldsymbol{x};\mathcal{K})$ and

	Baseline (cloud)	Osia *et al.* [41, 42]	Tramer *et al.* [50]	Ours	
Computation (on edge)	0%	93%	∼1%*	1%	13%
Transmission size	0.6 MB	0.4 MB	86.5 MB	12.3 MB	3.0 MB
Transmission time	0.4 sec	0.3 sec	33.1 sec	6.6 sec	1.7 sec
GPU utilization (cloud)	100%	100%	∼10%*	100%	100%
Input privacy	×	✓	✓	✓	✓
Output privacy	×	×	✓	✓	✓

Table 1: Our framework achieves high *efficiency* on the edge, introduces small *network communication overhead*, attains full *resource utilization* on the cloud, and protects both *input and output privacy*. All benchmarks are conducted on VGG-16 [48] with input image size of 224×224, and the transmissions are over the 4G LTE network with upload speed of 15 Mbps, download speed of 30 Mbps, and delay of 25 ms. As for our framework, we send the output activation of the first or second convolution layer to the cloud (the last two columns). In this table, the red entries are unsatisfactory.

decryption function $\mathcal{D}(\boldsymbol{x};\mathcal{K})$. We assign each user with a *unique* private key $\mathcal{K}$, which, we assume, is only possessed by the user, and the cloud does not have access to. We use the encryption and decryption functions to preserve the privacy of the data transmitted to the cloud: the user encrypts the intermediate input $\tilde{\boldsymbol{x}}$ with the encryption $\mathcal{E}(\boldsymbol{x};\mathcal{K})$ before transmitting it to the cloud and decrypts the intermediate output $\tilde{\boldsymbol{y}}$ with the decryption function $\mathcal{D}(\boldsymbol{x};\mathcal{K})$ after receiving it from the cloud:

$$\begin{aligned}\tilde{\boldsymbol{x}} &= \mathcal{E}(\mathcal{M}^{\text{E}}_{\text{pre}}(\boldsymbol{x});\mathcal{K}),\\ \tilde{\boldsymbol{y}} &= \mathcal{M}^{\text{C}}_{\text{main}}(\tilde{\boldsymbol{x}}),\\ \boldsymbol{y} &= \mathcal{M}^{\text{E}}_{\text{post}}(\mathcal{D}(\tilde{\boldsymbol{y}};\mathcal{K})).\end{aligned} \tag{2}$$

3.3. Encryption $\mathcal{E}(x;\mathcal{K})$ and Decryption $\mathcal{D}(x;\mathcal{K})$

A natural question arises: what defines a good pair of encryption $\mathcal{E}(\boldsymbol{x};\mathcal{K})$ and decryption $\mathcal{D}(\boldsymbol{x};\mathcal{K})$?

Non-Invertibility. The encryption function should be *non-invertible* without the private key. Otherwise, the cloud can very easily recover the input from the *encrypted intermediate input* $\tilde{\boldsymbol{x}}$:

$$\boldsymbol{x} = (\mathcal{M}^{\text{E}}_{\text{pre}})^{-1}(\mathcal{E}^{-1}(\tilde{\boldsymbol{x}})), \tag{3}$$

where $\mathcal{E}^{-1}$ is the inverse function of the encryption, and $(\mathcal{M}^{\text{E}}_{\text{pre}})^{-1}$ is the inverse preprocess model (approximated by neural networks). The output $\boldsymbol{y}$ can be recovered by rerunning the inference $\mathcal{M}(\boldsymbol{x})$.

Compatibility. Let us consider an extreme case where we leverage a complicated cryptographic hash function (such as MD5) as our encryption. It is indeed secure as it is empirically not invertible; however, it breaks the *continuity* and *locality* of the input data, which are the foundations for the convolution to be effective. Therefore, it is very critical that the designed encryption and decryption functions are *compatible* with the model so that it will not hurt the accuracy.

* Numbers are adopted from the authors' oral presentation at ICLR'19.

Efficiency. Both encryption and decryption functions should be very efficient to compute as they need to be computed locally on the edge so that it can protect the privacy of the data sent to the cloud.

We design a pair of encryption and decryption functions that satisfy all three properties. We first start with the case where the main model on the cloud $\mathcal{M}^{\text{C}}_{\text{main}}$ is a single-layer convolution (without the activation function). As the convolution is linear, the main model, in this case, is a linear function.

