DataMix: Efficient Privacy-Preserving Edge-Cloud Inference

Zhijian Liu[1,*], Zhanghao Wu[1,2,*], Chuang Gan[3], Ligeng Zhu[1], and Song Han[1]

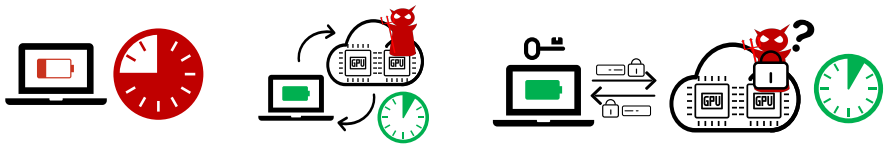
[1] Massachusetts Institute of Technology
[2] Shanghai Jiao Tong University
[3] MIT-IBM Watson AI Lab

Abstract. Deep neural networks are widely deployed on edge devices (*e.g.*, for computer vision and speech recognition). 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 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, DataMix. We off-load the majority of the computations to the cloud and leverage a pair of mixing and de-mixing operation, inspired by *mixup*, to protect the privacy of the data transmitted to the cloud. Our framework has three advantages. First, it is *privacy-preserving* as the mixing cannot be inverted without the user's private mixing coefficients. Second, our framework is *accuracy-preserving* because our framework takes advantage of the space spanned by images, and we train the model in a mixing-aware manner to maintain accuracy. Third, our solution is *efficient* on the edge since the majority of the workload is delegated to the cloud, and our mixing and de-mixing 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 computer vision and speech recognition 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

The high performance and superior accuracy of deep neural networks always comes at the expense of larger model size and more computations. Meanwhile, large models are difficult to be deployed on resource-constrained edge devices (such as mobile phones, self-driving cars and smart speakers): mobile applications interact with the users and require low latency, 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 [23, 47] or accelerate the

∗ indicates equal contributions; order determined by a coin toss.



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

Fig. 1. 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.

existing models by compression [27, 17, 52]. 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 [22] can only achieve 75% of top-1 accuracy on ImageNet, which is still 10% lower than the best performances [59].

In contrast, 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 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. User's data, 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, and health status). Therefore, privacy-preserving inference is of critical importance.

This paper presents a novel perspective to tackle this challenge, in Figure 1. We introduce the *privacy-preserving edge-cloud inference* framework, DataMix, 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. Inspired by *mixup* [60], we design the mixing and de-mixing operation for the privacy of the data transmitted to the cloud and train the model in an mixing-aware manner to maintain the accuracy. Our framework is a general method for cloud-edge inference and can be applied to multiple modalities. We evaluate our proposed framework on multiple tasks on two modalities including facial attribute classification and keyword spotting. 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. We provide an example of attacking for different methods in Figure 2.

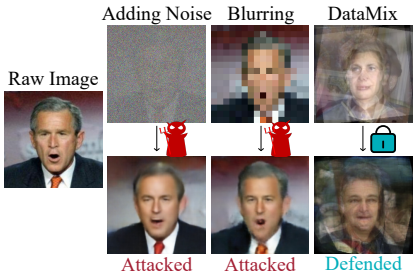


Fig. 2. Adding noise and blurring are not secure: the original image can be recovered by the GAN-based attack model. Our proposed DataMix preserves the privacy better.

2 Related Work

2.1 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 [10, 24], MobileNets [23, 47] ShuffleNets [61, 37], TSM [34] and modifications of Transformers [50, 57]. 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 [18, 17] or the entire channels [21, 35, 20]; others have proposed to quantize the network [8, 62, 30, 55] 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.

2.2 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, 26], model sharing [9, 48, 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 [46, 6, 45, 7]. As for the face-related tasks, researchers have introduced various face de-identification methods to help protect the privacy [38, 4, 28, 39, 33]. The k-anonymity [51, 12, 15, 14, 13] methods are also proposed to protect the information in the data by averaging k closest samples, but it does not take model inference into consideration. However, most of them either require much computation or compromise to the model accuracy degradation.

Inspired by the generative adversarial networks (GANs) [11], 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 manner [40,

	Baseline (cloud)	Osia *et al.* [41, 42]	Tramer *et al.* [53]	DataMix (Ours)	
Computation (on edge)	0%	93%	∼1%*	1%	13%
Transmission size	0.6 MB	0.4 MB	86.5 MB	12.3 MB	3.1 MB
Transmission time	0.4 sec	0.3 sec	33.1 sec	6.5 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*. 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. *Numbers are adopted from the authors' oral presentation at ICLR'19.

44, 31, 58, 29, 43, 5]. 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 [44].

2.3 Hybrid Edge-Cloud Inference

There are several preliminary attempts [41, 42, 32, 54] 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 age can lead to some ageist behaviors, and it therefore should be equally important as the input.

