Task-Oriented Training Data Privacy Protection for Cloud-based Model Training

Zhiqiang Wang, Jiahui Hou[*], Haifeng Sun, Jingmiao Zhang, Yunhao Yao, Haikuo Yu, and Xiang-Yang Li[*]
zhiqiang.wang@mail.ustc.edu.cn, jhhou@ustc.edu.cn,
{sun1998, jingmiao, yaoyunhao, yhk7786}@mail.ustc.edu.cn, xiangyangli@ustc.edu.cn
University of Science and Technology of China

Abstract

Cloud-based model training presents significant privacy challenges, as users must upload personal data for training high-performance models. Once uploaded, this data goes beyond the user's control and could be misused for other purposes. Users need tools to control the usage scope of the uploaded training data, preventing unauthorized training without compromising authorized training. Unfortunately, existing solutions overlook this issue.

In this paper, we propose and achieve a unique privacy-utility goal tailored for cloud-based model training, considering both user demand and legal requirements. Our approach provides task-level control of training data usage, simultaneously ensuring each protected data exhibits noticeable visual changes to address fundamental privacy concerns. We introduce carefully designed noise to each training data for privacy protection. These noises are designed to provide visual protection while minimizing the shifts in the feature domain through adversarial optimization. By adjusting the correlation between noise and class labels, we guide the model to learn the correct features for the target task while preventing unauthorized privacy task training. Additionally, we introduce the overflow matrix for compatibility with existing encoding and transmission frameworks. Real-world experiments demonstrate that it can simultaneously protect visual privacy (SSIM is 0.028) and prevent unauthorized model training (protection success rate achieved 100%), while the accuracy of the target task model is slightly reduced by about 1.8%.

1 Introduction

Cloud-based model training provides a scalable and efficient approach for training machine learning models. It allows users to utilize the computational resources of the cloud for training task models by uploading their training data [1–3]. In such a scenario, the cloud collects users' personal data and then trains a task model based on user requirements, subsequently sending well-trained models to users. It is crucial to emphasize that the data used for model training is not a public dataset but the user's private data. However, once the private data is uploaded to the cloud, it surpasses the user's control and can give rise to significant privacy issues and concerns [4–6].

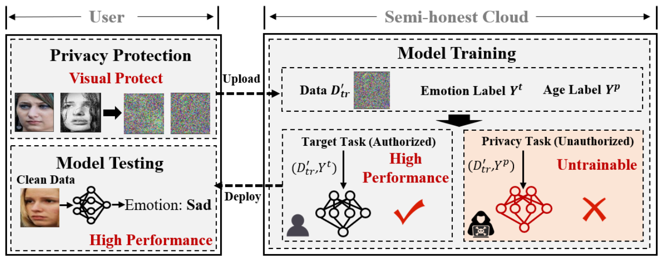


Figure 1: Unique privacy-utility goal tailored for cloud-based model training. Users hope that their training data uploaded to the cloud will be protected (fundamental visual privacy and misused for training unauthorized tasks) without compromising the training of the target task models. We generate protected versions of training data with noticeable visual variations (Visual Protect) while providing users with task-level control of the training data usage scope (High performance for target task-Untrainable for privacy task).

Privacy protection is more challenging in cloud-based model training scenarios due to its complex privacy-utility requirements. As illustrated in Figure 1, users desire to utilize the cloud for training a high-performance target task model while ensuring fundamental visual privacy protection *(User Demands)*. Simultaneously, they have the right to control their data usage scope, allowing certain tasks to be trainable while others are not (*Legal Requirements*). Based on this, we propose a unique privacy-usability goal specifically for cloud-based model training, which safeguards visual privacy and provides task-level control over data usage.

Existing privacy protection for cloud-based model inference [7–10] primarily focus on key features crucial for decision-making in pre-trained models only, would overlook

[*]Corresponding authors.

diverse and complex data patterns essential for robust model training [11]. Privacy-preserving model training (federated learning [12–14], differential privacy [15, 16], and secure multi-party computation [17, 18]) and augmentation [19] mainly focus on generating ML models (without the leakage of raw training data) rather than creating protected datasets. These methods couple privacy protection with model training, requiring users to bear the burden of training on their devices. Even though some local differential privacy methods can generate protected datasets for sharing, most of them cannot satisfy strict differential privacy guarantees and have severe utility issues [20, 21]. Unlearnable Examples [22–25] aim to prevent attackers from training commercial models, without compromising the visual quality. However, they do not guarantee usability for authorized target tasks and fail to address visual privacy concerns. Dataset distillation [26–29] focuses on extreme compression ratio, inevitably leading to significant accuracy decreases. Besides, finding a balance between privacy and model accuracy is challenging for dataset distillation. Alternatively, we propose a privacy protection mechanism that generates a protected version of the training dataset (with discernible changes to protect visual privacy). This protected dataset can be used to train target tasks while preventing unauthorized training for private attributes (privacy tasks). The target task model trained using the protected data exhibits excellent performance in the deployment environment with clean data. Ultimately, by generating protected versions of the training datasets, we can decouple model training from privacy protection. It allows users to offload the intensive and repetitive training process (model architecture selection and hyperparameter tuning) entirely to the cloud without changing existing training pipelines. A detailed comparison with existing works is shown in Table 1.

Challenge. There are four challenges we should address: **C1 Distribution Mismatch.** Privacy protection on the training data may cause a discrepancy between the distribution of the training set and the clean test set, resulting in models that perform well on the training set but underperform when deployed in real-world scenarios. **C2 Task-Oriented Data Usability Control.** The protected data itself should exhibit different training usability for different tasks, rather than through modifications to the training process. It should be trainable for the target task but untrainable for privacy tasks. **C3 Model Adaptability and Transferability.** The model structure utilized in training can change, and the model parameters during training are variable. The protected data should exhibit strong cross-model transferability. **C4 Encoding Compatibility.** After protection, the pixel values may exceed the standard 0-255 range (for image data), rendering them incompatible with existing image encoding and transmission frameworks.

We achieve privacy protection by introducing carefully crafted pixel-level noise to each training data. Firstly, we generate noises that provide visual protection while minimizing the shifts in the feature domain through adversarial optimization (C1). Specifically, we primarily focus on the low-level features of the CNN feature extractor for better transferability (C3). Meanwhile, we guide the model training by adjusting the correlation between noise and class labels for task-oriented training usability. Based on this, we also designed a plug-and-play module to enhance protection against unauthorized model training by embedding an incredibly tiny amount of class-wise noise (C2). Finally, we propose the Overflow Matrix M_o to address the overflow issue during real-world image transmission and storage (C4).

Contributions. We make the following contributions:

- We propose a novel privacy-usability goal tailored for cloud-based model training, considering both user demand and legal requirements. Then, we designed a training data protection mechanism that enables task-level control of data training scope while achieving efficient visual privacy protection.
- Our approach introduces an adversarial optimization-based noise generation mechanism and two plug-and-play privacy-enhanced modules to generate protected data for each training data, decoupling model training from privacy protection. With only one-time protection (for given tasks), we can enable computationally intensive and repetitive training processes to be performed entirely on the cloud.
- Real-world implementation demonstrates the practicality and effectiveness of our approach. The target task's training usability is only affected by approximately 1.7% with strong visual privacy protection (SSIM<0.028). Meanwhile, the protected dataset can achieve a protection success rate of over 100% for privacy task training.