Motivated by this linearity, we propose to *linearly combine multiple data* together to protect the data privacy. Concretely, given a sequence of input data with size of n: $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$, the edge first runs the preprocess model on the edge to obtain their intermediate inputs: $\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2, \ldots, \tilde{\boldsymbol{x}}_n$. Then, for each data $\boldsymbol{x}_k$, instead of transmitting its $\tilde{\boldsymbol{x}}_k$ to the cloud, the edge sends the *encrypted intermediate input* $\tilde{\boldsymbol{x}}^{\text{C}}_k$, which is a *linear combination* of the intermediate inputs of the data $\boldsymbol{x}_k$ and other data $\boldsymbol{x}_i$'s:

$$\tilde{\boldsymbol{x}}^{\text{C}}_k = \mathcal{E}(\tilde{\boldsymbol{x}}_k;\mathcal{K}) = \sum_{i=1}^{n} \alpha_{ki} \times \tilde{\boldsymbol{x}}_i, \tag{4}$$

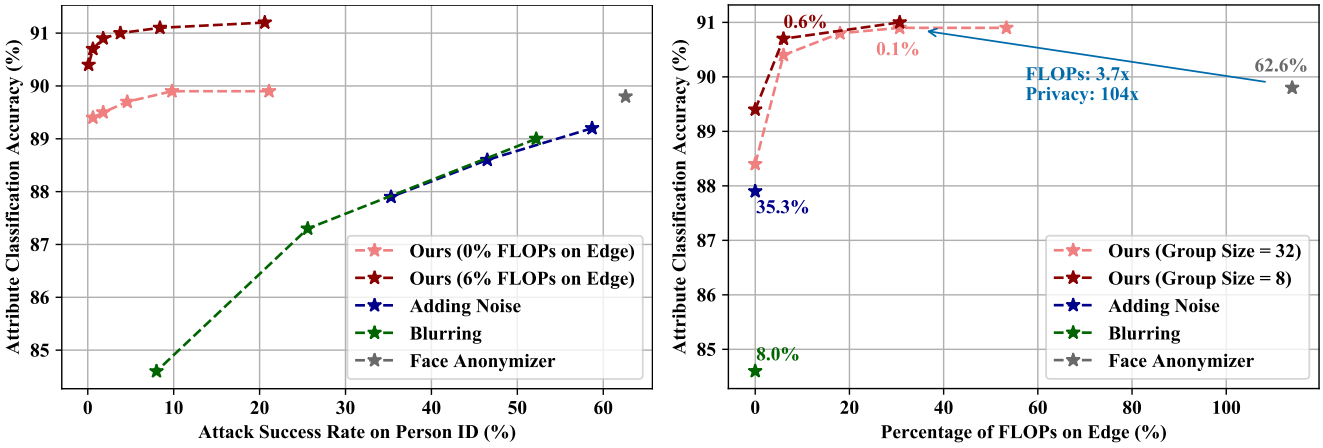
where α_{ki}'s are *random* coefficients, which serve as the private key $\mathcal{K}$ of the user. For each data $\boldsymbol{x}_k$, it is hard for the cloud to recover $\tilde{\boldsymbol{x}}_k$ from its encrypted intermediate input $\tilde{\boldsymbol{x}}^{\text{C}}_k$. This is because the cloud does not know what the other $\tilde{\boldsymbol{x}}_i$'s, neither does it know how these $\tilde{\boldsymbol{x}}_i$'s are combined together (determined by the coefficients α_{ki}, which the cloud does not have access to).

After the cloud runs the inference of the main model on the encrypted intermediate input $\tilde{\boldsymbol{x}}^{\text{C}}_k$,

$$\begin{aligned}\tilde{\boldsymbol{y}}^{\text{C}}_k &= \mathcal{M}^{\text{C}}_{\text{main}}(\tilde{\boldsymbol{x}}^{\text{C}}_k) = \mathcal{M}^{\text{C}}_{\text{main}}\Big(\sum_{i=1}^{n} \alpha_{ki} \times \tilde{\boldsymbol{x}}_i\Big)\\ &= \sum_{i=1}^{n} \alpha_{ki} \times \mathcal{M}^{\text{C}}_{\text{main}}(\tilde{\boldsymbol{x}}_i) = \sum_{i=1}^{n} \alpha_{ki} \times \tilde{\boldsymbol{y}}_i\end{aligned} \tag{5}$$

	Efficiency (↓ is better)		**Accuracy** (↑ is better)		**Privacy** (↓ is better)	
	Edge Params	Edge FLOPs	Valid Acc.	Test Acc.	Person ID	Face Attrs.
Baseline (all on Edge)	11.21M	1.50G	91.6%	91.0%	0.1%	50.0%
Baseline (all on Cloud)	0	0	91.6%	91.0%	85.5%	79.3%
Adding Noise $\mathcal{N}(0, 4)$ [43]	0	0	89.2%	88.6%	46.5%	73.1%
Adding Noise $\mathcal{N}(0, 8)$ [43]	0	0	88.5%	87.9%	35.3%	70.8%
Blurring (16×16) [43, 45]	0	0	89.6%	89.0%	52.2%	73.1%
Blurring (8×8) [43, 45]	0	0	87.9%	87.3%	25.6%	68.7%
Face Anonymizer [43]	11.38M	47.13G	90.5%	89.8%	62.6%	76.3%
FaceMix (Ours) ($S_G = 8$, $N_{pre} = 1$)	**0.05M**	**0.09G**	91.2%	90.7%	**0.6%**	**51.5%**
FaceMix (Ours) ($S_G = 8$, $N_{pre} = 2$)	0.12M	0.28G	91.2%	90.7%	**0.6%**	**51.6%**
FaceMix (Ours) ($S_G = 8$, $N_{pre} = 3$)	0.20M	0.46G	**91.4%**	**91.0%**	**0.6%**	**51.5%**