Recently, Tramer *et al.* [53] 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. Specifically, all benchmarks in the table are conducted on VGG-16 [49] 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 network delay of 25 ms. The transmission time

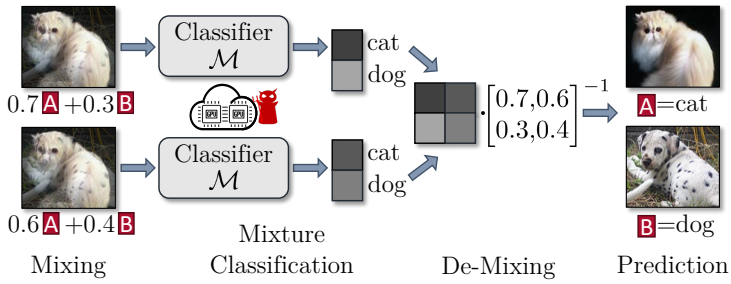


Fig. 3. A motivating example for privacy-preserving inference with mixing. The classifier trained with *mixup* takes mixtures of images and outputs mixed predictions. We can then solve the correct labels from the outputs. Though having the mixtures and the parameter weights of the classifier, the cloud cannot recover the private information without the private coefficients owned by users for mixing.

can be calculated by Equation 1.

$$\mathrm{T} = \frac{\text{input feature}}{\text{upload bandwidth}} + \frac{\text{output feature}}{\text{download bandwidth}} + 2 \times \text{network latency} \quad (1)$$

3 A Motivating Example

Mixup [60] is a general training technique for neural networks. It encourages the model to behave linearly on the mix of training examples to smooth the decision boundaries among classes for generalization. That is to say, neural networks can not only learn from the raw images but also the space spanned by two random images from the distribution of images.

As shown in Figure 3, the two raw images of a cat $\boldsymbol{A}$ and a dog $\boldsymbol{B}$ are mixed with a pair of coefficients $\boldsymbol{c} = [0.7, 0.3]^T$. We denote the mixture as:

$$\boldsymbol{m}(\boldsymbol{c}) = [\boldsymbol{A}, \boldsymbol{B}] \cdot \boldsymbol{c} = 0.7\boldsymbol{A} + 0.3\boldsymbol{B}.$$

When fed with the mixed images, a neural network trained for the dataset with *mixup* can output the mixed probabilities $\tilde{\boldsymbol{y}}_{\boldsymbol{m}(\boldsymbol{c})}$ for the classes with the same coefficients $\boldsymbol{c}$, *i.e.*

$$\tilde{\boldsymbol{y}}_{\boldsymbol{m}(\boldsymbol{c})} = [\tilde{y}_{\boldsymbol{A}}, \tilde{y}_{\boldsymbol{B}}] \cdot \boldsymbol{c}.$$

The case will be more interesting, when we introduce another mixture of the same two images with a different pair of coefficients $\boldsymbol{c'} = [0.6, 0.4]^T$. The new mixture is denoted by $\boldsymbol{m}(\boldsymbol{c'})$. The model output for the $\boldsymbol{m}(\boldsymbol{c'})$ provides another mixed probabilities $\tilde{\boldsymbol{y}}_{\boldsymbol{m}(\boldsymbol{c'})}$ with the coefficients $\boldsymbol{c'}$. Aware of the exact values of the two pairs of coefficients $\mathcal{C} = [\boldsymbol{c}; \boldsymbol{c'}]$, we can solve (or de-mix) the correct prediction from the $[\tilde{\boldsymbol{y}}_{\boldsymbol{m}(\boldsymbol{c})}, \tilde{\boldsymbol{y}}_{\boldsymbol{m}(\boldsymbol{c'})}]$ for the each raw inputs, where

$$[\tilde{\boldsymbol{y}}_{\boldsymbol{A}}, \tilde{\boldsymbol{y}}_{\boldsymbol{B}}] = [\tilde{\boldsymbol{y}}_{\boldsymbol{m}(\boldsymbol{c})}, \tilde{\boldsymbol{y}}_{\boldsymbol{m}(\boldsymbol{c'})}] \cdot \mathcal{C}^{-1}.$$

The mixing and de-mixing operation lead to an effective and efficient protocol for privacy-preserving inference on the cloud. Using the operation, we can offload

the model trained with *mixup* to the cloud and only transmit mixed inputs and outputs for model inference. Since both the attackers on the network and cloud cannot access the mixing coefficients, they are not able to recover the original private data from the mixed inputs and outputs transmitted. That makes the method an excellent privacy-preserving operation. On the other hand, the operation only contains a small amount of computation, *i.e.* a few additions, making the method efficient enough for the resource-constrained edge devices. That enables computational-intensive neural network inference for privacy-sensitive situations on mobiles and IoTs.

However, though mixing and de-mixing operation protect the original data from being recovered, some of the private information can still be recognized, *e.g.* the color of the cat in Figure 3. We extend the method to our DataMix for a general privacy-preserving framework.