2 Related Work

2.1 Privacy for Cloud-based Model Inference

This type of work shares a similar background with our study, as users aim to utilize cloud resources while ensuring protection against potential misuse by untrusted cloud.

Explainable AI-based Methods. These works such as [9] leverage the interpretability of the model, introducing noise or confusion to the unimportant features. This can protect privacy while minimizing the impact on the model's accuracy.

Adversarial-based Methods. In such work [30, 31], the attacker and task are typically given. Subsequently, an adversarial network is utilized to simulate both the task and the attacker, generating images that meet the task requirements and demonstrate resilience against the specific attack. Moreover, [7] protects the detailed information of the video based on the Generative Adversarial Networks, ensuring visual privacy without compromising large-scale detection tasks.

Frequency Domain-based Methods. Some works explored the trade-off between visual privacy and model in-

Table 1: Privacy-Utility Comparison with Existing Works.

Method	**Protect Target**[*]	**Train Usability**	**Visual Privacy**	**Control of Data Usage Scope**[†]
Privacy for Cloud-based Model Inference	DS	○	●	○
Privacy-preserving Model Training	M	●	●	○
Augmentation	M	●	○	○
Unlearnable Examples	DS	○	○	◐
Dataset Distillation	DS	◐	◐	○
Ours	DS	●	●	●

1. ●: Yes; ○: No; ◐: Partial, which means a significant gap exists between the current state and the desired goal.
2. *: Generate ML model without leaking training data (M). Generate protected dataset (DS). DS incurs lower overhead for users by decoupling model training from privacy protection, enabling computationally intensive and repetitive training processes to be performed entirely on the cloud.
3. †: The protected data can exhibit different training usability based on user demands. We can achieve task-level control, which is suitable for black-box cloud-based model training compared to model-level control.

ference by leveraging the differences between human perception and model feature extraction. [32] investigated this difference first, highlighting that human eyes tend to focus more on low-frequency segmentation information, while models can extract higher-frequency information to accomplish classification tasks. [10, 33] leveraged this characteristic by introducing confusion to the low-frequency information of images, thereby achieving visual privacy protection without compromising the model's ability to recognize faces.

However, They primarily concentrate on privacy protection for inference tasks. Although the protected data can preserve the inference accuracy for specific tasks, it would overlook diverse and complex data patterns essential for robust model training. [33] shares a similar objective to our research: to train task models with privacy-protected data. The difference is that their approach requires applying the same privacy protection measures to the test data. However, it is essential to highlight that the testing environment is under user control, thereby eliminating the necessity for additional privacy protection.

2.2 Privacy-Preserving Model Training

Privacy-preserving model training aims to safeguard raw training data while generating well-trained ML models. Federated learning [12–14] enables distributed training by performing local training on individual devices and aggregating models on the cloud. Secure multi-party computation [17, 18] allows participants to update model parameters without sharing raw data. These mechanisms indeed offer effective protection for the privacy of raw training data. However, they focus on privacy protection during model training, ultimately producing ML models rather than protecting datasets. This demands participants' capability to fully train models (Horizontal Federated Learning [34]) or partially train models (Vertical Federated Learning [35]). When the model architecture or training parameters change, privacy-preserving training must be repeated, incurring significant overhead. Thus, they have limited applicability in cloud-based model training.

2.3 Unlearnable Examples

Unlearnable samples aim to protect data privacy by rendering them unsuitable for training specific models. [22] utilized a min-min optimization process, opposite to adversarial learning, to generate unlearnable samples. This process ensures that deep learning models cannot use those samples for training. [23] employed a carefully designed cloak to obfuscate the image distribution at the feature level, thereby preventing unauthorized model training. [24] added imperceptible class-wise noise to images based on causal inference theory. These methods primarily concentrate on image sharing in social networks, aiming to achieve unlearnable without causing significant visual alterations. However, users in cloud-based model training scenarios are concerned principally about visual privacy, which implies that the protected images should exhibit noticeable visual alterations. The aforementioned unlearnable samples do not meet our requirements.

2.4 Dataset Distillation

Data distillation aims to generate a small synthetic dataset (which can be viewed as the protected dataset). It ensures that the model trained on the synthetic dataset performs similarly to the model trained on the original dataset by matching a series of metrics (such as distribution matching [27], training trajectory matching [28], gradient matching [29], etc). [36] uses dataset distillation to protect training datasets, thereby achieving privacy-preserving remote training. [37] introduce differential privacy in the distillation process to enhance privacy protection. However, finding a balance between privacy protection and model accuracy in data distillation is challenging. At high compression rates, the model accuracy significantly decreases. Conversely, low compression rates lead to the distilled (protected) data being very similar to the original data yet facing accuracy bottlenecks. Similarly, the aforementioned privacy protection methods based on data distillation also struggle with significant accuracy issues.

3 Overview

3.1 Threat Model

Our system primarily involves two participants: **User** and **Cloud**.

User. Users commonly have limited computational resources, such as personal computers. They own personal training data and aim to train target task models using this dataset. However, the limited computational resources prevent them from completing model training independently on their local device. Consequently, they upload their data to the cloud and rely on it to train the model, thereby obtaining a well-trained model from the cloud. The testing environment for the model is users' local devices, which are considered secure and trusted. Users prefer not to allocate their limited resources to privacy protection for local testing data. Generally, users (or laws) have specific privacy requirements that must be protected. We customize privacy tasks based on these requirements. This blacklist strategy is practical because it is unrealistic to know all the potential attacks.

Cloud. Cloud, considered semi-honest, is responsible for training authorized task (Target Task) models using data provided by users. However, in addition to fulfilling the authorized tasks, there is a potential risk of the cloud misusing the uploaded data, thereby raising significant privacy concerns. Traditional encryption-based access control methods are not suitable for a semi-honest cloud, as the cloud is an authorized legitimate identity. We primarily focus on two types of potential privacy breaches in the cloud:

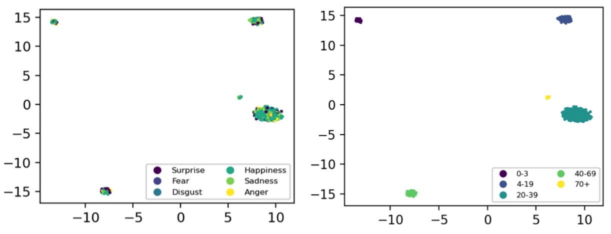
Visual Privacy Leakage. Visual privacy leakage from human observation is the most intuitive form of privacy breach and represents a fundamental privacy concern for users.

Unauthorized model training. Besides training the target task model, the cloud can potentially train other unauthorized commercial models using the training data without the user's knowledge. We assume that the cloud attacker possesses sufficient attack capabilities to access all attribute labels of the training dataset (By assuming a higher level capability of attackers, we can provide more robust protection against privacy breaches). Our methods can still achieve adequate protection even under this strong attack assumption.

3.2 Privacy & Usability Goals

Our privacy-utility goals are tailored specifically for non-collaborative cloud-based model training (users upload datasets to obtain well-trained models), considering both user demands and legal requirements (shown in Figure 1):

Usability. Users require a task model with negligible utility degradation when deploying in user ends (*User Demands*) – Models trained by the protected dataset can retain high accuracy with the clean testing dataset for authorized target tasks (Eq. (2)).