Table 2: Privacy-preserving facial attribute classification on CelebA. The red entries are unsatisfactory (efficiency: the fewer FLOPs the better; privacy: the lower attack success rate the better). We require fewer computations on the edge, while maintaining higher accuracy and lower attack success rate.



(a) Accuracy *vs*. Privacy

(b) Accuracy *vs*. Efficiency

Figure 4: Our framework provides much better trade-offs among efficiency, accuracy and privacy. In 4b, the number next to each framework represents the attack success rate (the lower the better).

will be transmitted back to the edge. Here, each response $\tilde{\boldsymbol{y}}_k^{\mathrm{C}}$ is essentially a *linear combination* of the intermediate outputs of the data $\boldsymbol{x}_k$ and other data $\boldsymbol{x}_i$'s. After receiving all the responses from the cloud, the edge can then solve the intermediate output for each data $\boldsymbol{x}_k$:

$$\tilde{\boldsymbol{y}}_k = \mathcal{D}(\tilde{\boldsymbol{y}}_k^{\mathrm{C}}; \mathcal{K}) = \sum_{i=1}^{n} \beta_{ki} \times \tilde{\boldsymbol{y}}_i^{\mathrm{C}}, \tag{6}$$

where $\mathcal{B} = [\beta_{ki}]$ is the inverse matrix of $\mathcal{A}$. With these intermediate outputs $\tilde{\boldsymbol{y}}_k$'s, the edge finally runs the inference of the postprocess model to obtain the final output $\boldsymbol{y}_k$ for each data $\boldsymbol{x}_k$. This design satisfies the three desired properties. First, the encryption is *non-invertible* because the cloud does not have access to the data being combined and does not know how they are combined together. Second, the encryption and decryption functions are *compatible* with the model as they take advantage of the linearity and therefore will not introduce any errors in this linear case. Third, both encryption and decryption are very efficient to compute as they are only composed of a few additions.

3.4. Encryption-Aware Training

In the general case, the main model is usually composed of multiple layers and is therefore non-linear. One solution (as in Tramer *et al*. [50]) is to partition the model so that the cloud only needs to process the linear layers; however, this will introduce significant communication overhead (see Table 1). In this paper, we instead think outside the box

and *implicitly encourage the main model to be as linear as possible* by the *encryption-aware training*. Specifically, we train the model along with our encryption and decryption functions: at each training iteration, we first sample a set of random coefficients; we then forward the model through the encryption and decryption; we finally optimize the model in an end-to-end manner (since our encryption and decryption are both differentiable).

In order to encourage the linearity of the model, we need to carefully choose the random coefficient matrix $\mathcal{A}$ from specific groups. For simplicity, we take the fully connected layer (convolution layer can be regarded as multiple fully connected layers for local features) as an example. Given a fully connected layer $\mathcal{M}$ with weight $\boldsymbol{W} \in \mathbb{R}^{n\times m}$, an input $\boldsymbol{x} \in \mathbb{R}^{B\times n}$ and an output $\tilde{\boldsymbol{y}} \in \mathbb{R}^{B\times m}$, the forwarding of the network and the encrypted network can be represented as:

$$\tilde{\boldsymbol{y}} = \boldsymbol{x}\boldsymbol{W}; \quad \tilde{\boldsymbol{y}}^{\mathrm{C}} = \mathcal{A}\boldsymbol{x}\boldsymbol{W}^{\mathrm{C}} \tag{7}$$

When the gradient of $\tilde{\boldsymbol{y}}$ is calculated as $\delta\tilde{\boldsymbol{y}}$, the gradient of $\boldsymbol{W}$ and $\boldsymbol{x}$ are respectively denoted as:

$$\delta\boldsymbol{W} = \boldsymbol{x}^T\delta\tilde{\boldsymbol{y}}; \quad \delta\boldsymbol{x} = \delta\tilde{\boldsymbol{y}}\boldsymbol{W}^T \tag{8}$$

For the encrypted version, the gradients of the weight and the input will be:

$$\begin{aligned}\delta\boldsymbol{W}^{\mathrm{C}} &= \boldsymbol{x}^T(\mathcal{A}^T\mathcal{A})\delta\tilde{\boldsymbol{y}};\\ \delta\boldsymbol{x} &= \delta\tilde{\boldsymbol{y}}(\boldsymbol{W}^{\mathrm{C}})^T\end{aligned} \tag{9}$$