4 Method

In this section, we will extend the privacy-preserving inference method in Section 3. We first describe the problem setting; we then extend the mixing and de-mixing for DataMix with larger group size and intermediate features processing; we finally analyze the design choice and provide some techniques for practice.

4.1 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 both the attackers on the internet and cloud want to recover users' private information from each request. We also assume that the attackers and the cloud do not relate multiple requests. For the users, they have multiple inputs to be inferred, *e.g.* image classification in smart albums, and automatic medical analysis in hospitals with many patients. In another situation, the users can have a pool of images, from which they can sample some random images to protect privacy of the real image for inference. Our goal is to develop an effective and efficient method so that attackers cannot recover users' data and the inferred results from users' requests to the cloud.

4.2 DataMix

The motivating example provides a prototype for privacy-preserving inference on the cloud. With the classifier trained by *mixup*, users can send the privacy-preserving mixtures to the cloud for inference. Practically, we further extend it as DataMix for more secure and better performance.

Larger Group for Mixing. Mixing the images with random (private) coefficients can hide information in the data. That is because the entropy of the mixed data increases when pixels from the two images are combined. If we enlarge the group of images for mixing, the entropy will become larger as more information is combined, and therefore better preserve the privacy. It is also the case for the output predictions.

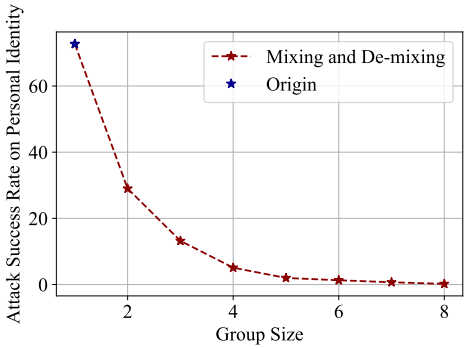


Fig. 4. The influence of the group size of mixing for data privacy on CelebA dataset. The personal identity accuracy indicates the how much information can an attacker get from the data transmitted to the cloud. When the group size increases, the privacy of the data are better preserved.

We examine the intuition on figures of human faces from CelebA dataset. We trained attack models for the personal identity for the raw images and mixtures with different group sizes, separately. In Figure 4, we observe that by mixing two images, the attack success rate will be reduced by 43.7%. When the group size S_{G} of image mixing increases, the success rate further decreases, *i.e.* better preserves the privacy. In our DataMix, rather than merely using a group size of 2 for privacy protection, we apply a larger group for mixing, *e.g.* $S_{\mathrm{G}} = 8$.

Intermediate Features Mixing. We also extend the mixing and de-mixing operation to the intermediate features, *e.g.* the outputs of the first convolution layer.

In the motivating example (Section 3), we apply mixing and de-mixing directly to the raw images and outputs of the classifier so that all the attackers on the network and the cloud can only access to the mixed data. The *mixup* training technique can be regarded as a regularization term for the classifier that encourages the model to perform well on the space spanned by two images from the dataset. After we enlarge the group size S_{G} for mixing, the space becomes more complicated. To improve the performance of the classifier on the mixtures, instead of applying mixing to the raw data, we adopt the operation on the intermediate features. The model can then learn to leverage the first several layers to re-project raw data to another space that provides better mixtures for the suffix layers to model the patterns.

Formalization. To formalize our method, we illustrate our method in Figure 5. We partition the neural network model $\mathcal{M}$ sequentially into three parts:

$$\mathcal{M} = \mathcal{M}^{\mathrm{E}}_{\mathrm{post}} \circ \mathcal{M}^{\mathrm{C}}_{\mathrm{main}} \circ \mathcal{M}^{\mathrm{E}}_{\mathrm{pre}}, \tag{2}$$

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.

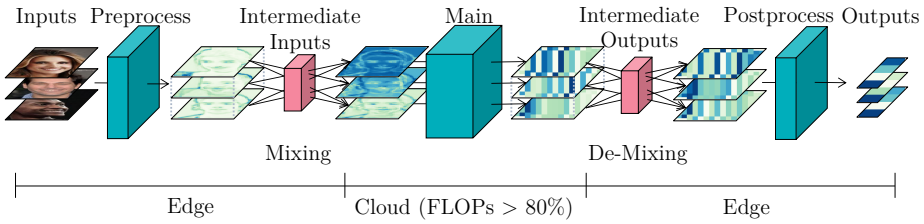


Fig. 5. Partitioning the neural network into three parts, our framework leverages the mixing and de-mixing operation to protect the privacy of data transmitted to the cloud.

During the inference, the edge first runs the preprocess model on the *raw inputs* and sends its output (*i.e.*, *intermediate input*) to the cloud; then, the cloud runs the main model and transmits its output (*i.e.*, *intermediate output*) back to the edge; finally, the edge executes the postprocess model to obtain the *final output*. Throughout the whole procedure, the attackers can only access to the intermediate input instead of the raw input data.

