You Can Use But Cannot Recognize: Preserving Visual Privacy in Deep Neural Networks

Qiushi Li[†]*, Yan Zhang[†]*, Ju Ren[✉]*[‡], Qi Li[§‡], Yaoxue Zhang*[‡]
* Department of Computer Science and Technology, Tsinghua University, Beijing, China
[‡] Zhongguancun Laboratory, Beijing, China
[§] Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing, China
{lqs@, yan-zhan23@mails., renju@, qli01@, zhangyx@}tsinghua.edu.cn

***Abstract*—Image data have been extensively used in Deep Neural Network (DNN) tasks in various scenarios, e.g., autonomous driving and medical image analysis, which incurs significant privacy concerns. Existing privacy protection techniques are unable to efficiently protect such data. For example, Differential Privacy (DP) that is an emerging technique protects data with strong privacy guarantee cannot effectively protect visual features of exposed image dataset. In this paper, we propose a novel privacy-preserving framework *VisualMixer* that protects the training data of visual DNN tasks by pixel shuffling, while not injecting any noises. *VisualMixer* utilizes a new privacy metric called Visual Feature Entropy (VFE) to effectively quantify the visual features of an image from both biological and machine vision aspects. In *VisualMixer*, we devise a task-agnostic image obfuscation method to protect the visual privacy of data for DNN training and inference. For each image, it determines regions for pixel shuffling in the image and the sizes of these regions according to the desired VFE. It shuffles pixels both in the spatial domain and in the chromatic channel space in the regions without injecting noises so that it can prevent visual features from being discerned and recognized, while incurring negligible accuracy loss. Extensive experiments on real-world datasets demonstrate that *VisualMixer* can effectively preserve the visual privacy with negligible accuracy loss, i.e., at average 2.35 percentage points of model accuracy loss, and almost no performance degradation on model training.**

I. Introduction

Neural network models have been applied to a wide range of promising image applications, e.g., computer vision, autonomous driving, and medical image analysis [48]. Existing studies [17, 13, 54, 53] show that these neural network models can leak the training datasets, *e.g.*, by constructing model reconstruction attacks, *i.e.*, reconstructing training data according to the model weights, gradients, and other model information. However, image data that are used to train these models often contains personal privacy information, such as facial characteristics, license plate numbers, and geographic locations. Similarly, medical image data used for training models also involves a large amount of sensitive patient information.

[†]Both authors contributed equally to this work.
[✉]Corresponding author.

Network and Distributed System Security (NDSS) Symposium 2024
26 February - 1 March 2024, San Diego, CA, USA
ISBN 1-891562-93-2
https://dx.doi.org/10.14722/ndss.2024.241361
www.ndss-symposium.org

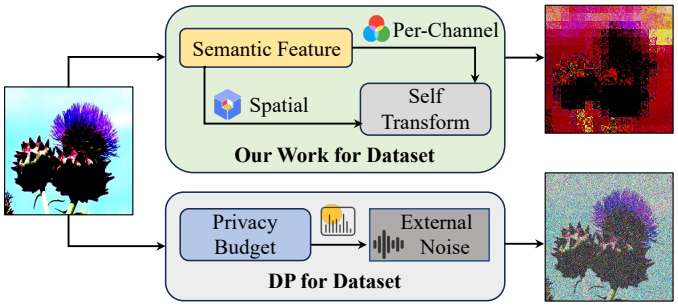

Fig. 1: Our work attempts to protect visual privacy through self-transformation guided by metric of semantic features. DP adds external noise to images, preventing adversaries from distinguishing whether a sample is present in the dataset. This approach is not intended for visual privacy protection in the context of dataset publication.

A number of privacy-preserving techniques [4] have been developed to address the privacy issues above. For example, homomorphic encryption (HE) enables rigorous data privacy guarantees by encrypting data and ensures that data remains usable yet invisible. Trusted Execution Environments (TEEs) based methods utilize trusted hardware to protect data for training DNNs. However, these methods requires substantial extra computational resources or requires specific hardware [7] and thus their practicality for image tasks is not clear in practice. Differential Privacy (DP) is a promising technique that can effectively protect data membership with privacy guarantee. By adding controlled noises from predetermined distributions of datasets, DP incurs negligible training delays with acceptable inference accuracy. However, as shown in Figure 1, DP is unable to effectively protect visual features of exposed image data because the generated noises that are in high frequency domain can be filtered by human eyes [50]. Moreover, DP may incur the obvious performance decrease if the privacy guarantee is strong enough to protect visual feature [51]. Thus, it is crucial to protect the privacy of these image data during model training, while maintaining the model performance.

In this paper, we propose a privacy-preserving framework *VisualMixer* that aims to protect the training data of visual DNN tasks by shuffling pixels without injecting any noises, which ensures that *VisualMixer* achieves data protection while retaining the performance of the DNN tasks. Our framework utilizes a new metric called *Visual Feature Entropy* (VFE) to effectively quantify the visual features of an image from

both biological and machine vision aspects. It can accurately evaluate task-agnostic visual features of an image with a collection of pixel chromatic value gradients. We can evaluate the uncertainty of visual features in an image by measuring VFE so that we can reduce the amount of privacy information contained in the image. In order to achieve desired VFE, we devise VisualMixer, a task-agnostic image obfuscation method to protect the visual privacy of data for DNN training and inference. As shown in Figure 1, for each image in the dataset, VisualMixer decides regions for pixel shuffling in the image and the sizes of these regions according to the desired VFE, and then shuffles pixels both in the spatial domain and in the chromatic channel space in the regions. By shuffling pixels within the determined regions, it can prevent visual features from being discerned and recognized, while incurring negligible accuracy loss.

Furthermore, since image obfuscation may incur abrupt gradient changes in the global feature space and significant weight gradient fluctuations, it may impede the normal convergence process of the model. In *VisualMixer*, to address these issues incurred by image obfuscation, we develop an optimizer algorithm called *Adaptive Momentum Stochastic Gradient Descent* (i.e., ST-Adam). ST-Adam dynamically adjusts update momentum based on the current gradient and historical gradients to accelerate the model convergence speed and ensure the stability of model training.

We summarize the contributions of the paper as follows.

- We propose the first DNN image data protection framework that can effectively protect image data while retaining the accuracy of DNN tasks.
- We develop a new metric, Visual Feature Entropy (VFE), which is an effective indicator to measure the visual features of an image and can quantify the visual privacy of an image.
- We propose VisualMixer, a noise-free and task-agnostic image obfuscation method, to protect the visual privacy of image data in DNN training and inference according to the desired VFE, and devise an optimizer, ST-Adam, to accelerate the model training performance over the obfuscated iamges.
- We provide a comprehensive analysis to demonstrate the effectiveness of VisualMixer and the correlation between the deviation of model outputs and the sizes of shuffling regions.
- Extensive experiments on real-world datasets demonstrate that VisualMixer can effectively preserve the visual privacy with negligible accuracy loss, i.e., at average 2.35 percentage points of model accuracy loss, and almost no performance degradation on model training.

Reproducibility. To help researchers reproduce and verify our results, we release the source code of VisualMixer on https://github.com/Edison9419/ndss.

II. Threat Model and Design Goals

A. Threat Model

We consider a general scenario of image data protection in DNN tasks, where clients send their image data (plain or obfuscated) to the server for training a DNN model and then use the trained model for inference. The server honestly perform the tasks but tries to steal the corresponding data, *i.e.*, manually identifying images and utilizing attack methods to extract features or information from the DNN model trained by data uploaded from clients. Specifically, we focus on the following three attacks.

Accessing Data uploaded by Clients. Adversaries on servers can directly access data uploaded by clients. Even when clients obfuscate their images, adversaries still try to recover the original visual features using brute-force or heuristic attacks.

Reconstructing DNN Trained Data. Adversaries can perform membership inference attacks on the trained DNN model to identify the ownership of the data used in the training process [43]. Additionally, they can leverage data augmentation methods such as GAN-based data reconstruction to generate the visual features and recover partial information of the private training data based on the model weights trained by uploaded data from clients [25].

Recovering Intermediate Gradients and Features. Adversaries reconstruct visually distinguishable images based on the intermediate gradients [31] and feature maps during the training and inference process [32].

It is noticed that the label of the image is not private, because it is necessary for training utility.

B. Design Goals

The goal of the paper is to preserve the visual privacy of client image data during the model training and inference process, while retaining the accuracy and performance of the DNN models. The design goals can be summarized as follows.

Quantifying the Task-Agnostic Visual Privacy. In the domain of DNN-based vision tasks, the majority of models lack specific interpretability, leading to many visual features involved in the learning process. How to obfuscate the visual features to strike a trade-off between utility and privacy is still an open challenge. Therefore, we should first design a metric to quantify the task-agnostic privacy level of the obfuscated visual features.

Visual-Semantic Obfuscation. Generic data encryption methods, provide strict security guarantees but come with significant computational overhead, especially for high-dimension image data. Unlike traditional data-level obfuscation methods, such as DP that injects noises with a privacy budget, image obfuscation should consider the semantic information of the visual features, to prevent adversaries from obtaining meaningful visual privacy information while keep the data utility for diverse vision tasks. Thus, the second goal is to design a visual-semantic image obfuscation method to balance the model accuracy and privacy level.

Optimization for Gradient Oscillation. Data obfuscation often causes the gradient oscillation problem, making the DNN

model difficult to converge due to the increased randomness of gradients. Existing gradient descent strategies fail to keep the model stability and convergence speed over the obfuscated data. A tailored optimizer should be designed, associate with the data obfuscation method, to tackle the gradient oscillation problem.

III. Visual Feature Entropy: A New Metric to Measure Visual Privacy

A. Limitation of DP on Releasing Visual Datasets

Some traditional methods, *e.g.* Differential privacy (DP), originally designed for statistical data, are facing significant limitations in the field of visual privacy protection. Liu et al. [33] introduced external noise to the feature map to protect the released model from membership detection, but this approach cannot be applied to protecting the released dataset. Dwork [10] proposed to protect the released model by adding external noise to the dataset, but it fails to protect the visual privacy of image data. As shown in Figure 1, the derived dataset can still reveal some visual information.

The fundamental reason for the insufficient privacy of DP for vision dataset lies in the lack of metrics that can measure the visual privacy. When adding external noise in DP, only the privacy budget is taken into account, rather than the visual features of the images themselves. This results in the inability to eliminate visual privacy from the dataset, even if DP introduces excessive noises that significantly reduce the data utility. Therefore, to eliminate privacy features from the dataset, it is critical to first define a new metric that can measure the visual privacy.

Besides, the privacy metric should be task-agnostic. This is because a dataset usually exhibits multitasking characteristics, where the features that are not privacy-sensitive in one task context may become privacy-sensitive in another. For instance, in an image dataset for autonomous driving, license plates and pedestrians might expose sensitive privacy, while non-task-related information such as weather and architectural style can still reveal privacy-related details like geographic location. Therefore, in order to eliminate a sufficient amount of visual privacy, we introduce Visual Feature Entropy (VFE) to measure the visual features, ensuring the ability to assess privacy in multitasking scenarios.