(a) Class-agnostic for Target Task (b) Class-wise for Privacy Task

Figure 2: Visualization of noise through UMAP [38] dimensionality reduction. Different colors correspond to different class labels. Class-wise noise exhibits good clustering effects for the same label.

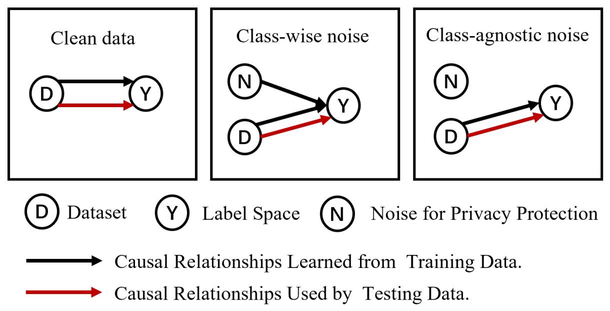

Clean data
Class-wise noise
Class-agnostic noise
D
Y
N
Dataset
Label Space
Noise for Privacy Protection
Causal Relationships Learned from Training Data.
Causal Relationships Used by Testing Data.

Figure 3: Casual graph of model training and testing. Causal relationships refer to the direct influence of one variable on another. Their consistency is crucial for an ML model's generalizability across different data distributions [39, 40].

Privacy. 1) Users need abilities to control their data usage scope (*Legal Requirements*) – The protected dataset should allow certain tasks to be trainable (Target Task) while others are not (Privacy Task). 2) If the protected images show little visible differences, convincing users that their privacy is protected is challenging (*User Demands*) – The noise applied for protection should be significant enough to achieve visual privacy protection for human observation (Eq. (3)).

Therefore, we propose the unique privacy-utility goal for cloud-based model training, which both safeguards visual privacy and controls the usage of training data by setting different tasks. We believe this task-oriented protection is more user-friendly and practical. Users can test different model architectures while training models in the cloud without repeatedly regenerating the protected dataset.

3.3 Problem Formalization

Task Definition. Generally speaking, a machine learning task typically involves two phases: 1) Model Training, where a model f is trained using training data and labels ($f_{D_{tr}\to Y} = T(D_{tr}, Y),\ Y \in \mathbf{Y}$) and 2) Model Inference/Test, where the model's accuracy is evaluated on a test dataset ($E(f_{D_{tr}\to Y}, D_{te}, Y),\ Y \in \mathbf{Y}$). D_{tr} and D_{te} denote the data used for training and testing, respectively. Similarly, $T(\cdot)$ and $E(\cdot)$

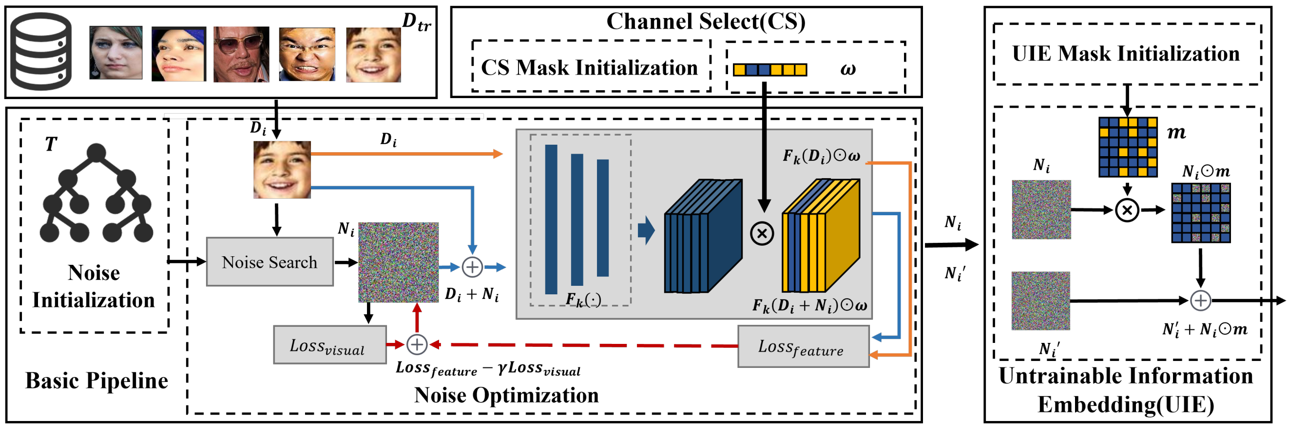


Figure 4: System overview. Our system consists of a **Basic Pipeline** and two plug-and-play privacy enhancement modules: **Channel Selection (CS)** and **Untrainable Information Embedding (UIE)**. The **Basic Pipeline** built upon two carefully designed losses for adversarial optimization, aiming to generate noise that satisfies visual privacy protection without affecting the training of the target task model. The **CS** reduces potential privacy leakage risks in intermediate features through channel importance screening. The **UIE** guides the privacy task model to learn noise by embedding class-wise noise, further preventing unauthorized model training.

denote the training and testing methods, respectively. $\mathbf{Y}$ represents the label space with multiple dimensions, as each data can have various annotations across different dimensions. For example, for a facial image, $\mathbf{Y}$ might include dimensions such as emotion, race, age, gender, and more. Therefore, we define the task as a triplet $(D, Y, f_{D_{tr}\to Y_{tr}})$, where $D = D_{tr} \cup D_{te}$ represents the data domain, and $f_{D_{tr}\to Y}$ represents the learned task model mapping. Given the dataset D, we categorize tasks based on label dimensions, achieving different training usability for the same dataset (trainable on target label dimensions $\mathbf{Y^t}$ (e.g., emotions), while untrainable on private label dimensions $\mathbf{Y^p}$ (e.g., race). Specifically, the definitions of the target task and privacy task are as follows:

Target Task $(D, Y, f_{D_{tr}\to Y})$, $Y \in \mathbf{Y^t}$**.** Tasks explicitly declared by users to be completed by the cloud.

Privacy Task $(D, Y, f_{D_{tr}\to Y})$, $Y \in \mathbf{Y^p}$**.** Tasks that involve the privacy attribute labels declared by the user (or laws).