However, without mixing 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 intermediate inputs. For the output data, the cloud can simply rerun the inference of the postprocess model on intermediate outputs since the cloud has access to the weights.

To solve these issues, as previously mentioned, we apply the mixing after the preprocess model and de-mixing before the postprocess model on the intermediate features (inputs and outputs). As mentioned above, the preprocess model is used as the projection of the raw image to the space that is more mixing-friendly, and the postprocess model works similarly for the outputs. The mixing and de-mixing operation are both computed on the edge so that only the mixed data are exposed to public, protecting the privacy of original data.

Mixing-Aware Training. With intermediate feature mixing, we design a mixing-aware training process to improve the model performance on the mixture. While training, a batch of training data is firstly fed into the preprocess model; we then mix images in each group with coefficients randomly sampled from orthogonal matrix group; the main model takes the mixtures and generates intermediate outputs; after de-mixing, we feed the outputs to the postprocess model and calculate the loss of outputs with the correct label for each image. The whole model, including $\mathcal{M}^{\mathrm{E}}_{\mathrm{pre}}$, $\mathcal{M}^{\mathrm{C}}_{\mathrm{main}}$ and $\mathcal{M}^{\mathrm{E}}_{\mathrm{post}}$, is updated with the gradients.

Inter-Group Shuffling. In order to achieve better protection for the privacy of the data, the clients can also shuffle the mixed 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 further increases the ambiguity.

4.3 Design Analysis

We analyze several properties of our DataMix to be a good privacy-preserving method for model inference on the cloud.

Non-Invertibility. The private-preserving method for the data transmitted to the cloud should be *non-invertible* without the private keys. Otherwise, the cloud can very easily recover the raw input $\boldsymbol{x}_{\text{raw}}$ from the transmitted input $\boldsymbol{x}_{\text{in}}$:

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

where $\mathcal{E}^{-1}$ is the inverse function of the mixing for the transmitted data, 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 $\mathcal{M}(\boldsymbol{x})$.

In our DataMix, the coefficients are randomly generated by the users and kept private. For mixed images, one of the possible attacking methods is to average a large set of mixtures that contain the same raw image (not easy to collect without knowledge of what and how the images are mixed). In that way, when the images are i.i.d. and have a mean of 0, the average of the mixtures will be equal to the same raw image in expectation. However, the method is applicable only for training, where one image will appear in many mixtures with different groups of images. Since our DataMix focuses on inference, an image is only mixed with a same group of images, and the average of these mixtures is still a mixture with unknown coefficients (not close to zero) for the attackers. Without the coefficients, our DataMix is non-invertible because the attackers do not have access to the data being mixed and do not know how they are mixed.

Compatibility. Let us consider an extreme case where we leverage a complicated cryptographic hash function (such as MD5) as our encryption for the transmitted data. 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. Our DataMix encourages the model to make inference on the space spanned by inputs. As shown in *mixup* [60], the space is compatible with neural networks, when $S_{\text{G}} = 2$. We will provide extensive experiments for larger group size S_{G} in the next section, where our mixing and de-mixing operation also only causes negligible accuracy degradation for larger S_{G}.

Efficiency. The privacy-preserving inference on the cloud should also be very efficient to compute locally on the edge while protecting the privacy of the data sent to the cloud. The mixing and de-mixing operation is only composed of a few additions and very efficient. The computation on the edge is still small, even when we place some of the layers for the preprocess and postprocess model on the edge.

5 Experiments

We conduct experiments on two modalities, facial attribute classification and keyword spotting, to demonstrate the consistent effectiveness of our framework.

	Efficiency(↓better)		**Accuracy**(↑better)		**Privacy**(↓better)	
	Params	FLOPs	Valid	Test	ID	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 $\mathcal{N}(0, 4)$ [44]	0	0	89.2%	88.6%	46.5%	73.1%
Adding Noise $\mathcal{N}(0, 8)$ [44]	0	0	88.5%	87.9%	35.3%	70.8%
Blurring (16×16) [44, 46]	0	0	89.6%	89.0%	52.2%	73.1%
Blurring (8×8) [44, 46]	0	0	87.9%	87.3%	25.6%	68.7%
Face Anonymizer [44]	11.38M	47.13G	90.5%	89.8%	62.6%	76.3%
DataMix (Ours) (N_{pre}=1)	**0.05M**	**0.09G**	91.2%	90.7%	**0.6**%	**51.5**%
DataMix (Ours) (N_{pre}=2)	0.12M	0.28G	91.2%	90.7%	**0.6**%	**51.6**%
DataMix (Ours) (N_{pre}=3)	0.20M	0.46G	**91.4**%	**91.0**%	**0.6**%	**51.5**%

Table 2. Privacy-preserving facial attribute classification on CelebA. All of our DataMix have $S_{\text{G}} = 8$. 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.