B. Definition of Visual Feature Entropy

In information theory, entropy measures the expected (*i.e.*, average) amount of information conveyed by identifying the outcome of a random trial. The entropy of a random variable is the average level of "uncertainty" inherent to the variable's possible outcomes. Inspired by that, we propose **Visual Feature Entropy** (VFE) to quantify the "uncertainty" of visual features. For an image, a higher VFE exhibits more disorder, indicating a higher uncertainty in visual features, resulting in fewer visual features that can be accurately discerned by biological vision. With this intuition, the definition of VFE is as follows.

For a given image I, we denote the RGB value of a pixel at an arbitrary position (x, y) as $I(x, y), x \in \{0, 1, \ldots, N_1 - 1\}, y \in \{0, 1, \ldots, N_2 - 1\}$, where N_1 and N_2 represents the width and height of an image, respectively. Consequently, $I(\cdot)$ can be regarded as a discrete function defined over a discrete domain. The "gradient" of $I(\cdot)$ is defined as

$$\begin{aligned} \nabla_x I(x, y) &= I(x + 1, y) - I(x, y), \quad x \in \{0, 1, \ldots, N_1 - 1\} \\ \nabla_y I(x, y) &= I(x, y + 1) - I(x, y), \quad y \in \{0, 1, \ldots, N_2 - 1\} \end{aligned} \tag{1}$$

The definition of VFE can be used to measure the visual privacy of any region in an image. For any region R_I with width w and height h, if we denote the location of the left top pixel of the region as (x_0, y_0), then the VFE of this region can be calculated as

$$VFE_R(R_I) = \sum_{x=x_0}^{x_0+w-1} \sum_{y=y_0}^{y_0+h-1} \left(\nabla_x I(x, y)^2 + \nabla_y I(x, y)^2 \right) \tag{2}$$

Then, the set $\boldsymbol{R_I}$ represents the collection of all sub-regions R_I of the image I. The VFE of I can be calculated as

$$VFE(I) = \frac{F}{N_1 N_2} \sum^{R_I \in \boldsymbol{R_I}} VFE_R(R_I) \tag{3}$$

where F is a constant scaling factor (typically set to 1) used to prevent the VFE from being too small or too large value. The aforementioned definition pertains to the VFE of a single channel, *i.e.*, $VFE(I)$. For a multi-channel image, the VFE is defined as

$$VFE_M(I) = \frac{1}{C} \sum^{I_c \in I} VFE(I_c) \tag{4}$$

, where C is number of channels, I_c is one channel of the image I, and $VFE_M(I)$ is the VFE of a multi-channel image. In this context, the VFE of a multi-channel image is the average of the VFEs across all channels.

Under this definition, an image with a higher visual feature entropy indicates that there is a higher density of gradient variations in the image. It also means that the visual features have more "uncertainty", are more difficult to identify and hence the image has a higher visual privacy.

C. Intuition of Visual Feature Entropy

Fourier transform can provide information in the frequency domain, and it appears to be a potential tool for quantifying this uncertainty and disorder. However, regions of images with frequent gradient changes may not necessarily yield high-frequency information. For instance, multiple low-frequency signals with phase differences can be expressed as high VFE, but Fourier transform only displays as multiple low-frequency signals.

In fact, characterizing this uncertainty using the gradient of the image might be more appropriate. This is because the gradient represents only the absolute difference between adjacent pixels in an image, without reflecting its frequency content [16]. This implies that we can compute the sum of image variations from the differences between neighboring pixels using the image gradient, thereby avoiding the misleading effects of low-frequency signals. Additionally, by calculating the mean gradient, we can derive a measure of uncertainty akin to what high-frequency signals convey, resulting in an actual representation of VFE. Consequently, the definition of VFE is primarily designed by insights from the image gradient.

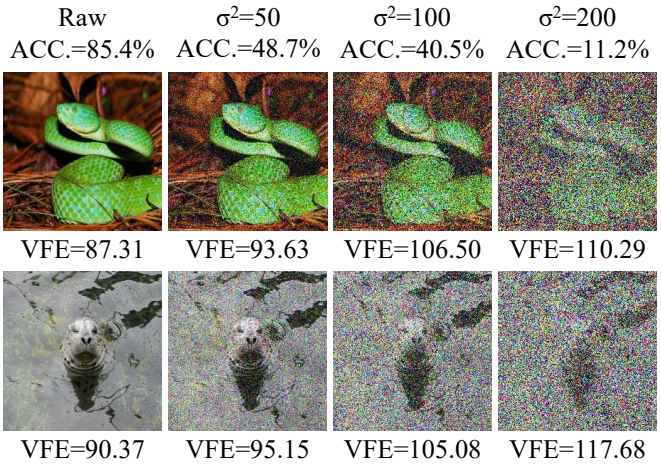

Fig. 2: VFE of Obfuscated Images by Adding More Noises with DP. (σ^2 reflects the amount of noises added to the image dataset and feature map during training. ACC denotes the accuracy of the ShuffleNet model that is trained using the obfuscated images.)

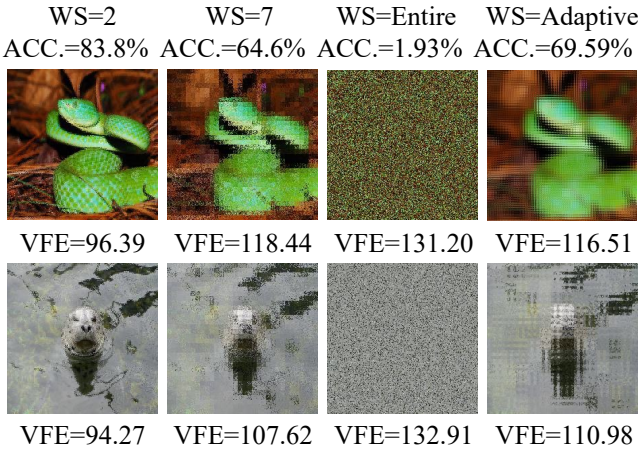

Fig. 3: VFE of Obfuscated Images under Different Shuffling Strategies. (*WS* means the window size we used in the corresponding shuffling strategy. ACC denotes the accuracy of the ShuffleNet model that is trained using the obfuscated images.)

D. How VFE Reflects the Privacy of Visual Features?

VFE is an universal metric that can quantify semantically irrelevant visual features for an image. In other words, the methods capable of obfuscating visual features, including Differential Privacy (DP), should enhance VFE. In comparison to the image by DP in Figure 1, we add more Gaussian noise, following DP's rules, to the image dataset to perform actual visual obfuscation. It can be seen from Figure 2 that, as the adding noise increases, the VFEs of the images are increasing correspondingly and the visual features are becoming more difficult to identify.

However, additional noises on the dataset and feature map may significantly reduce the accuracy of DNN model. Since DP introduces external Gaussian noise that is typically unrelated to the underlying data, it leads to the possibility of some noise occupying a portion of the visual feature space, resulting in a decrease in accuracy. As shown in Figure 2, after applying DP and introducing external noise, the accuracy of the ShuffleNet model decreases dramatically. Particularly, when we add noise of $\sigma^2 = 100$, some detail visual features are not recognizable, such as texture on leaves, but the accuracy of the ShuffleNet model drops to 40.5% making the dataset useless.

Given that adding external noise may potentially lead to a decrease in accuracy, an alternative way to increase the VFE is to spatially shuffle the pixels of the image. However, there are different shuffling strategies to achieve different privacy protection strengths. The extreme way is to randomly shuffle all the pixels of the entire image. It can bring the maximum strength of visual privacy protection, because all the visual features are destroyed entirely, as shown in Figure 3 (c). In this case, the VFE naturally increases to a very large value but the model accuracy also drops to be zero.

There is also a soft way to shuffle the images. We can first divide an image to multiple disjoint windows, where each window has equal width and height, denoted by window size *WS*. For instance, if $WS = 7$, it means that we divide the image into a number of windows, each of which contains 7×7 pixels. Then, in each window, we randomly shuffle the pixels. It is obvious that, as the window size increases, pixel shuffling will break more structural information in the image and lead to an enhanced privacy protection. We can also find from Figure 3 that the VFE is increasing with the increment of *WS*, while the model accuracy goes in an opposite direction.

Comparing Figure 2 and Figure 3, we can find that VFE accurately identifies the protection strength of visual privacy under different obfuscation methods. Although both of the methods are facing the dilemma between privacy and data utility, shuffling performs much better than adding-noise in preserving the visual privacy and keeping the data utility. However, to meet the desired privacy requirement, the utility of the obfuscated images, i.e., the accuracy of the DNN model trained by the obfuscated images, is far from satisfaction under such a simple shuffling strategy. In the next subsection, we will discuss how to use VFE to guide the design of a sophisticated shuffling strategy that may achieve the trade-off between privacy and model accuracy.

E. How VFE Guides the Design of Shuffling Strategy?

The above section demonstrates that the value of window size, *i.e.*, *WS*, determines the trade-off between privacy level and data utility when we adopt pixel shuffling to protect the visual privacy. In each window, besides spatially shuffling the RGB pixels in the image, we can additionally perform per-channel shuffling, *i.e.*, shuffle elements in each channel to increase the VFE and hence to improve the protection of visual features. Per-channel shuffling can effectively obfuscate the color features of an image to preserve the visual privacy but has little impact on the structure and texture features that are more important for the data utility of various vision tasks.

In addition to adding a new shuffling dimension, we should focus on why the image obfuscation methods used in Figure 2 and Figure 3 experience dramatic accuracy drop. The main

reason is that both of them adopt a uniform data obfuscation strategy in the whole image, without considering the semantic information. It consequently causes that the regions of interests (RoIs) are obfuscated too much to keep the data utility. According to the definition of VFE, different regions of an image usually have various VFE values. This follows a common sense that different parts of an image contain different amount of visual feature information. With the guidance of VFE, we can design a non-uniform shuffling strategy, where the regions with low VFE should be shuffled under a larger window size while the regions with high VFE can be shuffled under a smaller window size. In such a way, we can increase the VFE of the image to enhance the protection strength on visual privacy and keep more structure and texture information of RoIs.

Based on the design principle, we performed some experiments with a non-uniform shuffling strategy, where we set $WS = 8$ for the windows with original VFE less than average VFE and set $WS = 2$ for the windows with larger VFE. As shown in Figure 3 (d), we can find that the non-uniform shuffling strategy can improve the model accuracy under a similar VFE.

Thus, it is crucial to design a VFE-guided non-uniform image shuffling strategy to strike the trade-off between visual privacy and data utility. The strategy design should answer the following question: how to divide an image to a number of disjoint windows with non-uniform sizes? Then, the key problem comes to determining the optimal window sizes for different regions in an image to maximize the model accuracy under a given visual privacy protection requirement (i.e., the VFEs of the shuffled images are larger than a specific value).