Our task-driven training data privacy protection is defined as generating the significantly changed protected version D'_{tr} of D_{tr} (Eq. (1)), which needs to fulfill the requirements of both usability (D'_{tr} can be used to train target task, Eq. (2)) and privacy (training the privacy task by D'_{tr} would fail, Eq. (3)). $f_{D_{tr}\to Y}$ and $f_{D'_{tr}\to Y}$ utilize the same training process $T(\cdot)$, which includes the same model structures, hyperparameters, etc. The only difference lies in the training data.

$$||D_i - D'_i||_2 > \delta,\ D_i \in D_{tr}, D'_i \in D'_{tr}. \tag{1}$$

$$E(f_{D'_{tr}\to Y}, D_{te}, Y) \sim E(f_{D_{tr}\to Y}, D_{te}, Y),\ Y \in \mathbf{Y^t}. \tag{2}$$

$$E(f_{D'_{tr}\to Y}, D_{te}, Y) << E(f_{D_{tr}\to Y}, D_{te}, Y),\ Y \in \mathbf{Y^p}. \tag{3}$$

3.4 Fundamental Idea

We apply carefully designed noise to each training data to generate a new protected dataset, decoupling model training from privacy protection. Furthermore, we utilize the correlation between noise and task labels to guide the model training, aiming to achieve different training usability across different tasks. Specifically, building upon the work presented in [24, 25], we introduce two types of noise: 1) class-wise noise (Figure 2(b)): Noise is added separately for each class, i.e., data from different classes have distinct noise applied; 2) class-agnostic noise (Figure 2(a)): Noise is independent of class, i.e., the noise added to each class is random. Figure 3 employs causal graphs [41, 42] to illustrate the effects of different noise types on model training and testing. In Figure 3 (Clean data), without applying privacy protection to the training set, the model maintains consistent causal relationships ($D \to Y$) during both training and testing, resulting in better inference performance on the test set. In Figure 3 (Class-wise noise), when employing class-wise noise for privacy protection, the trained model makes predictions by considering both the noise N and the original data D ($N \to Y, D \to Y$). However, during the testing phase, the model can only rely on the original data D for inference ($D \to Y$). This inconsistent causal relationship between training and testing leads to poor performance for the test set (**High performance for training data while poor performance for test data**). Surprisingly, we found that even small amounts of class-wise noise can significantly impact model training. On the other hand, in Figure 3 (Class-agnostic noise), when utilizing class-agnostic noise, the established causal relationship during training is solely $D \to Y$ due to the noise being unrelated to the label. Consequently, the causal relationships between the training and testing phases remain consistent, thereby exerting minimal impact on the model's training (**High performance for both training and test data**). Thus, to achieve task-level training usability control, we should make sure that the noise

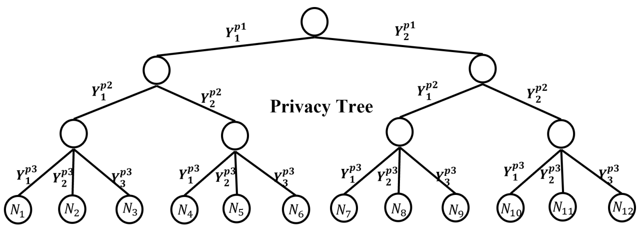


Figure 5: Example of privacy tree where we assume that there are three privacy tasks (2-class task: $Y^{p1} = \{Y_1^{p1}, Y_2^{p1}\}$, 2-class task: $Y^{p2} = \{Y_1^{p2}, Y_2^{p2}\}$, 3-class task: $Y^{p3} = \{Y_1^{p3}, Y_2^{p3}, Y_3^{p3}\}$). The edges of the tree represent different class labels.

is class-agnostic for the target task while class-wise for the privacy task.

Furthermore, it is crucial to strike a balance between the magnitude of the noise and preserving the intrinsic features of the data: 1) The noise added should be significant enough to protect visual privacy. 2) It is equally important to prevent the noise from interfering with the intrinsic features of the data (ML models should be capable of extracting the features of D from the protected data $D+N$, enabling it to learn the causal relationship $D \rightarrow Y$). This balance can be summarized as Eq.(4).

$$\min_{N_i} Dist(F_k(N_i), F_k(D_i + N_i)),\ D_i \in D_{tr},$$
$$subject\ to\ ||N_i||_2 > \delta, \tag{4}$$

N_i represents the noise applied for privacy protection on the original data D_i. $Dist(\cdot)$ denotes the distance calculation for features, while $F_k(\cdot)$ represents the features extracted by the k-th layer of a feature extractor. It can directly use publicly available pre-trained models. Given the unknown model structure and uncontrollable parameter variations during training, we compute the distances using the lower-level features of the feature extractor ($k <= 4$), as lower-level CNN features tend to exhibit better transferability [43, 44] (Figure 12).

4 System Design

4.1 Basic Pipeline

This section proposes an adversarial optimization-based noise generation mechanism to generate noise that satisfies Eq.(4). It is built upon two carefully designed loss functions, generating noise that meets the requirements of visual privacy protection without interfering with the training of the target task model. The part highlighted in the red box in Figure 4 outlines our basic pipeline, primarily consisting of two components: **Noise Initialization** and **Noise Optimization**.

Noise Initialization. Noise Initialization aims to generate initialized noise for each D_i ($\{D_i, N_i\}$) that is class-agnostic for the target task and class-wise for the privacy task (Figure 2). The storage overhead of $\{D_i, N_i\}$ becomes significant when the dataset is large. To address this issue, we introduce a privacy tree structure (T) to generate and query the initialized noise (shown in Figure 5). Each leaf node in the privacy tree corresponds to a distinct random noise ($N_j^{C,H,W} \sim \alpha \cdot Uniform(-1, 1)$), and all the data sharing the same privacy label path as N_j utilize this noise as their initialization noise. This ensures that data with the same privacy labels has the same initialization noise (class-wise), while the noise remains random in the target task dimension (class-agnostic).

The privacy tree needs to be constructed only once during the initialization phase. Moreover, we only need to store the privacy tree structure T without storing the data pairs $\{D_i, N_i\}$. Subsequently, for each data point $D_i(\mathbf{Y^t}, \mathbf{Y^p})$, we can use its privacy label $\mathbf{Y^p}$ (Y^{p1}, Y^{p2}, Y^{p3}) to query the privacy tree and retrieve its corresponding initialization noise N_i.

Noise Optimization. Noise optimization employs an adversarial approach to refine the initialized noise, ensuring its compliance with Eq.(4). We have developed two meticulously designed loss functions ($Loss_{feature}$ and $Loss_{visual}$), leveraging the adversarial approach to extract the differences between human vision and machine learning. Building upon this, we generate noises that fulfill the requirements of visual privacy protection while minimizing significant shifts in the feature domain.

$$Loss_{feature} = MSE(F_k(D_i), F_k(D_i + N_i)). \tag{5}$$

$Loss_{feature}$ is employed to ensure that a CNN can still extract useful features from protected data. It is defined as the mean squared error between the output features of the feature extractor before and after adding the noise. Minimizing $Loss_{feature}$ ensures no significant shifts in the feature domain before and after adding noise.

$$Loss_{visual} = ||N_i||_2. \tag{6}$$

$Loss_{visual}$ is employed to ensure the level of visual privacy protection. To achieve better visual privacy protection, we aim to create a discernible difference between the protected and clean images. Therefore, we quantify this difference using $Loss_{visual}$. It is necessary to ensure that $Loss_{visual}$ exceeds the threshold δ to fulfill the requirements of visual privacy protection.

The initialized noise is optimized using the gradient descent method as Eq.(7).

$$\min_{N_i}(Loss_{feature} + \gamma \cdot Relu(\delta - Loss_{visual})). \tag{7}$$

Following the two steps mentioned above, the generated noise N_i fulfills the three requirements we previously proposed: 1) The noise is class-agnostic for the target task and class-wise for the privacy task. 2) After adding noise, it does not exhibit significant changes in the feature space of the target task. 3) The magnitude of the noise is substantial enough to meet the demands of visual privacy protection.

4.2 Privacy-enhanced Plugin

To enhance privacy protection, we have designed two plug-and-play modules: Channel Selection (CS) and Untrainable Information Embedding (UIE). These modules strengthen the protection of intermediate features and unauthorized task training. Additionally, UIE module enhances the scalability of our method, allowing users to adjust privacy requirements to some extent without re-optimizing the entire dataset (as described in Section 5.6).