Comparing the Equation 8 and Equation 9, the coefficient matrix should satisfy the equation, $\mathcal{A}^T\mathcal{A} = \mathcal{I}$ to ensure $\delta\boldsymbol{W}^{\mathrm{C}} = \delta\boldsymbol{W}$. In orther words, $\mathcal{A}$ should be randomly chosen from the orthogonal matrix group. For the non-linear model $\mathcal{M}^{\mathrm{C}}_{\mathrm{main}}$ with activation functions, the orthogonal coefficient matrix encourages the model to approximate the linearity so that $\mathcal{M}^{\mathrm{C}}_{\mathrm{main}}(\mathcal{A}\boldsymbol{x}) \approx \mathcal{A}\mathcal{M}^{\mathrm{C}}_{\mathrm{main}}(\boldsymbol{x})$.

3.5. Inter-Group Shuffling

Another challenge of encrypting the inputs with randomized linear combination is that Equation 5 does not hold very well for very large n's in the non-linear case. To this, we split the input data into groups with different visual images of size S_{G}, and we apply our encryption and decryption functions to these smaller groups. This is equivalent to reducing the size of n to achieve better approximations.

In order to achieve better protection for the privacy of the data, the clients can shuffle the encrypted intermediate inputs across the groups before sending the data to the cloud, which will not affect the throughput of the inference process. In that case, the cloud will not be able to relate data from the same group, which increases the ambiguity.

4. Experiments

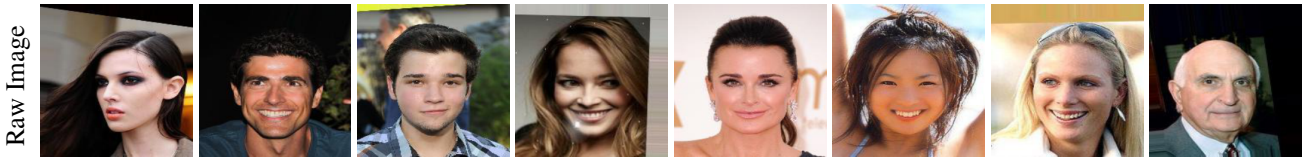
We conduct experiments on two popular facial attribute classification benchmarks to demonstrate the consistent effectiveness of our framework and design three attack methods to evaluate the preservation of privacy.

Setups. We conduct experiments on two benchmarks: CelebA [36] and LFWA [36], which are large-scale facial attribute datasets. CelebA contains more than 200,000 celebrity images of more than 10,000 identities. LFWA has more than 10,000 images of more than 5,000 identities. For the two datasets, each image is annotated with 40 attributes, some of which are privacy-sensitive (*e.g.*, age, and gender). With Han *et al.* [11] as baseline and ResNet-18 [15] as backbone, we train the models for 100 epochs for CelebA and 600 epochs for LFWA using ADAM [27] with weight decay of 10^{-4}. We evaluate the average classification accuracy over 40 attributes on both datasets. We run all experiment for four times and report the average results.

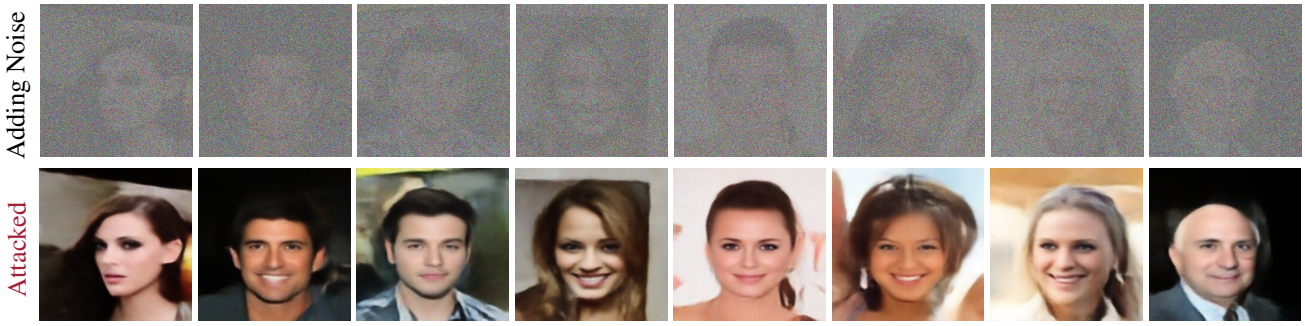
Metrics. Previous work [31] uses the pixel-wise reconstruction error (*e.g.*, PSNR) as a measurement for privacy: lower PSNR means worse reconstruction and better privacy. However, reconstruction error is not directly correlated to privacy: *e.g.*, a small distortion will lead to a large reconstruction error, but we can still identify the person from the image with small distortion. Instead, we propose to train an attack model and use the attack success rate as the evaluation metrics for privacy: a lower success rate indicates better privacy. We consider both the *person identity* and the *facial attributes* as the private information under attack. Concretely, we train two separate attack models (with similar architecture as the baseline model) that takes the *encrypted intermediate input* $\tilde{\boldsymbol{x}}^{\mathrm{C}}_k$ and predicts the person identity and facial attributes corresponding to the input data $\boldsymbol{x}_k$. We report the class-balanced attack success rate for the facial attributes (lower the better).