5.1 Computer Vision: Facial Attribute Classification

We test our framework on two large-scale facial attribute classification benchmarks, CelebA [36] and LFWA [36], and design three attack methods to evaluate the preservation of privacy.

Setups. 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). With Han *et al.* [16] as baseline and ResNet-18 [19] as backbone, we train the models for 100 epochs for CelebA and 600 epochs for LFWA using SGD with weight decay of 10^{-4}. We also decay the learning rate with cosine annealing when training. 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 for each privacy-preserving methods (with

	Efficiency(↓better)		**Accuracy**(↑better)		**Privacy**(↓better)	
	Params	FLOPs	Test	Bal.	Recon.	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 $\mathcal{N}(0, 4)$ [44]	0	0	88.5%	82.5%	-0.03	82.7%
Adding Noise $\mathcal{N}(0, 8)$ [44]	0	0	87.7%	81.3%	-0.02	81.4%
Blurring (16×16) [44, 46]	0	0	88.8%	83.4%	-0.03	83.6%
Blurring (8×8) [44, 46]	0	0	87.0%	80.6%	-0.07	77.8%
DataMix (Ours) (N_{pre}=1)	**0.05M**	**0.09G**	90.5%	86.8%	**-0.37***	**50.6**%
DataMix (Ours) (N_{pre}=2)	0.12M	0.28G	90.7%	86.9%	**-0.37***	**50.7**%
DataMix (Ours) (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.* denotes the balanced accuracy on the test set and *Recon.* represents the inverse mean square error of the reconstructed images (GAN-based) with the raw inputs. All of our DataMix have $S_{\text{G}} = 8$. *The GAN-based attack model is applied on the encrypted input image without the *preprocessing* model for fair comparison.

similar architecture as the baseline model) that takes the mixed intermediate input $\tilde{\boldsymbol{x}}_k^{\text{C}}$ 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 and Model Settings. We compare our framework with two hand-crafted approaches (*i.e.*, adding noise and blurring) and one learning-based adversarial obfuscator (*i.e.*, face anonymizer) [44, 45]. As for our framework, we investigate different group sizes S_{G} (*i.e.*, number of images to be mixed) 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 mixed inputs. Since the cloud cannot relate the data from the same group, as mentioned in Section 4.2, the attacker has to reconstruct all the faces $\boldsymbol{x}_k$ from the transmitted mixed intermediate input $\tilde{\boldsymbol{x}}_k^{\text{C}}$:

$$\tilde{\boldsymbol{x}}_k^{\text{C}} = \mathcal{M}_{\text{pre}}^{\text{E}}(\boldsymbol{X}) \cdot \boldsymbol{c}. \tag{4}$$

We use Pix2Pix [25] as our attack model to recover the raw input image. We train the model to recover all inputs $\boldsymbol{x}_k$'s from the mixed intermediate input due to the ambiguity: *i.e.*, the model does not know which $\boldsymbol{x}_k$ corresponds to the desired image. During training, we use the Chamfer distance as the optimization

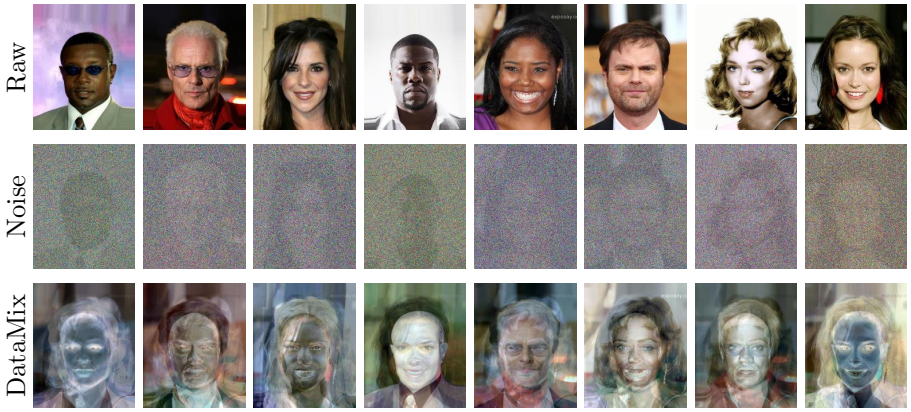


Fig. 6. Qualitative results of defending methods on the CelebA dataset. The images represent the ones accessible to the clouds and attackers with different defending methods. Personal identity and some private attributes like hair style are still recognizable with strong noise. Instead, our DataMix provides much better privacy.