Channel Selection (CS). The CS mechanism aims to mitigate potential privacy leakage risks in intermediate features through channel importance screening. It is well-known that not all feature channels contribute to a specific task [44, 45], but they still pose the risk of privacy leakage. Hence, we have developed the CS mechanism to identify the crucial channels relevant to the task. Consequently, we concentrate solely on ensuring the consistency of features from the channels deemed necessary for the task while disregarding the irrelevant channels (Eq.(8)). Thus, the features extracted from those irrelevant channels are considered random, preventing potential privacy leakage of intermediate features. To achieve this, we introduce a channel selection mask, denoted as ω in Eq.(8).

$$Loss_{feature} = Dist(F_k(D_i) \otimes \omega, F_k(D_i + N_i) \otimes \omega), \quad (8)$$

ω is a binary mask with the same size as the number of feature channels, where 0 indicates an irrelevant channel for the task, and 1 represents a relevant channel for the task. We propose **CS Mask Initialization** to generate ω based on the configured target task and privacy task. Firstly, The selected features need to satisfy the following two requirements:

Minimize the disclosure of private information. 1) The selected features should contain minimal information (Eq.(9)). 2) The selected features should exhibit poor clustering performance in the privacy task label dimension. Therefore, accurately classifying the privacy-related labels of the data is not sufficient (Eq.(10), $CH(X,Y)$ is Calinski-Harabasz Index [46]. A larger value of CH index means the better the clustering effect, implying that it is easier to infer privacy attributes).

Fulfill the objectives of the target task. The selected features should exhibit good clustering performance in the target task label dimension, enabling successful classification of the target task (Eq.(11)).

$$Loss_f = \sum(||F_k(D_i)||_2 \otimes Relu(\omega - \varepsilon)). \quad (9)$$

$$Loss_p = CH(F_k(N_i) \otimes Relu(\omega - \varepsilon), Y^p). \quad (10)$$

$$Loss_t = CH(F_k(N_i) \otimes Relu(\omega - \varepsilon), Y^t). \quad (11)$$

Finally, we get the channel selection mask ω by solving the following joint optimization problem Eq.(12). To ensure that the channel selection mask is in binary form (0 or 1), we apply the *ReLU* function to threshold the values. By setting threshold ε close to 1 (e.g., 0.93 in our experiment), the $ReLU(\omega - \varepsilon)$ operation effectively approximates the binary selection form, where the values are either close to 1 or 0. To ensure that ω remains within the range of [0, 1], we introduce $\omega = \frac{1}{1+exp(-\omega')}$ and perform gradient descent on ω' during optimization. Based on the given task configuration, the CS module only needs to be initialized once.

$$\min_{\omega} \alpha \cdot Loss_f + Loss_p - Loss_t. \quad (12)$$

Untrainable Information Embedding (UIE). The noise optimization will change the correlation between noise and classes, reducing the suppressive effect on the training of the privacy task model. To mitigate this issue, we propose the UIE module. It guides the privacy task model to learn noise information by embedding additional class-wise noise, further preventing unauthorized model training.

This module is built upon a crucial discovery: Even embedding a small proportion of class-specific noise will significantly reduce the training performance of the model (details can be found in Appendix A) while not causing significant alterations in the feature domain. Eq.(13) illustrates the core steps of the UIE module.

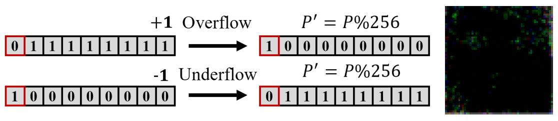
$$N_i^{UIE} = N_i' + N_i \otimes m, \quad (13)$$

N_i' is the output of noise optimization, and N_i is the initialization noise queried from the privacy tree. N_i^{UIE} is the noise enhanced by the UIE module. By intentionally adding class-wise noise for the privacy task, the model learns more about the relationship between the noise and the labels during training, resulting in an unusable privacy task model for attackers. m, defined by Eq.(14) denotes the UIE mask.

$$m^{H,W} \sim \rho \cdot Bernoulli(r), \quad (14)$$

ρ and r represent the magnitude and ratio of UIE embedding. They control the trade-off between usability and privacy (higher values leading to a higher level of privacy protection), which is initialized once the task is determined.

As stated in section 3.7, a stronger correlation between the noise embedded by UIE ($N_i \otimes m$) and the privacy task labels results in more effective privacy protection. However, we have observed that increasing the depth of the privacy tree (resulting in a higher number of initial noises (n) within each privacy class label) leads to significant fluctuations in the correlation between the noise and privacy labels after random sampling using Eq.(14). These fluctuations can have adverse effects on privacy protection.



(a) Overflow and Underflow (b) M_o

Figure 6: Examples of Overflow and Overflow Matrix M_o

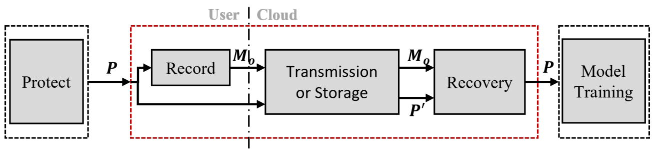


Figure 7: The Pipeline of Overflow Matrix M_o for Image Transmission & Storage.

To address this issue, we propose the **UIE Mask Initialization**. Specifically, we advocate for performing multiple random samplings and calculating the CH index of $N_i \otimes m$ to mitigate the negative impact of these fluctuations on privacy protection. An increase in the CH index corresponds to improving the clustering performance, signifying a stronger correlation between the noise and the label. By selecting m with the highest CH index, we can mitigate the adverse effects of sampling fluctuations on privacy protection. This approach can improve the stability and consistency of privacy protection (As shown in Figure 16(b)).

4.3 Overflow Matrix for Real-world Image Transmission & Storage

This section ensures that the protected image data can be compatible with current image encoding and transmission frameworks. In real-world applications, image data is typically transmitted and stored using the 8-bit unsigned integer (uint8) data type, where pixel values range from 0 to 255 [47]. However, our approach generates a high magnitude of noise to enhance privacy protection, leading to protected image pixel values exceeding the 0-255 range. When converting these values to the uint8 data type, overflow occurs, significantly compromising the usability of the protected data.

Overflow Details. Figure 6(a) illustrates the overflow details when storing pixel values using uint8 data type. Overflow occurs when the pixel value exceeds 255, while underflow arises when the pixel value falls below 0. Once an overflow occurs, the actual stored pixel value becomes $P'(P' = P\%256)$. Knowledge of $\lfloor (P/256) \rfloor$ is required to recover the original pixel value.

Therefore, we propose *overflow matrix* M_o, which serves two purposes: 1) It records the overflow status of each pixel during image transmission/storage, and 2) It assists in recovering the protected tensor data before training. Figure 6(b) shows an example of M_o.