IV. VisualMixer (VIM): A VFE Guided Privacy-Preserving Image Shuffling Strategy

This section introduces the details of the image obfuscation method, VisualMixer (VIsualMixer, VIM), and the tailored DNN training optimizer, ST-Adam. Guided by the VFE of the images, VisualMixer determines the optimal window sizes for the shuffling strategy to achieve a trade-off between visual privacy and data utility. The tailored optimizer, ST-Adam, is then proposed to address the gradient oscillation problem caused by VisualMixer.

A. Approach Overview

This section presents the architecture and working flow of VisualMixer. As shown in Figure 4, VisualMixer primarily focuses on the data preprocessing stage, aiming to eliminate the visual semantics while preserve the trainable image information. It is guided by the VFEs of different regions in an image to determine the optimal window sizes. In each window, VisualMixer randomly shuffles the pixels in space and channels to obfuscate the visual features. In addition, since the shuffled images make the training process of the DNN model unstable and very difficult to converge, we design a tailored optimizer, named ST-Adam, to work with VisualMixer. By combining momentum optimization and adaptive learning rate adjustment, ST-Adam can significantly improve the convergence speed of model training over the shuffled image data.

The working flow of VisualMixer is as follows. Before the model training stage, clients first use VisualMixer to obfuscate their images for meeting the privacy requirement. Specifically, VisualMixer adaptively determines the image shuffling strategy based on the value of VFE and the expectation of model accuracy. Then, clients send shuffled images to server for training. The trained model need work on shuffled images during inference, but do not need decryption.

During the inference stage, we also only need to use VisualMixer to shuffle the image and then send the shuffled images to the server for inference. It implies that there is no need to modify the model architecture or the underlying computational framework associated with the model itself. Therefore, VisualMixer can be applied not only in model training and inference scenarios but also in various DNN-related learning tasks.

In the following subsections, we first introduce how to determine the lower bounder of window size to meet the required VFE. Based on that, we illustrate how to determine the optimal window sizes for different regions to maximize the accuracy of a DNN model. Then, we summarize the key step of VisualMixer into an algorithm, followed by the design details of ST-Adam.

B. Determining the Lower Bound of Window Size for VisualMixer

This section primarily elucidates the relationship between VFE and per-channel shuffling. In particular, it analyzes the mathematical relationship between VFE and the window size on a single channel. This is directly applicable to monochromatic images. For multi-channel images, it remains usability. As inferred from Equation 4, the VFE across multiple channels should be the average of the VFEs of each channel, *i.e.*, the effect of shuffling on VFE is additive. Consequently, in the following, we first derive the relationship between VFE and per-channel shuffle.

In the shuffling strategy, since a larger window size brings a higher VFE, this section first derives the lower bound of window size for VisualMixer, which can be used to guarantee basic requirement of VFE.

For any region R_I in an image I, we denote its window size as *WS*. Without the loss of generality, we assume that the pixel values in R_I follow a statistically normal distribution. Additionally, after randomly shuffling the pixels within the region, we consider adjacent pixels to be independently and identically distributed.

Based on the assumptions, we can calculate the approximate distribution of the VFE of R_I. We denote the pixel values in R_I as $\{p_1, p_2, \ldots, p_{WS^2}\}$, then we can obtain the maximum likelihood estimates of their mean and variance:

$$\mu = \frac{\sum_{x,y}^{R_I} I(x,y)}{WS^2}, \sigma^2 = \frac{\sum_{x,y}^{R_I} (I(x,y) - \mu)^2}{WS^2 - 1} \tag{5}$$

With the calculated mean μ and variance σ^2, we can convert the pixels in R_I into a standard normal distribution

$$\hat{I}(x,y) \leftarrow \frac{I(x,y) - \mu}{\sigma} \tag{6}$$

Now, $\hat{I}(x,y)$ follows a standard normal distribution independently. Therefore, for any two different pixels, *i.e.*, $x_1 \neq$

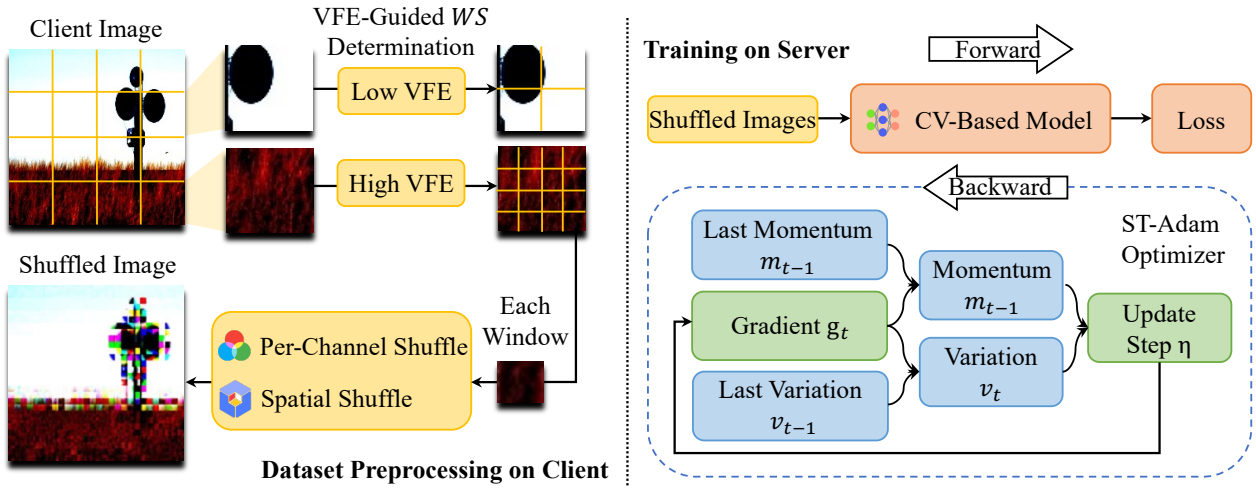

Fig. 4: The Architecture and Working Process of VisualMixer

x_2 or $y_1 \neq y_2$, we have

$$\hat{I}(x_1,y_1) - \hat{I}(x_2,y_2) \sim N(0,2) \Rightarrow \frac{\sqrt{2}}{2}(\hat{I}(x_1,y_1) - \hat{I}(x_2,y_2)) \sim N(0,1) \tag{7}$$

We need to calculate the distribution of VFE of R_I, *i.e.*, $\hat{VFE}(R_I)$. We substitute Equation 1 and Equation 6 into Equation 2, and get

$$VFE_R(R_I) = \sum_{x,y}^{R_I}(\sigma^2(\hat{I}(x+1,y) - \hat{I}(x,y))^2 + \sigma^2(\hat{I}(x,y+1) - \hat{I}(x,y))^2) \tag{8}$$

By substituting Equation 7 to Equation 8, applying the computational rules for normal distributions, we can obtain the $VFE_R(R_I)$'s distribution $\hat{VFE}(R_I)$

$$\begin{aligned} \hat{VFE}_R(R_I) &\sim \sum_{x,y}^{R_I}(\sigma^2(N(0,2))^2 + \sigma^2(N(0,2))^2) \\ &\sim \sigma^2 \sum_{x,y}^{R_I}((N(0,2))^2 + (N(0,2))^2) \end{aligned} \tag{9}$$

For R_I, there are $WS(WS-1)$ pairs of ∇_x and $WS(WS-1)$ pairs of ∇_y. By the definition of the chi-square distribution, we can derive

$$\begin{aligned} \hat{VFE}_R(R_I) &\sim 2\sigma^2 \sum_{x,y}^{R_I}((N(0,1))^2 + (N(0,1))^2) \\ &\sim 2\sigma^2 \chi^2_{2WS(WS-1)} \\ \frac{1}{2\sigma^2}\hat{VFE}_R(R_I) &\sim \chi^2_{2WS(WS-1)} \end{aligned} \tag{10}$$

It implies that $\frac{1}{2\sigma^2}VFE_R(R_I)$ follows a chi-square distribution $\chi^2_{2WS(WS-1)}$. Meanwhile, $VFE_R(R_I)$ of different R_I can be regarded as following independent identical distribution. Therefore, the sum of the $VFE_R(R_I)$ of an image I follows a χ^2 distribution, as well, sepcifically,

$$\sum VFE_R(R_I) \sim \chi^2_{\frac{2wh}{WS^2}WS(WS-1)} \sim \chi^2_{\frac{2wh(WS-1)}{WS}} \tag{11}$$

where w and h mean the width and height of the image. The degree of freedom, $\frac{2wh(WS-1)}{WS}$ is so large that we can regard the χ^2 distribution as a normal distribution with the mean of $\frac{2wh(WS-1)}{WS}$ and the variance of $\frac{4wh(WS-1)}{WS}$. According to Equation 3, the distribution of the *VFE* of an image can be got from following equations:

$$\begin{aligned} \frac{wh}{WS^2}VFE(I) &\sim N\left(\frac{2wh(WS-1)}{WS}, \frac{4wh(WS-1)}{WS}\right) \\ VFE(I) &\sim N\left(2WS(WS-1), \frac{4WS^3(WS-1)}{wh}\right) \end{aligned} \tag{12}$$

When the width and height of the image is set to 224 (for ImageNet-100) and the window size is set to 8, the mean of $VFE(I)$ will be 112; and the variance of $VFE(I)$ will be $\frac{2}{7}$.

At this point, we have established the probabilistic relationship between *WS* and VFE. Based on this probability relationship, for each image obfuscation, we can compute lower bound of *WS* of the ideal VFE with offline manner. This lower bound of obfuscation strength ensures the privacy preservation during the actual obfuscation process.

C. Determining the Upper Bound of Window Size for VisualMixer

In Section IV-B, we present that the lower bound of window size WS_l is constrained by the desired VFE. Since the increment of window size leads to a decrease in data utility, this section proceeds to derive an upper bound of *WS* that can optimize the accuracy of the DNN model. Based on the two bounds, we can attain the optimal *WS* for VisualMixer to calibrate the privacy-utility trade-off correctly.

The challenge of optimizing the data utility of VisualMixer lies in quantifying the impact of shuffling on the accuracy of a DNN model under given window sizes. Instead of directly measuring the model accuracy, we use the output bias of the model to represent the impact of shuffling on data utility. The idea is that, for a specific shuffling strategy with a given *WS*, we calculate the maximum output bias between the original

image and the shuffled image. Then, we use the maximum output bias as the indicator of data utility to determine the upper bound of window size WS_u. In what follows, we use mathematical induction to perform our analysis.

Currently, in the domain of DNN for CV-tasks, the primary architectural choices of are Convolution-based or Transformer-based. Both Transformer and Convolution are linear structures and share fundamental similarities. Therefore, in this section, we provide the calculation proof for WS_u using the CNN as an example.

For base case in mathematical induction, we begin by considering the typical module of an CNN model, consisting of a convolutional layer followed by a max-pooling layer. Denote W as the weight of convolutional kernel with a size of N_{CK} (CK represents ConvKernel) and a stride of N_{CS} (CS represents ConvStride), where its parameters w_i follows a normal distribution, i.e., $w_i \sim N(\mu_W, \sigma_W)$. The elements within W are arranged from largest to smallest as $\{w_1, w_2, \ldots, w_{size}\}$. Denote the size of the max-pooling operator as N_{MP} and the size of the stride as N_{MPS}, where $N_{MP} = N_{MPS} = 2$ is a common setting for max-pooling operators. We set the parameters of the convolutional layer as a minimum $N_{CK} = 2 \times 2$ and $N_{CS} = 1$.