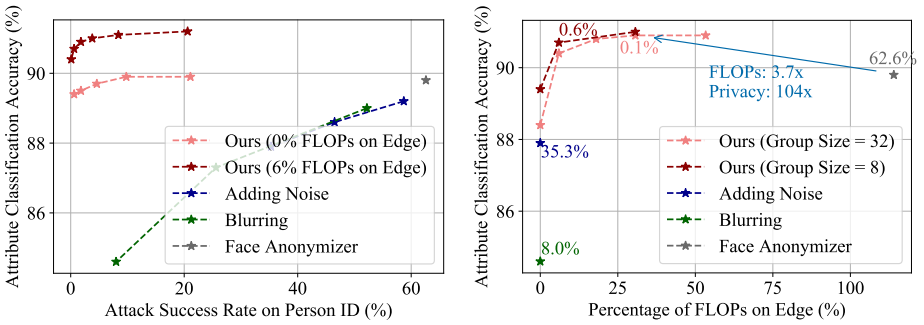
objective, since the ordering of the outputs does not matter:

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

where $\boldsymbol{x}$ and $\boldsymbol{y}$ are the original input images and the model's reconstructions, respectively. We train an attack model for each of the privacy-preserving methods, including adding noise, blurring, and our DataMix.

Results. In Figure 6, we show the images accessible to the cloud and attackers with different defending methods. Our DataMix provides the best protection for personal identity and private facial attributes like hair style. As in Table 2, compared with hand-crafted on CelebA dataset, 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 [44], 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). 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



(a) Accuracy *vs*. Privacy **(b)** Accuracy *vs*. Efficiency

Fig. 7. Our framework provides much better trade-offs among efficiency, accuracy and privacy. In 7b, the number next to each framework represents the attack success rate (the lower the better).

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 DataMix, 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 inverse mean square error of these pairs of images as the reconstruction error. The mixing operation increases the ambiguity of the transmitted data, which prevents the attacker from recovering the original images, giving a much lower inverse mean square error for our DataMix.

Trade-offs. In Figure 7, 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 7a, the space spanned by images becomes more complicated 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 mixed data for the attacker. In Figure 7b, 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 higher capacity for the projection from the raw images to the mixture space that is more friendly for classifier training, leading to a better performance of the main model for the mixing and de-mixing operation.

5.2 Speech: Keyword Spotting

Speech data also contains personal information such as speaker identity and sensitive content. We conduct experiments on Speech Commands [56] to show the generalization of our framework on different modalities.

Setups. The Speech Commands dataset has more than 100,000 utterances from 35 classes. For each utterance, we extract the normalized spectrogram from the waveform at a sampling rate of 16 kHz. We then leverage ResNet-18 [19] as

	Efficiency(↓better)		**Accuracy**(↑better)		**Privacy**(↓better)	
	Params	FLOPs	Val.	Test	ID.	Key.
Baseline (all on Edge)	11.18M	1.27G	96.6%	96.5%	1.2%	3.8%
Baseline (all on Cloud)	0	0	96.6%	96.5%	99.8%	96.5%
Adding Noise $\mathcal{N}(0, 4)$ [44]	0	0	92.8%	91.5%	94.4%	91.5%
Adding Noise $\mathcal{N}(0, 8)$ [44]	0	0	90.3%	89.1%	89.9%	89.1%
DataMix (Ours) (N_{pre}=1)	**0.02M**	**0.03G**	92.5%	92.2%	**19.4%**	**18.8%**
DataMix (Ours) (N_{pre}=2)	0.10M	0.18G	93.5%	93.3%	**15.2%**	**19.4%**
DataMix (Ours) (N_{pre}=3)	0.17M	0.34G	**94.4%**	**94.6%**	**12.7%**	**19.4%**

Table 4. Privacy-preserving keyword spotting on Speech Commands. *ID.* represents the speaker ID on the test set and *Key.* denotes the accuracy of keyword spotting.

baseline, which takes the spectrogram as input and classifies which class each utterance belongs to. We train all models for 100 epochs using SGD with cosine annealing for the learning rate decay.

Metrics. Similar to the previous task, we evaluate how the *speaker identity* and the *output content* are protected using two separate attack models. We then consider the attack success rates of these models as indicators of privacy.

Results. We present the quantitative results in Table 4. Adding noise only improves the privacy a little bit, but the accuracy degradation is significant. This is because large Gaussian noise will also weaken the capability of the model to extract effective features from the input and classify the keywords in the noised utterance. In contrast, our framework mixes the examples with different personal identities and contents with private random coefficient, making the data transmitted to the cloud ambiguous and non-invertible.

6 Conclusion

In this paper, we introduce the *privacy-preserving edge-cloud inference* framework, DataMix, to bring the best of the resource-hungry edge devices and the privacy-invasive cloud servers together for the model inference. We propose to delegate most of the model computations to the cloud and carefully design a mixing and de-mixing operation to protect the privacy of the data transmitted to the cloud. Our framework is efficient, accurate and privacy-preserving: extensive experiments on two computer vision datasets and a speech recognition dataset demonstrate that DataMix can greatly reduce the local computations on the edge with negligible loss of accuracy and no leakages of private information.