Figure 7 shows the pipeline using M_o. The protected tensor data P is converted to overflow matrix M_o and encoded uint8 image P' during transmission or storage. When performing model training in the cloud, the data used for training (P) is recovered through M_o and P'. During image transmission or storage, M_o tracks how pixels have exceeded the uint8 range. This information is transmitted or stored alongside the encoded image P', enabling us to identify and handle the overflowed pixels during subsequent processing (Eq.(15)). Before training, M_o is employed to restore the protected training data P. By utilizing M_o, we can recover the pixel values affected by the overflow, guaranteeing usability for training purposes(Eq.(16)).

$$record \rightarrow M_o = \begin{cases} \lfloor P/256 \rfloor, & P > 0, \\ \lfloor P/256 \rfloor - 1, & P < 0. \end{cases} \tag{15}$$

$$recovery \rightarrow P = P' + 256 * M_o. \tag{16}$$

To integrate M_o into image encoding and reduce transmission costs, we also utilize the uint8 data type to represent M_o. However, unlike the previous approach, in this case, the lower four bits are assigned to represent positive values, while the upper four bits represent negative values ($-128 < M_o < 128$). Consequently, when M_o is less than 0, the actual stored value is $M_o + 128$. This encoding scheme ensures the accurate representation of the negative part of M_o.

5 Evaluation

Our objective is to train models using protected data on the cloud without modifications to the existing framework, which would provide significant convenience for cloud model trainers. Therefore, in our experiments, we trained the task model using the same hyperparameters (learning rate, optimizer, etc.) on both unprotected (clean) and protected training data. Subsequently, we evaluated the model's performance using clean test data.

5.1 Experiment Setup

Dataset. To validate the effectiveness of our method across multiple tasks, we selected a diverse array of datasets and tasks encompassing varying scales, ranging from attribute-level to object-level, behavior-level, and scene-level. Table 2 presents the specific configurations of the datasets and tasks.

- RAF-DB (Real-world Affective Faces) [48] is a large-scale facial expression database with 29672 great-diverse facial images. Each image has 7 classes of basic emotion attributes, 3 classes of race attributes, and 3 classes of gender attributes.
- CelebA (Large-scale CelebFaces Attributes) [49] is a celebrity face dataset comprising 202,599 facial images belonging to 10,177 celebrity identities. Each image in the dataset is annotated with 40 binary attributes.

Table 2: Dataset statistics and task configure.

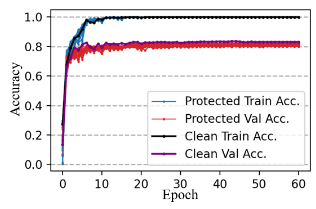
Dataset	Scale	#of Images	Target Task	Privacy Task
RAF-DB	Attribute	29672	7-class Emotion	5-class Age 3-class Race 3-class Gender
CelebA	Object	202599	10177-class Identity	2-class Young 2-class Smile
			2-class Smile	10177-class Identity
PubFig	Object	5879	53-class Identity	-*
SFDDD	Behavior	102152	10-class Behavior	-*
MIT Indoor	Scene	6699	64-class Scene	-*

*: The dataset contains only one annotation type and cannot be used to train privacy tasks.

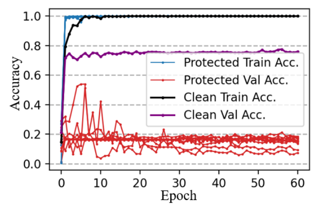
- PubFig (Public Figures Face) [50] is a collection of facial images of public figures, consisting of 58,797 face images from 200 individuals. However, we discovered a significant amount of noisy data after downloading the dataset. Therefore, we manually selected and cleaned the data for 53 individuals, ensuring its quality and reliability.
- SFDDD (State Farm Distracted Driver Detection) [51] is a dataset for in-vehicle driver behavior detection, comprising 10 different driving behaviors.
- MIT indoor (MIT Indoor Scene Recognition) [52] is a dataset commonly used for scene classification, which is a large-scale task. It consists of images of 64 indoor scenes.

Evaluation metrics.

- *PSNR (Peak Signal-to-Noise Ratio [0,100]) and SSIM (Structural Similarity [0,1])*, used for measuring the effectiveness of visual privacy protection, where a smaller value indicates a stronger privacy effect. SSIM<0.5 PSNR<20 are generally considered effective for strong visual protection
- *LPIPS (Learned Perceptual Image Patch Similarity [53])*, aims to capture the perceptual similarity between images, aligning more closely with human perception. A lower LPIPS value indicates a higher similarity between the two images. LPIPS>0.2 means better visual privacy protection.
- *Accuracy,* is the ratio of correctly classified samples to the total samples in the clean test set. It is used to measure usability, where a smaller difference between the accuracy before and after privacy protection indicates better usability (Target Task Models).
- *PSR (Protection Success Rate %),* defined by Eq.(17). $Accuracy_c$ and $Accuracy_p$ represent the model training accuracy before and after privacy protection. n_class means the class number of tasks. Thus, when accuracy is less than or equal to $1/n\_class$, it signifies a random model, and its corresponding PSR is 100%.



(a) Target Task (Emotion)



(b) Privacy Task (Age)

Figure 8: Training and Validation Curve. Due to the randomness of UIE, we performed 10 random UIE embeddings. Thus, the Protected Train/Val Acc. consists of 10 curves.

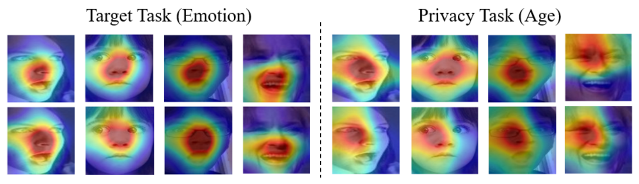


Figure 9: Visualization of Class Activation Maps [54] (CAMs) for Target Task and Privacy Task Models. The first line uses the model trained on clean data ($f_{D_{tr}\to Y}$), and the second uses the model trained on protected data ($f_{D'_{tr}\to Y}$). For the target task, $f_{D_{tr}\to Y}$ and $f_{D'_{tr}\to Y}$ focus on (orange-red areas) the same regions. In contrast, for the privacy task, they focus on significantly different regions.

$$PSR = 100 * \frac{(Accuracy_c - Accuracy_p)}{Accuracy_c - 1/n\_class}. \tag{17}$$

Hyperparameter. Hyperparameters have a significant impact on convergence, performance, and generalization. γ and δ control the trade-off between usability and visual privacy, with values $\gamma = 0.005$ and $\delta > 10000$. k determines the transferability (cross-model training performance, a smaller k means a better transferability) and $k = 4$ in our experiment. α ($\alpha = 0.3$) controls the magnitude of the initial noise and determines the convergence speed. Additionally, $\rho = 1.0$ and $r = 0.0008$ for UIE module.

Baseline. As we have emphasized, we propose a unique privacy-utility goal, especially for cloud-based model training. We did not find any related work completely consistent with our goal. Therefore, we compared with works with similar or partially aligned objectives. Specifically, this includes Cloud-based Model Inference (PPFR-FD [33]), Unlearnable Examples (Fawkes [23]), Augmentation (Mixup [19]), Dataset Distillation (DataDAM [27]) and Differential Privacy (DataDAM+DP-SGD [15]). Detailed experimental setup and discussion can be found in Appendix B.