Consistent with the previous statement, we denote the window size of a region in image I as *WS*, which is set to 3 for base case in mathematical induction. We first calculate the output bias in this base case and then generalize it to all cases where $N_{CK} \geq 2 \times 2$ and $WS \geq 3$. We normalize the pixel value in the window, denoted by $I_{ij} \in [0, 1]$. I' denotes the shuffled image of I; A and A' denote the feature map obtained through the convolution layer in which I and I' has been processed, respectively; B and B' denote the outputs after A and A' are processed by the max-pooling operator, respectively. Our purpose is to calculate the distribution of $diff_{max}$ that denotes the maximum difference between B and B'. Since the sizes of B and B' are 1×1, we can get possible B_{max} and B_{min}

$$B_{max} = \max\{A_{ij}\} \leq \sum I_{ij} \cdot W_{ij}, \text{if } W_{ij} > 0 \text{ then } I_{ij} = 1 \text{ else } I_{ij} = 0 \tag{13}$$

$$B_{min} = \max\{A_{ij}\} \geq \sum I_{ij} \cdot W_{ij}, \text{if } W_{ij} < 0 \text{ then } I_{ij} = 1 \text{ else } I_{ij} = 0 \tag{14}$$

Therefore, the maximum output bias can be calculated as

$$diff_{max} = |B - B'| = B_{max} - B_{min} \tag{15}$$

By probability theory, the expectation of $diff_{\max}$, i.e., $\mathbb{E}(diff_{\max})$, can be calculated as an integral over a four-dimensional space D (because of $N_{CK} = 2 \times 2$)

$$\iiiint_D (w_1 + w_2) p(w_1) \frac{p(w_2)}{\Phi(w_1)} \frac{p(w_3)}{\Phi(w_2)} \frac{p(w_4)}{\Phi(w_3)} dw_1 dw_2 dw_3 dw_4 \tag{16}$$
$$D = \{\vec{w} = (w_1, w_2, w_3, w_4) | \vec{w} \in \mathbb{R}^4, w_1 \geq w_2 \geq w_3 \geq w_4\}$$

where functions $p(\cdot)$ and $\Phi(\cdot)$ represent the density function and the cumulative distribution function, respectively.

However, it is infeasible to derive the closed-form expectation of $diff_{\max}$ by Equation (16). Then, we can give a complete induction to calculate B, by enumerating all possible cases of I in extreme value space, *i.e.*, $\{0, 1\}$. With a 3×3 matrix of I, it has totaling $2^{3\times3} = 512$ cases. Based on our assumptions, we know that there could be only five possibilities for the sign combination of w_1, w_2, w_3, w_4. Combined with the 512 cases of I, there are a total of 2560 cases. We have enumerated all of these.

TABLE I: Complete induction of all possible output B in $I \in \{0, 1\}$.

B	Number	Percentage
0	200	7.8%
w1	457	17.9%
w2	212	8.3%
w3	70	2.7%
w4	9	0.4%
w1 + w2	473	18.5%
w1 + w3	247	9.6%
w1 + w4	60	2.3%
w2 + w3	140	5.5%
w2 + w4	15	0.6%
w3 + w4	3	0.1%
w1 + w2 + w3	411	16.1%
w1 + w2 + w4	92	3.6%
w1 + w3 + w4	28	1.1%
w2 + w3 + w4	5	0.2%
w1 + w2 + w3 + w4	138	5.4%

The enumeration data in Table I allows us to perform Monte Carlo simulations, taking into account the probability of each combination and the assumption that the convolution kernel follows a normal distribution. This allows us to determine the parameter d at any level of confidence. As the size of the convolution kernel increases, similar methods can be used for calculation.

Moreover, to show the influence of the conclusion above given by size increasing of I and I', we define the following notations. $S_I(')$ is the size of $I(')$, similarly for $S_A(')$ and $S_B(')$. Since the stride of maxpooling operator is 2, we only consider the situation where $\Delta S_I = 2m, m = 1, 2, \ldots$. When $\Delta S_I = 2m - 1$, the maxpooling operator will trigger padding operation, which makes the situation similar to the case where $\Delta S_I = 2m$. Then $\Delta S_A = 2m$ and $\Delta S_B = m$, which means that there is m^2 elements in B and B'. Since each element of B and B' has independent predecessors in I, and $diff_{\max}$ only takes the maximum value, we can apply the multiplication rule of independent event probability calculations here. Therefore, we can obtain $\mathbb{P}(diff_{\max} \leq d)$. Then the probability that all of the $diff_{\max}$ with respect to all elements in B and B' are less than d should be

$$\mathbb{P}(diff_{\max} \leq d) = \alpha^{m^2}, \text{ if } N_{CK} \geq 2 \times 2 \text{ and } WS \geq 3 \tag{17}$$

When the size of W increases, *i.e.* m increases, which means the receptive field increases, according to Equation 13 and Equation 14, $diff_{\max}$ will definitely increase. It is unnecessary to consider the case where the S_I is larger than W, because $diff_{\max}$ is the maximum value obtained after exhausting all possible distributions of I. When the S_I is larger than the convolution kernel size, the convolution kernel will perform a sliding window on S_I. The size of each window remains the same size as the convolution kernel size during the sliding. Meanwhile, the $diff_{\max}$ obtained after all sliding windows must be smaller than or equal to the $diff_{\max}$ obtained after exhausting all possible distributions in the sliding window. It is worth noting that the mainstream structure of CNN consists of a convolution layer and a batch normalization layer. The

Algorithm 1: Image Processing in VisualMixer

Data: Input image I
Output: Shuffled image I' by VisualMixer

$WS_u \leftarrow WS_0 + 2 \times \lfloor \sqrt{log_{\alpha_0}\alpha} \rfloor$ // From Section IV-C
$WS_l \leftarrow WS$ from the distributions of target VFE
$WS \leftarrow Size(I)$
while $WS > WS_u$ **do**
  $WS \leftarrow 2^{\lfloor \log_2 WS/2 \rfloor}$
end
$\boldsymbol{R} \leftarrow$ Dividing image I into regions based on WS
while $\boldsymbol{R} \neq \varnothing$ **do**
  $\boldsymbol{R}_i \overset{R}{\leftarrow} \boldsymbol{R}$
  $WS \leftarrow Size(\boldsymbol{R}_i)$
  if $WS \leq 2^{\lfloor \log_2 WS_l \rfloor}$ **then**
    $WS = 2^{\lfloor \log_2 WS_l \rfloor}$
    Spatial and per-channel shuffle $\boldsymbol{R}_i$ with WS
    $\boldsymbol{R} = \boldsymbol{R} - \{\boldsymbol{R}_i\}$
  else
    $WS = \lceil Size(\boldsymbol{R}_i)/2 \rceil$
    if $VFE(\boldsymbol{R}_i) \leq VFE_m$ **then**
      Spatial and per-channel shuffle $\boldsymbol{R}_i$ with WS
      $\boldsymbol{R} = \boldsymbol{R} - \{\boldsymbol{R}_i\}$
    else
      $\boldsymbol{R} \leftarrow \boldsymbol{R} \cup$ Dividing $\boldsymbol{R}_i$ into regions based on WS
    end
  end
end
return I'

job of pooling layers, down-sampling, is done by convolution layers. However, the principle of mixing operation does not change; Thus, the conclusion also works in current CNNs.

Then, we adopt a reverse-solving method to determine the maximum allowable *WS*. The process to calculate the optimal value of *WS* for balancing privacy protection and accuracy loss begins with setting an initial value of *WS*, denoted as WS_0, to 3. Using a Monte Carlo simulation, we determine the value of α_0 that satisfies the condition $P(diff_{max} \leq d) = \alpha_0$. Next, we satisfy $P(diff_{max} \leq d) = \alpha$ by setting a given α. By setting $\alpha = \alpha_0^{m^2}$, we can calculate the value of m as $\sqrt{\log_{\alpha_0}\alpha}$. And finally, we compute *WS* by $WS = WS_0 + 2 \times \lfloor m \rfloor$. This process dynamically calculates the value of n to achieve a balance between data privacy and inference accuracy.

The shuffling is performed by randomly rearranging the pixels of the image, thereby changing the original feature distribution of the image. Shuffling can be done at the pixel level or at the block level, according to *WS*. Therefore, we first divide the image into sub-regions, and then we calculate the VFE of each sub-region. Then, we compare the VFE value of each sub-region with the median VFE_m. If $VFE_i > VFE_m$, we use a larger *WS* for shuffling; otherwise, we use a smaller *WS* for shuffling.

D. Algorithm of VisualMixer

The detailed algorithm of VisualMixer is summarized in Algorithm 1. Firstly, we need to calculate the upper and lower bounds of *WS* offline, WS_u and WS_l. The WS_u is used to control the maximum deviation of model output, *i.e.* diff_{max}, thereby preventing excessive loss of model accuracy. Based on the probability relationship proven in Section IV-C, we can obtain the WS_u for confidence probability α through the reversing method. The WS_l is used to control the expected VFE in order to ensure privacy. According to the conclusion proven in Section IV-B, we know that $\frac{1}{2\sigma^2}VFE$ follows a chi-square distribution. Therefore, by performing a backward table lookup, we can obtain the WS_l for the target VFE interval.

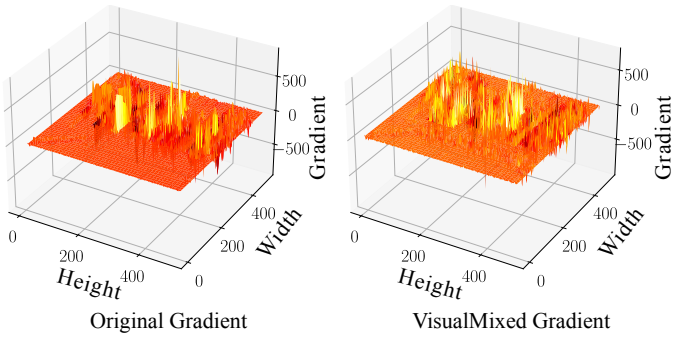

Fig. 5: Comparing gradients of original and VisualMixed images.

After obtaining the upper and lower bounds of *WS*, we need to adapt suitable *WS* for different regions of the image based on their VFE. Firstly, we scale down *WS* to the nearest power of two that is less than or equal to WS_u, ensuring the subsequent iterative *WS*. Next, we partition the image according to the current *WS* and add the segmented images to the set $\boldsymbol{R}$ of images to be processed. At each iteration, an image $\boldsymbol{R}_i$ is randomly selected from the set $\boldsymbol{R}$. Firstly, if its *WS* is lower than WS_l, it is directly shuffled using the *WS* closest to WS_l, which is a power of 2. If its *WS* is between the upper and lower bounds, it is determined based on the VFE. If the VFE is lower than the median VFE of the initial segmented set, it is shuffled using the next level of *WS*, *i.e.* $Size(\boldsymbol{R}_i)/2$. If the VFE is greater than the median VFE, it is partitioned into smaller blocks and added to the $\boldsymbol{R}$, awaiting the next round of *WS* and VFE checks.