Baselines. We compare our framework with two handcrafted approaches (*i.e.*, adding noise and blurring) and one learning-based adversarial obfuscator (*i.e.*, face anonymizer) [43, 44]. As for our framework, we investigate different group sizes S_{G} (*i.e.*, number of images to be linearly combined) and different model partitions (how many computations to be offloaded to the cloud): the preprocess model contains N_{pre} convolution blocks, and the postprocess model contains the final layer only.

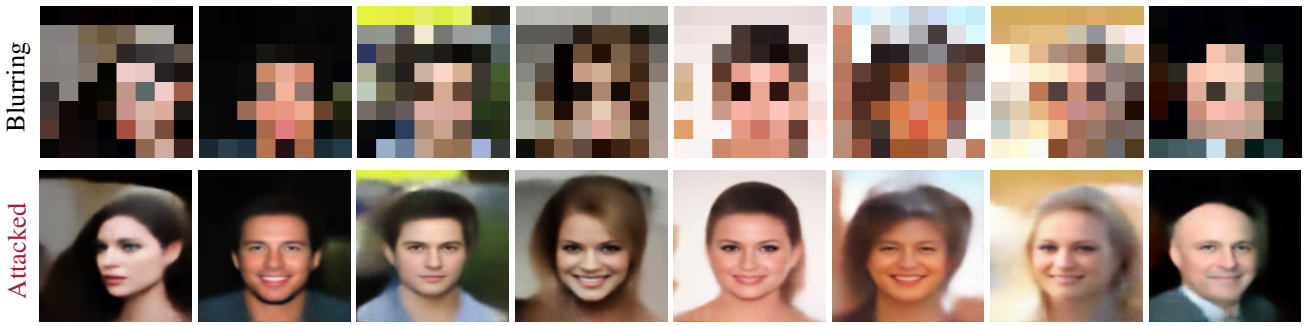
Recovering Face. We designed a GAN-based attack model to recovers the raw images from the encrypted inputs. Since the cloud cannot relate the data from the same group, as mentioned in Section 3.5, the attacker has to reconstruct all the faces that are combined together from the



(a) Raw images of human faces from the test set of the CelebA dataset.



(b) Adding Noise $\mathcal{N}(0, 8)$ [43]. The attack model can recover the faces with many details and recognizable personal identification.



(c) Blurring (8×8) [43, 45]. Artifacts appears in the reconstructed faces but many sensitive face attributes still retains, e.g. race and baldness.

(d) FaceMix (Ours). The attack model fails to separate the information from the mixed faces and the privacy is well preserved.

Figure 5: Defense and attack on th CelebA dataset. It is much harder for the attack model to reconstruct the raw data from our encrypted input sent to the cloud than adding noise and blurring. For each method, 1st row is the encrypted images applied the defense method on the raw image; 2nd row is the image recovered by the GAN-based attack model.

encrypted intermediate input x_k:

$$\tilde{\boldsymbol{x}}_k^{\mathrm{C}} = \sum_{i=1}^{n} \alpha_{ki} \times \tilde{\boldsymbol{x}}_i = \sum_{i=1}^{n} \alpha_{ki} \times \mathcal{M}_{\mathrm{pre}}^{\mathrm{E}}(\boldsymbol{x}_i). \tag{10}$$

We use Pix2Pix [21] as our attack model to recover the raw input image. Instead of only recovering a single input image $\boldsymbol{x}_k$. We train the model to recover all inputs $\boldsymbol{x}_i$'s from the encrypted intermediate input due to the ambiguity: *i.e.*, the model does not know which $\boldsymbol{x}_i$ corresponds to the desired $\boldsymbol{x}_k$. During training, we use the Chamfer distance as the optimization objective, since the ordering of the outputs does not matter:

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{y}) = \sum_k \min_i \|\boldsymbol{x}_k - \boldsymbol{y}_i\|_1 + \sum_k \min_i \|\boldsymbol{y}_k - \boldsymbol{x}_i\|_1,$$

where $\boldsymbol{x}$ and $\boldsymbol{y}$ are the original input images and the model's reconstructions, respectively.