Unless otherwise specified, the model architecture used for training in our experiments (both target task and privacy task) is ResNet34, and the feature extractor is ResNet18 ($k = 4$). All models are initialized using pre-trained parameters provided by torchvision. The dataset we use is RAF-DB, with the target task being emotion classification and the privacy task being

Table 3: Comparison of training usability for different tasks. For the target task, we aim to ensure similar performance between the models trained on protected data and those trained on clean data. For the privacy task, we desire a higher PSR for better privacy protection.

Dataset	**Target Task**			**Privacy Task**			
	Task Name	**Accuracy$^{\text{clean}}$**	**Accuracy$^{\text{protected}}$**	**Task Name**	**Accuracy$^{\text{clean}}$**	**Accuracy$^{\text{protected}}$**[1]	**PSR**[2]
RAF-DB	7-class Emotion	82.32%	81.42%	5-class Age	78.75%	17.57%	100%
				3-class Race	84.57%	18.73 %	100%
				3-class Gender	85.62%	11.35%	100%
CelebA	10177-class Identity	62.02%	57.26%	2-class Age	99.14%	31.5%	100%
				2-class Smile	99.28%	35.6%	100%
	2-class Smile	99.28%	99.14%	10177-class Identity	62.02%	0.00%	100%
PubFig	53-class Identity	80.60%	78.45%	-	-	-	-
SFDDD	10-class Behavior	96.54%	95.81%	-	-	-	-
MIT indoor	64-class Scene	70.53%	68.13%	-	-	-	-

1. Due to the challenges in achieving convergence and the noticeable fluctuations in testing accuracy when training a privacy task model using protected data, we view the average accuracy value obtained from the last few epochs as **Accuracy$^{\text{protected}}$**.
2. The maximum value for the PSR is 100%. Thus, if the result calculated by Eq.(17) exceeds 100%, we still consider it as achieving a 100% protection effect.

Table 4: Quantification of visual privacy protection.

Scale	Protected tensor (P)				Encoded Image (P')			
	MSE $\uparrow$	PSNR [0-100] $\downarrow$	SSIM [0-1]$\downarrow$	LPIPS $\uparrow$	MSE $\uparrow$	PSNR [0-100] $\downarrow$	SSIM [0-1]$\downarrow$	LPIPS $\uparrow$
Atrribute Level	9026	19.16	0.043	0.657	6542	8.67	0.018	1.16
Object Level	18254	23.22	0.046	0.611	8344	7.85	0.012	0.82
Behavior Level	12547	18.5	0.095	0.69	11507	8.75	0.048	0.87
Scene Level	10374	21.15	0.12	0.53	15244	9.09	0.037	0.78

age classification.

5.2 Training Usability for Different Tasks

Table 3 presents the evaluation of training usability of privacy-protected data on different tasks using various scenario datasets. For the target tasks, privacy protection has minimal impact on their training usability, with the accuracy difference between models trained on clean and protected data being within 2%. Conversely, the protected data is unsuitable for training privacy task models, with the PSR exceeding 100% In addition to facial datasets, we also utilized larger-scale datasets (A driver behaviors classification dataset: SFDDD dataset, and an indoor scene classification dataset: MIT indoor dataset) to validate the effectiveness of our method. The experimental results indicated that privacy protection had a minimal impact on training the target task for these datasets, with an accuracy degradation of less than 1.5%.

Figure 8 illustrates the training and validation accuracy curve for both the target and privacy tasks. When training privacy task models with protected data, the strong correlation between class labels and noise (class-wise noise) causes models to overfit noise in the training data (training accuracy quickly reaches 100%, but testing accuracy is inferior). However, this issue doesn't appear when training target task models, as the model cannot learn $N \rightarrow Y$ from class-agnostic noise. Further-

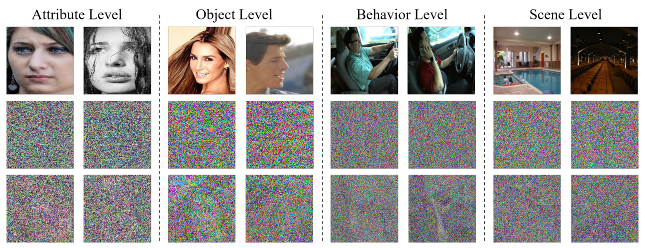
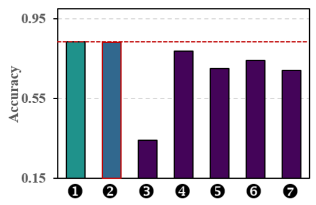


Figure 10: Visualization of Visual privacy protection. The first line is clean data D_i, the second line is noise N_i applied for privacy protection, and the third line is the protected data $D_i + N_i$.

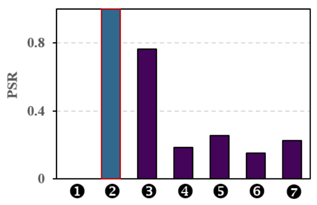
more, we visualize regions that the trained model focuses on by CAMs (Figure 9). For the target task, the CAMs between $f_{D_{tr} \rightarrow Y}$ and $f_{D'_{tr} \rightarrow Y}$ are largely consistent, indicating similar model performance. In contrast, it shows significant shifts in the privacy task, suggesting that the model has learned incorrect information.

5.3 Visual Privacy Protect

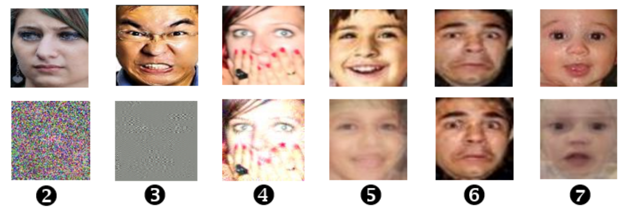
Figure 10 shows the visual effects of privacy protection. The noise applied for privacy protection is significant enough to conceal the visual features of the original image, achieving effective visual privacy protection. Table 4 quantifies the visual protection effectiveness from the perspectives of image quality and perceived similarity. The results demonstrate after protection (the protected tensor P), the SSIM is 0.076, the PSNR is 20.5, and the LPIPS is 0.622. When P is encoded to image P' for T&S, better visual privacy protection can be achieved: The SSIM decreased to 0.028, PSNR decreased to 8.59, and LPIPS increased to 0.91.



(a) Task Accuracy for Target Task



(b) PSR for Privacy Task



(c) Visaul Privacy. We randomly select one clean data (first line) and its corresponding protected data (second line) for each method. Since the protected data from methods ❺, ❻, and ❼ do not explicitly correspond with the clean data, we select the most similar protected data sample from the protected dataset.

Figure 11: Experimental Comparison with Exiting Works. ❶: Clean, ❷:Ours, ❸: PPFR-FD, ❹: Fawkes, ❺: Mixup, ❻: DataDAM, ❼: DataDAM+DP.