This process is convergent, ensuring that all sub-images on the entire image will be protected by shuffling with dynamic *WS* intensities between WS_l and WS_u. It also ensures that the shuffle intensity stays within the required range, not exceeding WS_u for accuracy and not going below WS_l for VFE.

E. Stable Adaptive Moment Estimation

When using Adam to train models on the mixed dataset, we find that it is hard to get an ideal minimum so that the performance of trained models decreases sufficiently. To address the issue of models struggling to converge due to gradient oscillation in certain scenarios involving small datasets and individual models, we propose Stable Adaptive Moment Estimation (ST-Adam). ST-Adam achieves the ability to converge as expected in the presence of gradient oscillation by employing dual adaptivity for both momentum and learning rate. As shown in Figure 5, when training VGG on CIFAR10, the original gradients exhibit distinct features that facilitate rapid gradient

descent for optimizers like Adam, if no data obfuscation is applied. However, when training on obfuscated data, there is a significant fluctuation in gradients, referred to as gradient oscillation. This phenomenon results in a substantial loss in the model's convergence capability.

Here is the description of the update rules of ST-Adam. Firstly, the optimizer calculates the gradient of the loss function based on the value of parameters, $g_t = \nabla f(w_t)$. Then, According to the gradient and pre-defined hyperparameter β, the optimizer calculate the momentum of the loss function, $m_t = \beta \times m_{t-1} + (1-\beta) \times g_t$. After that, using another hyperparameter, γ, it get the adaptive learning rate, $v_t = \gamma \times v_{t-1} + (1-\gamma) \times g_t^2$. Finally, based on all of the information above, the optimizer can update the parameters of models, $w_{t+1} = w_t - \eta * \frac{m_t}{\sqrt{(v_t)+\epsilon}}$.

Here, w_t represents the weights at timestep t, g_t represents the gradient, m_t represents the momentum, v_t represents the adaptive learning rate, η is the initial learning rate, β and γ are the decay coefficients of momentum and adaptive learning rate, respectively, and ϵ is a smoothing term to prevent the denominator from being 0. We assume the objective function $f(w)$ is convex. In convex optimization, we are concerned with whether the distance between the optimized objective function value and the optimal solution decreases as the number of iterations increases. We define

$$\Delta w_t = w_t - w^*, \quad \Delta f_t = f(w_t) - f(w^*) \tag{18}$$

Here, w^* is the optimal solution. For convex functions, according to Jensen's inequality, we have

$$\Delta f_t \leq g_t^T \times \Delta w_t \tag{19}$$

Substituting the update rule of ST-Adam, we can obtain

$$\Delta f_t \leq (\frac{m_t}{\sqrt{v_t}+\epsilon})^T \times \Delta w_t \tag{20}$$

We can observe that when the gradient g_t is large, the momentum m_t and the adaptive learning rate v_t will also increase accordingly. When the gradient is small, m_t and v_t will decrease. This means that in areas with larger gradients, the optimizer will use larger update steps, thus speeding up the convergence process; in areas with smaller gradients, the optimizer will use smaller update steps to maintain a stable optimization process. Therefore, we can conclude that the convergence of ST-Adam is guaranteed. We added weight decay and momentum terms. Weight decay helps prevent model overfitting, especially when encountering drastic and dense gradient distributions during training. The momentum term helps speed up the optimization process, making it easier for the model to converge when encountering drastic gradient changes.

The main difference between the Adam and ST-Adam is that during the calculation of update step, Adam will rescale the step length according to the increase of time; while the step length obtained by ST-Adam is more stable. More specifically, in Adam, we need extra two calculations:

$$\begin{aligned} \hat{m_t} &= m_t/(1-\beta^t) \\ \hat{v_t} &= v_t/(1-\gamma^t) \end{aligned} \tag{21}$$

Those two steps rescale the value of m_t and v_t, which can make the update step of the Adam more flexible. In most cases, this will be an improvement for an optimizer, for the reason that it make it possible to let optimizer choose the update step adaptively according to the shape of the loss function in the neighbor. However, In the scenario of VisualMixer, the gradients are more likely to change dramatically within a limited range due to the mixing operation. That is to say, the flexible length of update steps is more like a poison than a benefit for our model training process. Although adaptive update steps con let the model converge more quickly, it makes the model get trapped into a local minimum with higher probability. Therefore, it is better keep the update steps stable to avoid the model being trapped in local minimum. That's why our ST-Adam remove the above two steps.

By introducing momentum and adaptive learning rate adjustments, the ST-Adam optimizer can maintain good convergence performance when dealing with data after VisualMixer. In Section V, we applied the ST-Adam optimizer to different convolutional neural network models and compared it with traditional SGD optimizers and other popular optimizers, such as Adam. Experimental results show that the ST-Adam optimizer exhibits faster convergence and higher stability.

V. Evaluation

A. Experimental Setup

Testbed and Baselines. We employ a single NVIDIA Geforce RTX 3090 GPU as testbed. The code is executed on Ubuntu 20.04, using the framework of PyTorch version 1.9. To validate the performance of VisualMixer, we evaluate it against three advanced privacy-preserving methods: (a) InstaHide [24], received the 2020 Bell Labs Prize second place award, which is an obfuscation-based approach; (b) a differential privacy based method; (c) a federated learning method with differential privacy. These comparisons are presented in Table III.

Datasets and DNN Models. Table II present the datasets used in our work. Four representative datasets in the CV domain were selected for testing. Specifically, ImageNet-100 is a standard public dataset released by the well-known Kaggle contest [1]. And CIFAR-10 is a color image dataset consisting of 10 categories, with a total of 60,000 images; MNIST is a dataset for handwritten digit recognition, containing 60,000 training samples and 10,000 testing samples; AT&T dataset, also known as the ORL (Olivetti Research Lab) face database, consists of 400 grayscale face images belonging to 40 different individuals, with 10 images per individual. This dataset is commonly used for face recognition and face detection tasks. The selection criteria for the DNN models used in this work are as follows: (1) the test models should encompass a variety of mainstream frameworks for CV tasks, including Transformer structures (e.g., ViT-B [9] and Swin-T [35]), directly connected CNNs (e.g., AlexNet [29] and VGG [44]), and residual network models (e.g., ResNet [18] and DenseNet [23]); (2) the DNN models should cover network models with varying parameter scales and computational efforts, including very large networks like ViT-B, large networks like VGG, and lightweight networks like MobileNet [22] and ShuffleNet [52].

B. Validation on ST-Adam Optimizer

In this section, we conducted experiments to validate the performance of the ST-Adam optimizer. We trained models using both ST-Adam and widely used Adam optimizers on the CIFAR10, MNIST, and ImageNet-100 datasets. Throughout the training process, we recorded the accuracy and loss curves of the models for later comparison.

Based on the experimental results in Figure 6, the ST-Adam optimizer outperformed the Adam optimizer in terms of both accuracy and loss curves. This indicates that ST-Adam exhibits better optimization performance and faster convergence speed on these datasets. The reasons behind this superiority can be explained from the following three aspects.

C. Defend Against Privacy Leakages

Attack flow. Figure 7 illustrates the privacy attacks in the current landscape. In typical scenarios, users obfuscate their images before uploading them to the server for model training. In the threat model we defined in Section II-A, the server is a semi-honest adversary. Therefore, in this process, privacy primarily faces four types of threats. These attacks include exhaustive search, *i.e.*, brute-force crack, heuristic attacks by shredder recovering algorithm, weights leaking visual feature and membership inference on weights.

Exhaustive search. When attempting to restore VIM data through brute force cracking, we can analyze it using the principle of sorting. As explained in Section IV, the total number of permutations N of exhaustive search is calculated based on the probability ranking principle as $N = \sum_{i=0}^{\frac{wh}{WS^2}^2} (WS_i)^2!$. With the equation, we observe that when $\exists WS_i \geq 6$, the total number of permutations N exceeds 2^{128}, a number that can be reached regularly in our experiments. Thus, it becomes apparent that restoring an VIM image solely through brute force methods would necessitate an overwhelmingly large number of attempts, rendering it practically impossible.

Heuristic Attacks by Shredder Challenge Algorithm Based on the proposed design above, it is evident that restoring images processed by VisualMixer using brute-force methods is challenging. However, there is a possibility of attempting recovery using gradient information between pixels, which is used in Shredder Challenge Algorithm[27]. To test the resilience of VisualMixer, against this restoration method, we conducted experiments in this section.

The experimental results presented in Figure 8 demonstrate that the shredder recovering effect is not ideal. This is primarily due to the VIM process employed by VisualMixer, which alters the relative positions of pixels in both spatial and each channel, thereby disrupting the visual features and structural information of the original image. This means that the relative positions of the pixels in space and its channel have changed, thereby destroying the visual features and information structure of the original image. JigsawNet employs DNNs to construct latent relationships between sub-images, thereby attempting to piece together the original image [30]. It represents another heuristic attack method. However, such approaches typically perform well when the fragment size is larger. In our method, the fragment size is effectively 1, as we shuffle all pixels within the *WS*. As illustrated in the Figure 8, its attack efficacy is limited in our context.

Feature Restoration from Model Weights by GAN. We attempt to use the method presented in [20] to attack the weights from LeNet-like model obtained from training on the raw data (Celeba and MNIST), intermediate updates during federated learning, and training on the dataset processed by differential privacy($\sigma = 50$) and our VIM.

As shown in Figure 9, GAN methods can reconstruct many facial features from the original data model. It is similar for models with added DP noise in their weights. In the original federated learning approach, client-side trained intermediate updates are uploaded to the server, and facial features from client data can also be extracted. Our method not only protect dataset, but also effectively defends against such attacks, making it nearly impossible to recover identifiable features.

As shown in the figure, GAN methods can accurately reconstruct facial features from the original data model. The same holds true for models with added DP noise in their weights. In the original federated learning approach, client-side trained intermediate updates are uploaded to the server, and facial features from client data can also be extracted. Our method effectively defends against such attacks, making it nearly impossible to recover identifiable features.

Membership inference attacks. We also validate another attack method that specifically targets the training weights. This method, *i.e.*, membership inference, aims to determine whether a given dataset has been involved in the training process of the weights.

Figure 10 presents the impact of [42] method on MNIST and CIFAR-10 datasets. The purple curve indicates that [42] method can accurately infer on model trained by raw dataset, with a high probability, whether a given image belongs to the training set. On the other hand, the orange and red curves closely resemble the RG(Random Guess) curve. Other methods usually give different leakages of membership privacy.

D. Validating Training Performance

In this section, we evaluate the performance of our proposed approach in terms of accuracy loss and throughput introduced by privacy protection. We compare our method with baseline models, including data-level representative methods such as methods based on fully homomorphic encryption[39], as well as function-level representative methods like Differential Privacy and Federated Learning.