We train a GAN-based attack model for each of the encryption methods, including adding noise, blurring, and our

	Efficiency (↓ is better)		**Accuracy** (↑ is better)		**Privacy** (↓ is better)	
	Edge Params	Edge FLOPs	Test Acc.	Bal. Acc.	Face Recon.	Face Attrs.
Baseline (all on Edge)	11.21M	1.50G	91.1%	87.1%	-0.56	50.0%
Baseline (all on Cloud)	0	0	91.1%	87.1%	-0.00	87.1%
Adding Noise $\mathcal{N}(0, 4)$ [43]	0	0	88.5%	82.5%	-0.03	82.7%
Adding Noise $\mathcal{N}(0, 8)$ [43]	0	0	87.7%	81.3%	-0.02	81.4%
Blurring (16×16) [43, 45]	0	0	88.8%	83.4%	-0.03	83.6%
Blurring (8×8) [43, 45]	0	0	87.0%	80.6%	-0.07	77.8%
FaceMix (Ours) ($S_G = 8$, $N_{pre} = 1$)	**0.05M**	**0.09G**	90.5%	86.8%	**-0.37***	**50.6%**
FaceMix (Ours) ($S_G = 8$, $N_{pre} = 2$)	0.12M	0.28G	90.7%	86.9%	**-0.37***	**50.7%**
FaceMix (Ours) ($S_G = 8$, $N_{pre} = 3$)	0.20M	0.46G	**90.7%**	**87.1%**	**-0.37***	**50.6%**

Table 3: Privacy-preserving facial attribute classification on LFWA. Bal. Acc. denotes the balanced accuracy on the test set and Face Recon. represents the inverse mean square error of the reconstructed images (GAN-based) with the raw inputs. *The GAN-based attack model is applied on the encrypted input image without the *preprocessing* model for fair comparison.

FaceMix. From Figure 5, the attack model can synthesize realistic faces for all the encryption methods. For adding noise and blurring, the attack model can recover much detailed information, such as smiling and baldness. However, it fails to reconstruct any one of the faces in an encryption group from the encrypted input of our FaceMix. We conjecture that the attack model is simply recovering the average faces as it is too difficult for it to disentangle each face due to the ambiguity.

Results. As in Table 2, compared with hand-crafted approaches, our framework achieves 3% higher accuracy and much better privacy (more than 20× lower attack success rate on person ID). Another interesting observation is that adding large Gaussian noise is not secure for protecting the person identity, which also indicates that the pixel-wise error is indeed not a good privacy metrics as large noise will lead to large pixel-wise error. Compared with adversarial obfuscator [43], our framework achieves higher accuracy and significantly better privacy with two orders of magnitude fewer FLOPs on the edge. This is not surprising, since these obfuscators usually use the encoder-decoder framework and are rather computationally expensive. Apart from the personal identity, our framework can also protect the output privacy, which is quantified by the attack success rate on facial attributes (including personal information such as age and gender). However, previous approaches do not take this into consideration.

Similar conclusions can be drawn from the experiments on the LFWA dataset. As in Table 3, our framework outperforms the hand-crafted approaches on both the accuracy and the privacy, including the error of the reconstruction and the output privacy. Specifically, the reconstruction error is the mean square error between the reconstructed images and the raw inputs. To calculate the reconstruction error for FaceMix, we first match the reconstructed images and the raw images in the same group so that the sum of the mean square error between each pair of images is the minimum. We take the average mean square error of these pairs of images as the reconstruction error.

Trade-offs. In Figure 4, we present the trade-offs between accuracy *vs*. privacy (by changing the group size) and accuracy *vs*. efficiency (by changing the number of layers to execute on the edge). In Figure 4a, the approximation for Equation 5 becomes worse as the group size increases, leading to the accuracy degradation. At the same time, our framework achieves better privacy since the combination of a larger group introduces more ambiguities to the encrypted data for the attacker. In Figure 4b, when more convolution blocks are executed on the edge, more local computation will be required, making the fast and efficient inference more challenging. On the other hand, more layers on the edge means fewer non-linear layers in the cloud, leading to a better approximation for Equation 5.

5. Conclusion

In this paper, we introduce the *privacy-preserving edge-cloud inference* framework, FaceMix, to bring the best of the resource-hungry edge devices and the privacy-invasive cloud servers together for the facial attribute classifications. We propose to delegate most of the model computations to the cloud and carefully design a pair of encryption and decryption functions to protect the privacy of the data transmitted to the cloud. Our framework is efficient, accurate and privacy-preserving: extensive experiments on two facial datasets demonstrate that FaceMix can greatly reduce the local computations on the edge with negligible loss of accuracy and no leakages of private information.

Acknowledgements. We thank MIT Quest for Intelligence, Samsung and Facebook for supporting this research. We thank AWS Machine Learning Research Awards for providing the computation resource.