Acknowledgements. We thank MIT-IBM Watson AI Lab, 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. Abadi, M., Chu, A., Goodfellow, I., McMahan, H.B., Mironov, I., Talwar, K., Zhang, L.: Deep Learning with Differential Privacy. In: CCS (2016)
2. Agrawal, D., Aggarwal, C.C.: On the Design and Quantification of Privacy Preserving Data Mining Algorithms. In: PODS (2001)
3. Agrawal, R., Srikant, R.: Privacy-Preserving Data Mining. In: SIGMOD (2000)
4. Bitouk, D., Kumar, N., Dhillon, S., Belhumeur, P.N., Nayar, S.K.: Face Swapping: Automatically Replacing Faces in Photographs. In: SIGGRAPH (2008)
5. Chen, J., Konrad, J., Ishwar, P.: Vgan-based image representation learning for privacy-preserving facial expression recognition. In: CVPRW (2018)
6. Chen, J., Wu, J., Konrad, J., Ishwar, P.: Semi-Coupled Two-Stream Fusion ConvNets for Action Recognition at Extremely Low Resolutions. In: WACV (2017)
7. Chou, E., Tan, M., Zou, C., Guo, M., Haque, A., Milstein, A., Fei-Fei, L.: Privacy-Preserving Action Recognition for Smart Hospitals using Low-Resolution Depth Images. In: NeurIPS Workshop (2018)
8. Courbariaux, M., Hubara, I., Soudry, D., El-Yaniv, R., Bengio, Y.: Binarized Neural Networks: Training Deep Neural Networks with Weights and Activations Constrained to +1 or -1. arXiv (2016)
9. Dwork, C.: Differential Privacy: A Survey of Results. In: TAMC (2008)
10. Gholami, A., Kwon, K., Wu, B., Tai, Z., Yue, X., Jin, P., Zhao, S., Keutzer, K.: SqueezeNext: Hardware-Aware Neural Network Design. In: CVPR Workshop (2018)
11. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative Adversarial Nets. In: NIPS (2014)
12. Gross, R., Airoldi, E., Malin, B., Sweeney, L.: Integrating utility into face de-identification. In: Privacy Enhancing Technologies (2005)
13. Gross, R., Sweeney, L., Cohn, J., De la Torre, F., Baker, S.: Face de-identification. In: Protecting privacy in video surveillance. Springer (2009)
14. Gross, R., Sweeney, L., De La Torre, F., Baker, S.: Semi-supervised learning of multi-factor models for face de-identification. In: CVPR (2008)
15. Gross, R., Sweeney, L., De la Torre, F., Baker, S.: Model-based face de-identification. In: CVPRW (2006)
16. Han, H., Jain, A., Wang, F., Shan, S., Chen, X.: Heterogeneous Face Attribute Estimation: A Deep Multi-Task Learning Approach. TPAMI (2018)
17. Han, S., Mao, H., Dally, W.: Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. In: ICLR (2016)
18. Han, S., Pool, J., Tran, J., Dally, W.: Learning both Weights and Connections for Efficient Neural Networks. In: NIPS (2015)
19. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition. In: CVPR (2016)
20. He, Y., Lin, J., Liu, Z., Wang, H., Li, L.J., Han, S.: AMC: AutoML for Model Compression and Acceleration on Mobile Devices. In: ECCV (2018)
21. He, Y., Zhang, X., Sun, J.: Channel Pruning for Accelerating Very Deep Neural Networks. In: ICCV (2017)
22. Howard, A., Sandler, M., Chu, G., Chen, L.C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., Le, Q.V., Adam, H.: Searching for MobileNetV3. arXiv (2019)
23. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv (2017)