Accuracy. To comprehensively assess the impact of our method on the accuracy of the inference model, we utilize open-source datasets with varying input resolutions and color channels, including ImageNet-100, CIFAR10, and MNIST. Furthermore, we consider DNN models of different sizes and complexities, such as ViT-B, Swin-T, ResNet-50, ShuffleNet, MobileNet and VGG16.

The experimental results are presented in Table II are obtained through training and validating the models using the MNIST, CIFAR-10, and ImageNet-100 datasets. In general, on small datasets like MNIST, the accuracy loss is typically negligible. On larger datasets like ImageNet-100, our method

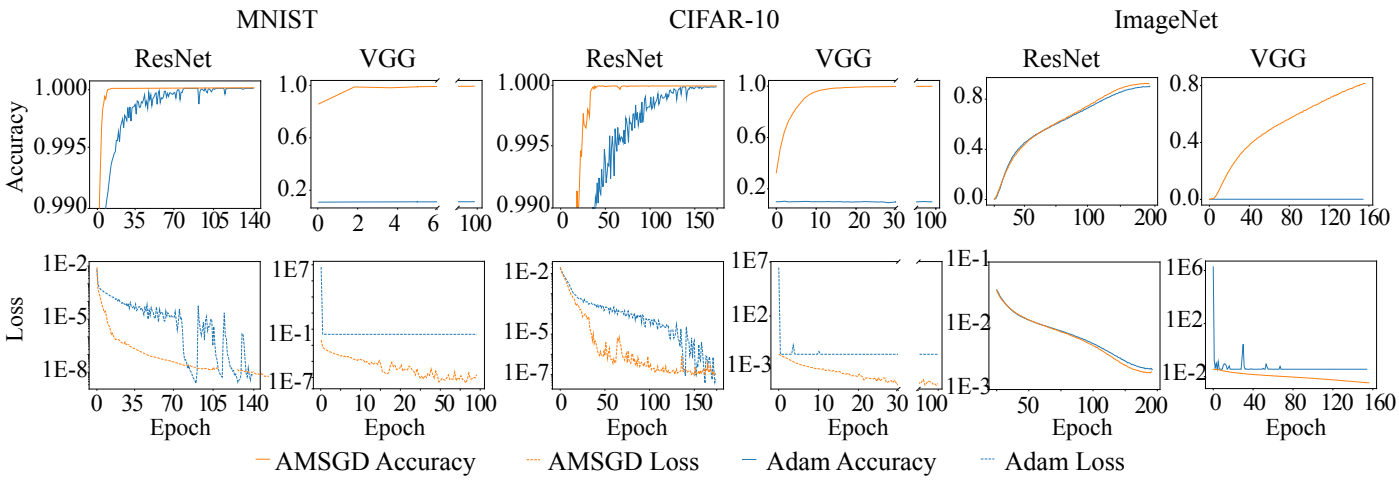

Fig. 6: The training curves of ST-Adam and Adam optimizers on MNIST, CIFAR-10, ImageNet.

TABLE II: Accuracy of trained models with different datasets.

Model	MNIST					CIFAR-10					ImageNet-100[5]			
Method	Plain	VIM	DP	FHE[4]	InstaHide [24]	Plain	VIM	DP	FHE[4]	InstaHide	Plain	VIM	DP	InstaHide
Privacy	✗	✓	○[1]	✓	○[7]	✗	✓	○[1]	✓	○[7]	✗	✓	○[1]	○[7]
ViT-B [9]	99.87%	99.14%	-[2]	-[4]	9.97%	98.63%	92.35%	-[2]	-[4]	10.03%	74.54%	72.98%	-[2]	1.03%
Swin-T [35]	98.72%	98.70%	-[2]	-[4]	10.16%	92.33%	85.73%	-[2]	-[4]	9.82%	84.80%	81.12%	-[2]	0.10%
ResNet [18]	99.27%	98.81%	61.36%[3]	-[4]	98.79%	97.23%	90.15%	62.74%[3]	87.84%[6]	90.04%	90.34%	83.78%	60.82%[3]	31.08%
ShuffleNet [52]	98.93%	97.19%	58.91%[3]	-[4]	96.27%	86.87%	84.07%	52.06%[3]	-[4]	84.97%	85.34%	83.64%	48.75%[3]	29.78%
MobileNet [22]	97.21%	97.20%	51.48%[3]	-[4]	97.13%	81.37%	81.02%	59.77%[3]	-[4]	75.53%	82.94%	81.38%	48.57%[3]	30.94%
VGG [44]	99.51%	98.12%	69.34%[3]	[4]	98.05%	82.64%	82.63%	53.89%[3]	84.76%[6]	82.57%	74.02%	73.88%	43.56%[3]	1.38%

[1] Here the σ of Gaussian noise is 50. As described in Section III-A, DP usually protect data membership, while it has limitation to protect releasing visual dataset.
[2] DP does not support the multi-head attention mechanism in the Transformer architecture.
[3] As DP does not support the batch norm layer in CNNs, it may influence accuracy.
[4] Due to too long execution time, we only provide data of accuracy that we can find in public papers.
[5] As ILSVRC's rule, we use Top-5 accuracy of models on the ImageNet dataset.
[6] Due to too long execution time, results of FHE are cited by [39] with VGG-7 and ResNet-20.
[7] The privacy of InstaHide [24] has been previously called into question [6].

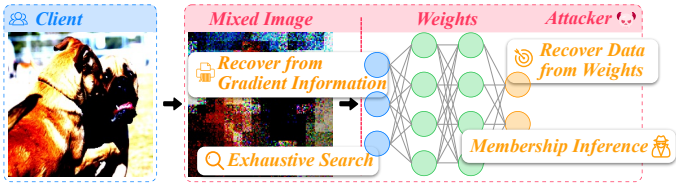

Fig. 7: The attack flowchart shows the possible parts that may be attacked: (1) Attacks against the shuffled images, such as exhaustive search restoration and restoration using gradient information; (2) Attacks against the training weights, such as GAN-based attack method [20] and membership inference attack method [42].

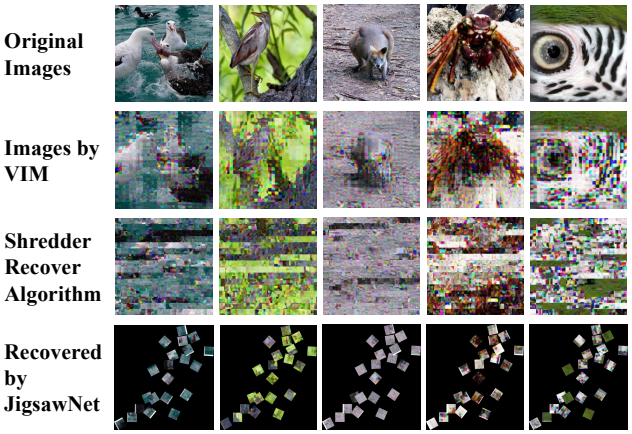

Fig. 8: Restoring results of the obfuscated images using shredder recovering algorithm by gradient information and [30].

exhibits significant advantages in terms of accuracy compared to DP. Even when compared to models trained on the original dataset, the accuracy loss of our method remains available. InstaHide [24] shows an accuracy similar to ours on the CNN model. However, it cannot support the Transfomer-based models well, which can be demonstrated by that both Vit-B and Swin-T show low accuracy in InstaHide. The homomorphic encryption, while preserving the model structure, necessitates the substitution of certain activation functions, leading to a decrease in accuracy. Its performance in terms of accuracy is inferior to our approach, but it is markedly superior to methods such as DP that alter the feature map and data.

TABLE III: Throughput (images per second) of different methods on different datasets.

Method	Privacy	ShuffleNet [52]	VGG [44]	ResNet [18]	MobileNet [22]	ViT-B [9]	Swin-T [35]
Plain	✗	1088.9	404.7	600.2	1070.8	322.2	472.3
DP [10]	◯[1]	212.3 [-80.5%]	66.1 [-83.7%]	187.8 [-68.7%]	92.1 [-91.4%]	-[2]	-[2]
FL[3][38]	◯[1]	291.8 [-73.2%]	385.2 [4.8%]	565.1 [5.8%]	1008.7 [-5.8%]	-[4]	-[4]
DP + FL[3] [10, 38]	◯[1]	10.9 [-99.0%]	2.0 [-99.5%]	7.4 [-98.8%]	12.8 [-98.8%]	-[4]	-[4]
FHE[5][39]	✓	0.006[-99.9%]	0.0009 [-99.9%]	0.0005[-99.9%]	0.005 [-99.9%]	0.000074 [-99.9%]	0.00045 [-99.9%]
InstaHide [24]	◯[6]	1087.1 [-0.17%]	399.2 [-1.4%]	594.1 [-1.0%]	1062.3 [-7.9%]	315.8 [-2.0%]	458.3 [-3.0%]
VIM	✓	1080.3 [-0.8%]	401.9 [-0.6%]	595.4 [-0.8%]	1062.1 [-0.8%]	319.9 [-0.7%]	466.1 [-1.3%]

[1] Both DP and AvgFL can only protect partial privacy while training[31].
[2] FL represents AvgFL[38]. This method also does not support transformer-based models.
[3] Federated learning employs virtual terminals, hence the network transmission delay can be considered as virtually non-existent.
[4] We employ a Python DP library[21] developed by IBM Research. This method does not support transformer-based models.
[5] Due to the excessive runtime of FHE, we calculate the average throughput within limited iterations.
[6] The privacy of InstaHide[24] has been previously called into question [6].

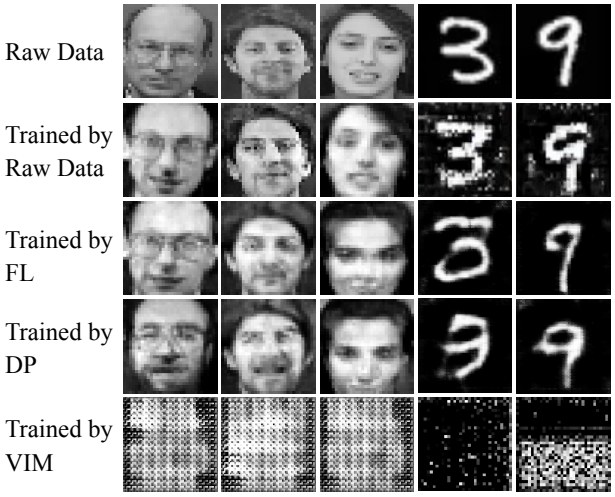

Fig. 9: Comparison of the attack effects on the model weights based on the GAN model. The results indicate that these attacks fail to recover any meaningful original information from the protected weights.

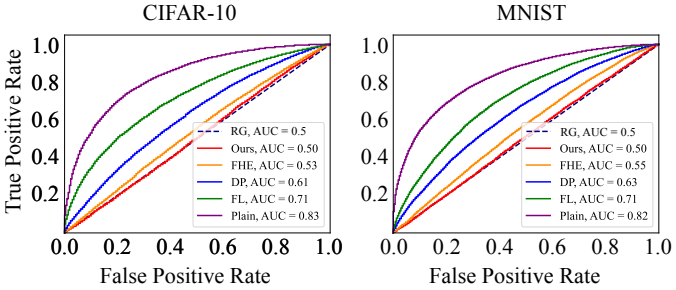

Fig. 10: ROC results of membership inference attack in [42] on the CIFAR-10 and MNIST datasets. The red curve represents ROC trained by VisualMixer. RG means random guess.