References

[1] Martín Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep Learning with Differential Privacy. In *CCS*, 2016. 2

[2] Dakshi Agrawal and Charu C Aggarwal. On the Design and Quantification of Privacy Preserving Data Mining Algorithms. In *PODS*, 2001. 2

[3] Rakesh Agrawal and Ramakrishnan Srikant. Privacy-Preserving Data Mining. In *SIGMOD*, 2000. 2

[4] Dmitri Bitouk, Neeraj Kumar, Samreen Dhillon, Peter N Belhumeur, and Shree K Nayar. Face Swapping: Automatically Replacing Faces in Photographs. In *SIGGRAPH*, 2008. 2

[5] Jiawei Chen, Jonathan Wu, Janusz Konrad, and Prakash Ishwar. Semi-Coupled Two-Stream Fusion ConvNets for Action Recognition at Extremely Low Resolutions. In *WACV*, 2017. 2

[6] Edward Chou, Matthew Tan, Cherry Zou, Michelle Guo, Albert Haque, Arnold Milstein, and Li Fei-Fei. Privacy-Preserving Action Recognition for Smart Hospitals using Low-Resolution Depth Images. In *NeurIPS Workshop*, 2018. 2

[7] Matthieu Courbariaux, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized Neural Networks: Training Deep Neural Networks with Weights and Activations Constrained to +1 or -1. *arXiv*, 2016. 2

[8] Cynthia Dwork. Differential Privacy: A Survey of Results. In *TAMC*, 2008. 2

[9] Amir Gholami, Kiseok Kwon, Bichen Wu, Zizheng Tai, Xiangyu Yue, Peter Jin, Sicheng Zhao, and Kurt Keutzer. SqueezeNext: Hardware-Aware Neural Network Design. In *CVPR Workshop*, 2018. 2

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. In *NIPS*, 2014. 2

[11] Hu Han, Anil Jain, Fang Wang, Shiguang Shan, and Xilin Chen. Heterogeneous Face Attribute Estimation: A Deep Multi-Task Learning Approach. *TPAMI*, 2018. 6

[12] Hu Han, Anil K. Jain, Fang Wang, Shiguang Shan, and Xilin Chen. Heterogeneous Face Attribute Estimation: A Deep Multi-Task Learning Approach. *TPAMI*, 2017. 2

[13] Song Han, Huizi Mao, and William Dally. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. In *ICLR*, 2016. 1, 2

[14] Song Han, Jeff Pool, John Tran, and William Dally. Learning both Weights and Connections for Efficient Neural Networks. In *NIPS*, 2015. 2

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *CVPR*, 2016. 6

[16] Yihui He, Ji Lin, Zhijian Liu, Hanrui Wang, Li-Jia Li, and Song Han. AMC: AutoML for Model Compression and Acceleration on Mobile Devices. In *ECCV*, 2018. 2

[17] Yihui He, Xiangyu Zhang, and Jian Sun. Channel Pruning for Accelerating Very Deep Neural Networks. In *ICCV*, 2017. 2

[18] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V Le, and Hartwig Adam. Searching for MobileNetV3. *arXiv*, 2019. 1

[19] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv*, 2017. 1, 2

[20] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William Dally, and Kurt Keutzer. SqueezeNet: AlexNet-Level Accuracy with 50x Fewer Parameters and ¡0.5MB Model Size. *arXiv*, 2016. 2

[21] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei Efros. Image-to-Image Translation with Conditional Adversarial Networks. In *CVPR*, 2017. 7

[22] Vijay S Iyengar. Transforming Data to Satisfy Privacy Constraints. In *KDD*, 2002. 2

[23] Max Jaderberg, Andrea Vedaldi, and Andrew Zisserman. Speeding up Convolutional Neural Networks with Low Rank Expansions. In *BMVC*, 2014. 1

[24] Jungseock Joo, Shuo Wang, and Song-Chun Zhu. Human Attribute Recognition by Rich Appearance Dictionary. In *ICCV*, 2013. 2

[25] Amin Jourabloo, Xi Yin, and Xiaoming Liu. Attribute Preserved Face De-identification. In *ICB*, 2015. 2

[26] Tae-Hoon Kim, Dongmin Kang, Kari Pulli, and Jonghyun Choi. Training with the Invisibles: Obfuscating Images to Share Safely for Learning Visual Recognition Models. *arXiv*, 2019. 3

[27] Diederik Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *ICLR*, 2015. 6

[28] Raghuraman Krishnamoorthi. Quantizing Deep Convolutional Networks for Efficient Inference: A Whitepaper. *arXiv*, 2018. 2

[29] Neeraj Kumar, Alexander C. Berg, Peter N. Belhumeur, and Shree K. Nayar. Describable Visual Attributes for Face Verification and Image Search. *TPAMI*, 2011. 2

[30] Ryan Layne, Tim Hospedales, and Shaogang Gong. Person Re-identification by Attributes. In *BMVC*, 2012. 2