24. Iandola, F.N., Han, S., Moskewicz, M.W., Ashraf, K., Dally, W., Keutzer, K.: SqueezeNet: AlexNet-Level Accuracy with 50x Fewer Parameters and ¡0.5MB Model Size. arXiv (2016)
25. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.: Image-to-Image Translation with Conditional Adversarial Networks. In: CVPR (2017)
26. Iyengar, V.S.: Transforming Data to Satisfy Privacy Constraints. In: KDD (2002)
27. Jaderberg, M., Vedaldi, A., Zisserman, A.: Speeding up Convolutional Neural Networks with Low Rank Expansions. In: BMVC (2014)
28. Jourabloo, A., Yin, X., Liu, X.: Attribute Preserved Face De-identification. In: ICB (2015)
29. Kim, T.H., Kang, D., Pulli, K., Choi, J.: Training with the Invisibles: Obfuscating Images to Share Safely for Learning Visual Recognition Models. arXiv (2019)
30. Krishnamoorthi, R.: Quantizing Deep Convolutional Networks for Efficient Inference: A Whitepaper. arXiv (2018)
31. Leroux, S., Verbelen, T., Simoens, P., Dhoedt, B.: Privacy Aware Offloading of Deep Neural Networks. In: ICML Workshop (2018)
32. Li, M., Lai, L., Suda, N., Chandra, V., Pan, D.Z.: PrivyNet: A Flexible Framework for Privacy-Preserving Deep Neural Network Training. arXiv (2017)
33. Li, T., Lin, L.: AnonymousNet: Natural Face De-Identification with Measurable Privacy. In: CVPR Workshop (2019)
34. Lin, J., Gan, C., Han, S.: Tsm: Temporal shift module for efficient video understanding. In: ICCV. pp. 7083–7093 (2019)
35. Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., Zhang, C.: Learning Efficient Convolutional Networks through Network Slimming. In: ICCV (2017)
36. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep Learning Face Attributes in the Wild. In: ICCV (2015)
37. Ma, N., Zhang, X., Zheng, H.T., Sun, J.: ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design. In: ECCV (2018)
38. Newton, E.M., Sweeney, L., Malin, B.: Preserving Privacy by De-Identifying Face Images. TKDE (2005)
39. Oh, S.J., Benenson, R., Fritz, M., Schiele, B.: Faceless Person Recognition; Privacy Implications in Social Media. In: ECCV (2016)
40. Oh, S.J., Fritz, M., Schiele, B.: Adversarial Image Perturbation for Privacy Protection: A Game Theory Perspective. In: ICCV (2017)
41. Osia, S.A., Shamsabadi, A.S., Taheri, A., Katevas, K., Sajadmanesh, S., Rabiee, H.R., Lane, N.D., Haddadi, H.: A Hybrid Deep Learning Architecture for Privacy-Preserving Mobile Analytics. TKDD (2017)
42. Osia, S.A., Taheri, A., Shamsabadi, A.S., Katevas, K., Haddadi, H., Rabiee, H.R.: Deep Private-Feature Extraction. In: TKDE (2018)
43. Raval, N., Machanavajjhala, A., Cox, L.P.: Protecting visual secrets using adversarial nets. In: CVPRW (2017)
44. Ren, Z., Lee, Y.J., Ryoo, M.S.: Learning to Anonymize Faces for Privacy Preserving Action Detection. In: ECCV (2018)
45. Ryoo, M.S., Kim, K., Yang, H.J.: Extreme Low Resolution Activity Recognition With Multi-Siamese Embedding Learning. In: AAAI (2018)
46. Ryoo, M.S., Rothrock, B., Fleming, C., Yang, H.J.: Privacy-Preserving Human Activity Recognition from Extreme Low Resolution. In: AAAI (2017)
47. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: MobileNetV2: Inverted Residuals and Linear Bottlenecks. In: CVPR (2018)
48. Shokri, R., Shmatikov, V.: Privacy-Preserving Deep Learning. In: CCS (2015)

49. Simonyan, K., Zisserman, A.: Very Deep Convolutional Networks for Large-Scale Image Recognition. In: ICLR (2015)
50. So, D., Le, Q., Liang, C.: The Evolved Transformer. In: ICML (2019)
51. Sweeney, L.: K-anonymity: A model for protecting privacy. Int. J. Uncertain. Fuzziness Knowl.-Based Syst. (2002)
52. Tian, Y., Krishnan, D., Isola, P.: Contrastive representation distillation. ICLR (2020)
53. Tramèr, F., Boneh, D.: Slalom: Fast, Verifiable and Private Execution of Neural Networks in Trusted Hardware. In: ICLR (2019)
54. Wang, J., Zhang, J., Bao, W., Zhu, X., Cao, B., Yu, P.S.: Not Just Privacy: Improving Performance of Private Deep Learning in Mobile Cloud. In: KDD (2018)
55. Wang, K., Liu, Z., Lin, Y., Lin, J., Han, S.: HAQ: Hardware-Aware Automated Quantization with Mixed Precision. In: CVPR (2019)
56. Warden, P.: Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition. arXiv (2018)
57. Wu, Z., Liu, Z., Lin, J., Lin, Y., Han, S.: Lite Transformer with Long-Short Range Attention. In: ICLR (2020)
58. Wu, Z., Wang, Z., Wang, Z., Jin, H.: Towards Privacy-Preserving Visual Recognition via Adversarial Training: A Pilot Study. In: ECCV (2018)
59. Xie, Q., Hovy, E., Luong, M.T., Le, Q.V.: Self-training with Noisy Student improves ImageNet classification. In: arXiv (2019)
60. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond Empirical Risk Minimization. In: ICLR (2018)
61. Zhang, X., Zhou, X., Lin, M., Sun, J.: ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices. In: CVPR (2018)
62. Zhu, C., Han, S., Mao, H., Dally, W.: Trained Ternary Quantization. In: ICLR (2017)