Throughput. We measured the training throughput of our method on different datasets and models in Table III. Our method involves offline processing of the dataset and does not modify the forward and backward computation processes of the model. Thus, it brings little additional computational overhead, similar to InstaHide [24]. As shown in Table III, our method and InstaHide [24] both achieve comparable throughput to the plain training process. Compared to DP-based method and FHE-based method, our method demonstrates significant performance improvement. However, it should be noted that in the case of ViT-B and Swin-T, an additional 50% of epochs were required for convergence, while maintaining the same number of epochs in other CNN models. The homomorphic encryption method requires the decomposition of underlying operators into supported operations to facilitate cryptographically-based procedures. This implies that data can be safeguarded through cryptographic measures without divulging almost any information. However, this also results in a substantial performance degradation, making it challenging for neural network accelerators, such as GPUs and NPUs, to enhance its computational speed. In terms of throughput, its performance is significantly inferior compared to other benchmarked approaches, even with its encryption-grade privacy.

TABLE IV: Federated learning accuracy of ResNet50 on ImageNet via plain scheme and VIM scheme.

Model	Method	MNIST	CIFAR10
ResNet [18]	FL	99.28%	70.83%
	FL+VIM	94.39%	66.11%
VGG [44]	FL	99.48%	77.29%
	FL+VIM	97.25%	76.87%
MobileNet [22]	FL	99.23%	75.26%
	FL+VIM	97.44%	73.11%
ShuffleNet [52]	FL	99.20%	72.63%
	FL+VIM	97.33%	72.08%
Swin-T [35]	FL	95.47%	67.53%
	FL+VIM	93.71%	61.37%
ViT-B [9]	FL	92.29%	60.03%
	FL+VIM	87.23%	57.70%

E. Validating DNN-Related Paradigms

In addition to its effectiveness in normal DNN training and inference, VisualMixer provides possibilities for various vision-based DNN-related tasks, including federated learning and knowledge distillation.

VisualMixer can be applied to the federated learning framework, improving data privacy. By shuffling the original data on each client before training, VisualMixer mitigates the risk

TABLE V: Knowledge distillation [19] accuracy of ResNet50 on ImageNet-100 via different training schemes.

Model	Top-1	Top-3	Top-5	Top-10
TO-Resnet50	74.55%	88.42%	92.02%	95.23%
TV[1]Resnet50	66.45%	82.67%	87.24%	91.78%
SO[2]MobileNetv3	21.1%	44.0%	53.6%	62.4%
SV-MobileNetv3	18.9%	43.1%	53.4%	62.4%

[1]**TV** is **T**eacher model with **V**IM. [2]**SO** is **S**tudent model with **O**riginal.

TABLE VI: Experimental results on VOC dataset.

Model	Method	Precision	Recall	mAP@50
YOLO v5 [26]	Plain	0.601	0.534	0.562
	VIM	0.602	0.415	0.441
SSD [34]	Plain	0.631	0.594	0.504
	VIM	0.556	0.418	0.372
EfficientDet [46]	Plain	0.817	0.660	0.765
	VIM	0.735	0.419	0.505

of sensitive information recovery, even in the event of model parameter leakage. The only concern we need to focus on is whether the accuracy loss caused by VIM is acceptable. Accuracy experimental results presented in Table IV showcase low accuracy loss. These results demonstrate the practical value and effectiveness of VisualMixer in the context of federated learning.

Furthermore, VisualMixer is leveraged within the knowledge distillation to ensure the privacy of data during the training process of teacher models, and protect privacy of teacher models during distillation.

Teacher models are often large and valuable models, which can potentially leak information about the dataset or features used to train them. Our method, as shown in Figure 9 and Figure 10, can protect the privacy of the model. In this section, we measure the potential accuracy loss that VIM may introduce during the knowledge distillation process. The experimental results, as illustrated in Table V, demonstrate ResNet50 achieving an accuracy of 97.3% accuracy, while the distilled MobileNet achieved a commendable accuracy of 62.4%, which is in close proximity to the accuracy achieved without VisualMixer processing. This highlights the effectiveness of VisualMixer within the knowledge distillation framework.

In conclusion, VisualMixer presents a promising solution to enhance data privacy in various DNN applications, including federated learning and knowledge distillation, while maintaining optimal model performance. It holds great potential in supporting diverse DNN scenarios.

F. Evaluation on Object Detection Tasks

Considering the underlying principles shared between object detection and image classification, the effectiveness of our proposed method in image classification tasks suggests its potential for performing object detection as well. The experimental results for object detection using our method are presented in Table VI.

According to the results, our encryption method demonstrates unique characteristics in the object detection task. While precision remains close to baseline levels, recall and mean average precision (mAP) show a decline. This performance disparity provides valuable insights into the interaction between the model and the obfuscated data. The relatively high precision suggests that the majority of bounding boxes identified by the model contain true positives. In other words, once the bounding box is established, the model can accurately classify the object within it. This aligns with the strong performance observed in classification tasks, indicating that the encryption method has minimal impact on the model's ability to classify objects within a bounded area.

The decrease in recall and mAP indicates that the model struggles to identify and establish bounding boxes around all instances of objects within the image. This results in a lower recall rate, as many true positives are not detected, and the model's effectiveness across various recall thresholds is diminished, leading to a lower mAP. This behavior may be attributed to the model's reliance on contextual information or background cues in classification tasks, which might be less effective in the object detection context, where precise delineation of object boundaries is required. However, this observation does not contradict the effectiveness of the encryption method in classification tasks. It merely highlights that while the model excels at accurately classifying objects within established bounding boxes (high precision), further fine-tuning or additional techniques may be necessary to enhance its ability to identify and establish bounding boxes around all relevant objects, thereby improving recall and mAP.

In conclusion, this experiment validates the applicability of our encryption method not only in classification tasks but also have potential in object detection tasks. Although there is a decrease in recall and mAP, the high precision emphasizes its practical value. These results further reinforce our earlier findings and establish the versatility of the proposed encryption method in various computer vision tasks.

VI. Related Work

As deep learning triggers the knowledge extraction capability from heterogeneous data, privacy preservation has been a significant concern and hot research topic for many years. Although existing works have demonstrated their effectiveness in different scenarios, how to protect the privacy of released visual data while preserve data utility is still an open challenge. We briefly review the related work in two categories.

A. Obfuscation-based Mechanisms

The dominating privacy-preserving mechanism for visual data is obfuscation, which is to obscure the private information (e.g., blurring [3] or blocking [49]) or add some noise (e.g., [41] and DP-based solutions [5, 11, 2, 55, 36, 8]) in the image before releasing it for analysis.

A significant problem in obscuring mechanisms is that they require perfectly accurate and comprehensive knowledge of the spatial locations of private information in the image. It is usually costly or even infeasible to semantically define what is private in different scenarios [5]. Besides, obscuring the private information of the image often leads to very low data utility,

making it useless for most vision tasks (like classification, detection, etc.).

Differential Privacy [11, 2] is another kind of obfuscation method. It preserves data privacy by adding noise, conforming to a specific distribution (like Laplace or Gaussian), to the original data. The key parameter in differential privacy is the privacy budget, which determines the amount of noise added. The advantage of differential privacy lies in its strict mathematical guarantee of data privacy under various data analysis attacks. This method has been successfully applied in many areas, e.g., social networks and medical information systems. Recently, Zhu, Yu, et al. [55] proposed a method avoiding splitting the training dataset and achieves comparable or better accuracy while reducing the privacy loss. Luo, Wu, et al [36] presented an approach minimizing trainable parameters, achieving commendable performance in extensive experiments on diverse visual recognition tasks. However, despite the competitive performance of differential privacy in many fields, it faces some challenges in computer vision, particularly in releasing image data [37]. In protecting the privacy of image data, differential privacy needs to add a considerable amount of noise to ensure every pixel in the image is sufficiently protected. However, for human visual system, even with the addition of substantial noise in the image, we can still recognize the main content of the image. This is because human visual system is highly sensitive to edge and texture information in the image, constituting the principal elements of visual features. Thus, although differential privacy can decrease privacy risk mathematically like data membership, it may fail to effectively prevent humans from extracting useful information from image data, especially when specific areas in the image contain sensitive information. It should be noted that the latest work by Google DeepMind [8] proposes a DP based privacy-preserving methods that can make some NF-model accuracy comparable to our solution. However, it still cannot avoid the aforementioned privacy issues, especially when it comes to the release of the dataset.

The k-anonymity algorithm [45, 12] can also be regarded as a data obfuscation method that protects data privacy by making records in a dataset consistent on certain attributes. Thus, any record is at least identical to $k-1$ other records in these attributes. Nevertheless, the application of the k-anonymity algorithm also has its limitations, because selecting suitable attributes for consistent processing is not a simple task for complex visual data.

Federated Learning (FL) [28, 15] has also emerged to be a feasible privacy preservation mechanism. It protects the data privacy by locally training a model via private data, and shares the model parameters instead of the original data to share knowledge. Although many studies have pointed out the privacy risks in FL[31], its design philosophy is orthogonal from our solution. In our experiments, we also show that applying our solution to FL, we can provide a stronger protection strength to resist the privacy attacks on uploaded local model parameters.

B. Encryption Based Mechanisms

There are numerous studies focusing on data privacy protection via encrypting the original data or using Trusted Execution Environment (TEE) to isolate private data computing.

Fully Homomorphic Encryption (FHE) [14, 47] is a cryptographic technique that allows arbitrary calculations on ciphertext without needing to decrypt it first. Through FHE, data owners can encrypt their data and send it to cloud service providers for processing, without worrying about data privacy leakage. However, while FHE is wonderful in theory, we are still facing some challenges in practice. The primary issue is computational complexity. FHE algorithms require a substantial amount of computational resources when executing encryption and decryption operations. In particular, arithmetic operations under FHE often have an exponential level of complexity. It usually bring 10x-10,000x additional time overhead on same hardware. Therefore, FHE might become a bottleneck for large-scale image data processing. When handling large-scale data, the computational complexity of FHE could make computation time and resource consumption excessively large, which limits the application of FHE in the field of image data processing.

Another method involves using TEE [40, 7], like Intel's SGX and ARM's TrustZone, which can protect the data being processed at the hardware level, preventing unauthorized access and modification. However, the hardware resources of TEE is usually limited and its compatibility to neural network accelerators (e.g., GPU and NPU) is still far from satisfaction, which significantly impacts the performance of DNN computing.