[31] Sam Leroux, Tim Verbelen, Pieter Simoens, and Bart Dhoedt. Privacy Aware Offloading of Deep Neural Networks. In *ICML Workshop*, 2018. 3, 6

[32] Meng Li, Liangzhen Lai, Naveen Suda, Vikas Chandra, and David Z Pan. PrivyNet: A Flexible Framework for Privacy-Preserving Deep Neural Network Training. *arXiv*, 2017. 3

[33] Tao Li and Lei Lin. AnonymousNet: Natural Face De-Identification with Measurable Privacy. In *CVPR Workshop*, 2019. 2

[34] Ji Lin, Liangliang Ren, Jiwen Lu, Jianjiang Feng, and Jie Zhou. Consistent-aware deep learning for person re-identification in a camera network. In *CVPR*, 2017. 2

[35] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning Efficient Convolutional Networks through Network Slimming. In *ICCV*, 2017. 2

[36] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep Learning Face Attributes in the Wild. In *ICCV*, 2015. 2, 6

[37] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design. In *ECCV*, 2018. 2

[38] Elaine M Newton, Latanya Sweeney, and Bradley Malin. Preserving Privacy by De-Identifying Face Images. *TKDE*, 2005. 2

[39] Seong Joon Oh, Rodrigo Benenson, Mario Fritz, and Bernt Schiele. Faceless Person Recognition; Privacy Implications in Social Media. In *ECCV*, 2016. 2

[40] Seong Joon Oh, Mario Fritz, and Bernt Schiele. Adversarial Image Perturbation for Privacy Protection: A Game Theory Perspective. In *ICCV*, 2017. 3

[41] Seyed Ali Osia, Ali Shahin Shamsabadi, Ali Taheri, Kleomenis Katevas, Sina Sajadmanesh, Hamid R Rabiee, Nicholas D Lane, and Hamed Haddadi. A Hybrid Deep Learning Architecture for Privacy-Preserving Mobile Analytics. *TKDD*, 2017. 2, 3, 4

[42] Seyed Ali Osia, Ali Taheri, Ali Shahin Shamsabadi, Kleomenis Katevas, Hamed Haddadi, and Hamid R Rabiee. Deep Private-Feature Extraction. In *TKDE*, 2018. 3, 4

[43] Zhongzheng Ren, Yong Jae Lee, and Michael S Ryoo. Learning to Anonymize Faces for Privacy Preserving Action Detection. In *ECCV*, 2018. 3, 5, 6, 7, 8

[44] Michael S Ryoo, Kiyoon Kim, and Hyun Jong Yang. Extreme Low Resolution Activity Recognition With Multi-Siamese Embedding Learning. In *AAAI*, 2018. 2, 6

[45] Michael S Ryoo, Brandon Rothrock, Charles Fleming, and Hyun Jong Yang. Privacy-Preserving Human Activity Recognition from Extreme Low Resolution. In *AAAI*, 2017. 2, 5, 7, 8

[46] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In *CVPR*, 2018. 1, 2

[47] Reza Shokri and Vitaly Shmatikov. Privacy-Preserving Deep Learning. In *CCS*, 2015. 2

[48] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *ICLR*, 2015. 4

[49] Chi Su, Fan Yang, Shiliang Zhang, Qi Tian, Larry S. Davis, and Wen Gao. Multi-Task Learning with Low Rank Attribute Embedding for Person Re-Identification. In *ICCV*, 2015. 2

[50] Florian Tramèr and Dan Boneh. Slalom: Fast, Verifiable and Private Execution of Neural Networks in Trusted Hardware. In *ICLR*, 2019. 3, 4, 5

[51] Ji Wang, Jianguo Zhang, Weidong Bao, Xiaomin Zhu, Bokai Cao, and Philip S Yu. Not Just Privacy: Improving Performance of Private Deep Learning in Mobile Cloud. In *KDD*, 2018. 3

[52] Kuan Wang, Zhijian Liu, Yujun Lin, Ji Lin, and Song Han. HAQ: Hardware-Aware Automated Quantization with Mixed Precision. In *CVPR*, 2019. 2

[53] Zhenyu Wu, Zhangyang Wang, Zhaowen Wang, and Hailin Jin. Towards Privacy-Preserving Visual Recognition via Adversarial Training: A Pilot Study. In *ECCV*, 2018. 3

[54] Qizhe Xie, Eduard Hovy, Minh-Thang Luong, and Quoc V. Le. Self-training with Noisy Student improves ImageNet classification. In *arXiv*, 2019. 1

[55] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices. In *CVPR*, 2018. 2

[56] Zhanpeng Zhang, Ping Luo, Chen Change Loy, and Xiaoou Tang. Learning Social Relation Traits from Face Images. In *ICCV*, 2015. 2

[57] Chenzhuo Zhu, Song Han, Huizi Mao, and William Dally. Trained Ternary Quantization. In *ICLR*, 2017. 2