VII. Discussion

We acknowledge that, compared to traditional encryption methods, VisualMixer lacks a formal security proof to prove its effectiveness against heuristic attacks. VisualMixer focuses on reducing the visual recognizability of images while enhancing their usability within DNNs. However, as demonstrated by our experiments, VisualMixer is capable of resisting state-of-the-art image restoration attacks. Unlike Differential Privacy (DP) or InstaHide [24], VisualMixer does not introduce external noise in image obfuscation. The advantage of such an intrinsic transformation is that it is not susceptible to the attacks based on statistical features.

Moreover, VisualMixer can be compatible with other privacy-preserving methods. In Section V, we validate the compatibility between VisualMixer and Federated Learning (FL). The experiment results therein illustrate that VisualMixer can mitigate privacy issues caused by gradient leakage in FL with minimal accuracy loss. Also, VisualMixer is orthogonal to Differential Privacy (DP). Thus, VisualMixer can enhance DP by further obfuscating images and feature maps without significant accuracy decrease. Note that, VisualMixer is compatible with FHE and obfuscates data before data encryption. However, as FHE already offers very stringent security guarantees, integrating VisualMixer may not be necessary.

VIII. Conclusion

In this paper, we introduced VisualMixer, an innovative and effective strategy for protecting data privacy in the training and inference processes of DNNs. The primary objective of VisualMixer is to enhance data privacy while preserving the performance of the model. To address the inherent trade-off between privacy and model accuracy in data obfuscation strategies, we introduce a new metric, named VFE, to measure the

visual privacy, and propose a non-uniform shuffling strategy VisualMixer to pre-process the image dataset. VisualMixer allows for effective data obfuscation while ensuring the output deviation of the model does not exceed a preset threshold. We also devised an optimizer, named ST-Adam, to tackle potential dense gradient issues during training. Extensive experiments demonstrate VisualMixer's capability to enhance data privacy without compromising the overall performance of the model. In addition, we have identified the potential of VisualMixer in both federated learning and knowledge distillation frameworks, showcasing its adaptability. In summary, VisualMixer presents a promising data privacy protection method for secure training and inference within DNNs. Our ongoing and in-depth research and exploration of VisualMixer aim to unlock further possibilities in data privacy protection and provide more safeguards for secure training and inference.

ACKNOWLEDGMENT

This research was supported in part by the National Key R&D Program of China under Grant No. 2022YFF0604502, the National Natural Science Foundation of China under Grant No. 62122095, 62341201, 62132011, 62072472 and U19A2067, the China Postdoctoral Science Foundation under Grant No. 2022M721827, a grant from Kuaishou Technology, and by a grant from the Guoqiang Institute, Tsinghua University.

REFERENCES

[1] "ImageNet100," https://www.kaggle.com/datasets/ambityga/imagenet100, [Accessed 11-10-2023].

[2] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 308–318.

[3] P. Aditya, R. Sen, P. Druschel, S. Joon Oh, R. Benenson, M. Fritz, B. Schiele, B. Bhattacharjee, and T. T. Wu, "I-pic: A platform for privacy-compliant image capture," in *Proceedings of the 14th annual international conference on mobile systems, applications, and services*, 2016, pp. 235–248.

[4] A. Boulemtafes, A. Derhab, and Y. Challal, "A review of privacy-preserving techniques for deep learning," *Neurocomputing*, vol. 384, pp. 21–45, 2020.

[5] F. Cangialosi, N. Agarwal, V. Arun, S. Narayana, A. Sarwate, and R. Netravali, "Privid: Practical, Privacy-Preserving video analytics queries," in *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*. Renton, WA: USENIX Association, Apr. 2022, pp. 209–228. [Online]. Available: https://www.usenix.org/conference/nsdi22/presentation/cangialosi

[6] N. Carlini, S. Deng, S. Garg, S. Jha, S. Mahloujifar, M. Mahmoody, A. Thakurta, and F. Tramèr, "Is private learning possible with instance encoding?" in *2021 IEEE Symposium on Security and Privacy (SP)*, 2021, pp. 410–427.

[7] Y. Chen, F. Luo, T. Li, T. Xiang, Z. Liu, and J. Li, "A training-integrity privacy-preserving federated learning scheme with trusted execution environment," *Information Sciences*, vol. 522, pp. 69–79, 2020.

[8] S. De, L. Berrada, J. Hayes, S. L. Smith, and B. Balle, "Unlocking high-accuracy differentially private image classification through scale," *arXiv preprint arXiv:2204.13650*, 2022.

[9] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[10] C. Dwork, "Differential privacy," in *Automata, Languages and Programming: 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II 33*. Springer, 2006, pp. 1–12.

[11] ——, "Differential privacy," in *Automata, Languages and Programming: 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II 33*. Springer, 2006, pp. 1–12.

[12] B. Gedik and L. Liu, "Protecting location privacy with personalized k-anonymity: Architecture and algorithms," *IEEE Transactions on Mobile Computing*, vol. 7, no. 1, pp. 1–18, 2007.

[13] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients-how easy is it to break privacy in federated learning?" *Advances in Neural Information Processing Systems*, vol. 33, pp. 16 937–16 947, 2020.

[14] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the forty-first annual ACM symposium on Theory of computing*, 2009, pp. 169–178.

[15] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, "An efficient framework for clustered federated learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 19 586–19 597, 2020.

[16] C. Guo, Q. Ma, and L. Zhang, "Spatio-temporal saliency detection using phase spectrum of quaternion fourier transform," in *2008 IEEE conference on computer vision and pattern recognition*. IEEE, 2008, pp. 1–8.

[17] N. Haim, G. Vardi, G. Yehudai, O. Shamir, and M. Irani, "Reconstructing training data from trained neural networks," *arXiv preprint arXiv:2206.07758*, 2022.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[19] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[20] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the gan: information leakage from collaborative deep learning," in *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, 2017, pp. 603–618.

[21] N. Holohan, S. Braghin, P. Mac Aonghusa, and K. Levacher, "Diffprivlib: the IBM differential privacy library," *ArXiv e-prints*, vol. 1907.02444 [cs.CR], Jul. 2019.

[22] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[23] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[24] Y. Huang, Z. Song, K. Li, and S. Arora, "InstaHide: Instance-hiding schemes for private distributed learning," in *Proceedings of the 37th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, H. D. III and A. Singh, Eds., vol. 119. PMLR, 13–18 Jul 2020, pp. 4507–4518. [Online]. Available: https://proceedings.mlr.press/v119/huang20i.html

[25] S. A. Hussein, T. Tirer, and R. Giryes, "Image-adaptive gan based reconstruction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, 2020, pp. 3121–3129.

[26] G. Jocher and Ultralytics, "Yolov5," Github repository, 2020. [Online]. Available: https://github.com/ultralytics/yolov5

[27] Jou, "Darpa shredder challenge 2011," https://www.ee.columbia.edu/ln/dvmm/shredder/.

[28] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[29] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[30] C. Le and X. Li, "Jigsawnet: Shredded image reassembly using convolutional neural network and loop-based

composition," *IEEE Transactions on Image Processing*, vol. 28, no. 8, pp. 4000–4015, 2019.

[31] Q. Li, W. Zhu, C. Wu, X. Pan, F. Yang, Y. Zhou, and Y. Zhang, "Invisiblefl: Federated learning over non-informative intermediate updates against multimedia privacy leakages," in *Proceedings of the 28th ACM International Conference on Multimedia*, ser. MM '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 753–762. [Online]. Available: https://doi.org/10.1145/3394171.3413923

[32] Q. Li, J. Ren, X. Pan, Y. Zhou, and Y. Zhang, "Enigma: Low-latency and privacy-preserving edge inference on heterogeneous neural network accelerators," in *2022 IEEE 42nd International Conference on Distributed Computing Systems (ICDCS)*, 2022, pp. 458–469.

[33] B. Liu, M. Ding, H. Xue, T. Zhu, D. Ye, L. Song, and W. Zhou, "Dp-image: Differential privacy for image data in feature space," *ArXiv*, vol. abs/2103.07073, 2021.

[34] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*. Springer, 2016, pp. 21–37.

[35] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.

[36] Z. Luo, D. J. Wu, E. Adeli, and L. Fei-Fei, "Scalable differential privacy with sparse network finetuning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 5059–5068.

[37] A. Machanavajjhala, X. He, and M. Hay, "Differential privacy in the wild: A tutorial on current practices & open challenges," in *Proceedings of the 2017 ACM International Conference on Management of Data*, 2017, pp. 1727–1730.

[38] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[39] S. Meftah, B. H. M. Tan, C. F. Mun, K. M. M. Aung, B. Veeravalli, and V. Chandrasekhar, "Doren: Toward efficient deep convolutional neural networks with fully homomorphic encryption," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 3740–3752, 2021.

[40] F. Mo, H. Haddadi, K. Katevas, E. Marin, D. Perino, and N. Kourtellis, "Ppfl: privacy-preserving federated learning with trusted execution environments," in *Proceedings of the 19th annual international conference on mobile systems, applications, and services*, 2021, pp. 94–108.

[41] Z. Qi, A. MaungMaung, Y. Kinoshita, and H. Kiya, "Privacy-preserving image classification using vision transformer," 2022.

[42] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *2017 IEEE symposium on security and privacy (SP)*. IEEE, 2017, pp. 3–18.

[43] ——, "Membership inference attacks against machine learning models," in *2017 IEEE symposium on security and privacy (SP)*. IEEE, 2017, pp. 3–18.

[44] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[45] L. Sweeney, "Achieving k-anonymity privacy protection using generalization and suppression," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 05, pp. 571–588, 2002.

[46] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and efficient object detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 10 781–10 790.

[47] M. Van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, "Fully homomorphic encryption over the integers," in *Advances in Cryptology–EUROCRYPT 2010: 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30–June 3, 2010. Proceedings 29*. Springer, 2010, pp. 24–43.

[48] A. Voulodimos, N. Doulamis, A. Doulamis, E. Protopapadakis *et al.*, "Deep learning for computer vision: A brief review," *Computational intelligence and neuroscience*, vol. 2018, 2018.

[49] J. Wang, B. Amos, A. Das, P. Pillai, N. Sadeh, and M. Satyanarayanan, "A scalable and privacy-aware iot service for live video analytics," in *Proceedings of the 8th ACM on Multimedia Systems Conference*, 2017, pp. 38–49.

[50] Y. Wen, B. Liu, M. Ding, R. Xie, and L. Song, "Identitydp: Differential private identification protection for face images," *Neurocomputing*, vol. 501, pp. 197–211, 2022.

[51] H. Wu, D. Li, and M. Becchi, "Compiler-assisted workload consolidation for efficient dynamic parallelism on gpu," in *2016 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE, 2016, pp. 534–543.

[52] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6848–6856.

[53] B. Zhao, K. R. Mopuri, and H. Bilen, "idlg: Improved deep leakage from gradients," *arXiv preprint arXiv:2001.02610*, 2020.

[54] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," *Advances in neural information processing systems*, vol. 32, 2019.

[55] Y. Zhu, X. Yu, M. Chandraker, and Y.-X. Wang, "Private-knn: Practical differential privacy for computer vision," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 854–11 862.

APPENDIX A
VIM-ED SAMPLE

There are some random samples of plain and VIM-ed images of ImageNet-100 dataset.