5.4 Comparison With Existing Works

We use the RAF-DB dataset, setting emotion classification as the target task and age classification as the privacy task. Implementation details of each method can be found in the Appendix B. As shown in Figure 11, our method achieves the optimal privacy-utility trade-off, with about a 1% drop in target task accuracy and 100% PSR for the privacy task. PPFR-FD has very poor training usability with clean test data (drops by 50%) for the target task and is not suitable for model training scenarios. Similarly, Mixup also suffers from a significant drop in target task accuracy (drops by 13+%) if we mix enough images to achieve privacy protection. Fawkes achieves minimal impact (except for ours) on the training of the target task (drops by about 5%). This is because the protected data it generates is very similar to the original data, significantly compromising the user's visual privacy (As shown in Figure.11(c)). Additionally, the PSR for the privacy task is very low in our setting. This might be due to the limited number of classes in the privacy task, resulting in insufficient confusion in the feature spaces. Consequently, Fawkes is only limited to tasks with more classes. Even though Data Distillation (DataDAM) can achieve some level of training usability for the target task, there remains a significant gap in model accuracy compared to training with the original dataset (drops by 9.8%). Moreover, it does not effectively protect privacy: 1) It treats all training tasks equally, which fails to prevent unauthorized privacy task training (PSR < 14%); 2) As the size of the synthetic dataset increases, the synthetic images become very similar to the original images (As shown in Figure.11(c)). Traditional differential privacy for ML model training aims to train models that satisfy differential privacy, ensuring that the privacy of individual training samples is not compromised. This approach does not align with our goal (generating an offline privacy-preserving dataset). Instead, we combine DP-SGD with data distillation to generate a protected dataset that satisfies differential privacy [20]. Clearly, the model training performance for the target task is constrained by data distillation (drops by 12%+). [21] also demonstrates that a simple combination of data distillation and differential privacy does not necessarily guarantee differential privacy. Importantly, our privacy-preserving objective is not to ensure differential privacy but to achieve a more practical privacy goal, especially for cloud model training.

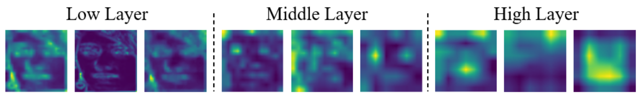


Figure 12: Feature visualization for different feature extractors by [55]. Each dashed-line-separated block contains three different feature extractors: ResNet18, VGG13, and DenseNet121.

Table 5: Transferability across different feature extractors and target models.

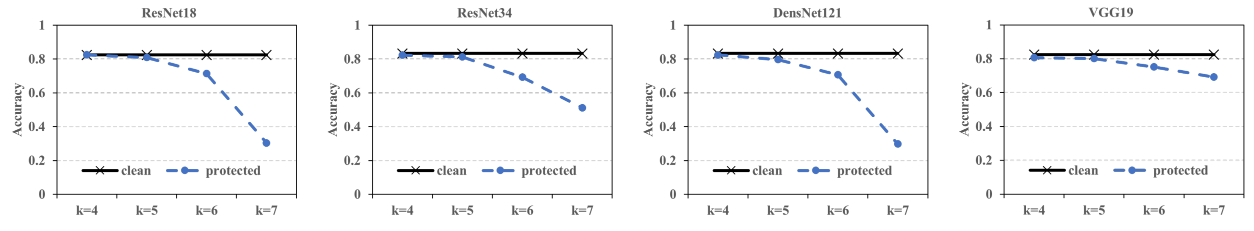
Feature Extractor	Target Task Model				
	ResNet18	ResNet34	VGG13	VGG19	DenseNet121
None*	82.91%	82.32%	82.04%	82.46%	81.31%
ResNet18[0:4]	83.08%	83.38%	79.68%	80.70%	82.40%
VGG13[0:5]	82.59%	81.93%	82.27%	83.31%	79.34%
DenseNet121[0:4]	80.64%	81.29%	79.92%	80.96%	80.57%

1. *: Training target task model with clean dataset.

5.5 Cross-Model Performance

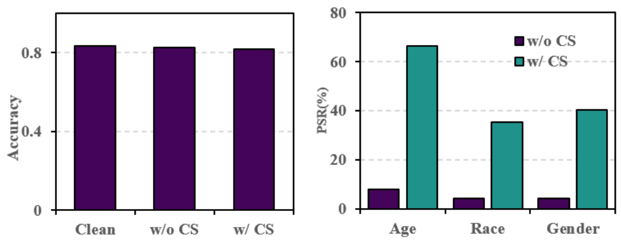
We aim to generate a protected dataset that is task-specific rather than model-specific. Therefore, the features utilized during the protection phase must exhibit strong transferability to adapt to different model structures. Figure 12 illustrates the visualization of features from different layers across various models. Despite the differences in model architectures, the features in the lower layers exhibit a high similarity, which means better transferability. Figure 13 further investigates the impact of different layers (k) for feature extractor $F_k(\cdot)$ on the training of the target task model. The results indicate that training high-accuracy models is achievable by setting k to 4 or 5, regardless of the structure of the target task model (privacy protection has minimal impact on the training process for the target task). However, a significant decline in model accuracy is observed when k is set to 6 or 7. Therefore, selecting a smaller value for k in the feature extractor is advised to enhance transferability and training usability.

Table 5 summarizes the training usability across different



(a) ResNet18 model (b) ResNet34 model (c) DenseNet121 model (d) VGG19 model

Figure 13: Target model performance with different k values for feature extractor (ResNet18).Lower k value ($k = 4$ or 5) can achieve training a high performance ML model.



(a) Target Task Accuracy (b) PSR of Privacy Attribute

Figure 14: Effectiveness of the CS Mechanism. We evaluate it from two aspects: usability (target task accuracy 14(a)) and privacy (PSR of privacy attribute inference 17(c)).

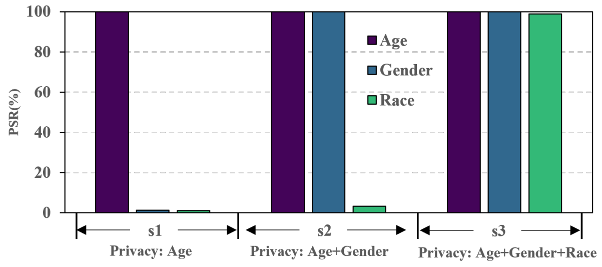


Figure 15: Example of the scalability for dynamic privacy requirements. In phase s1, the privacy task is age classification. In phase s2, gender classification is added, and in phase s3, race classification is further added. We only need to modify the UIE module to meet the dynamically changing privacy requirements.

feature extractors and target models. The protected dataset can still achieve excellent training usability (average accuracy drop of about 0.73%) despite the varying architectures between the feature extractor and the task model. Even in certain cases (e.g., ResNet18[0:4]–ResNet34), models trained with protected data achieve higher accuracy than those trained with clean data.

5.6 Effectiveness of CS Module

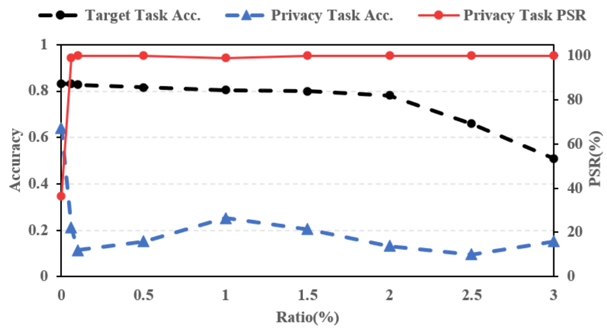
Figure 14 illustrates the impact of the CS mechanism on the usability of the target task and the inference of privacy attributes. It can be observed that the CS mechanism can effectively prevent the inference of privacy attributes (with an average PSR improvement of 41.9%) without compromising the training accuracy of the target task (within 0.9%).

5.7 Effectiveness of UIE Module

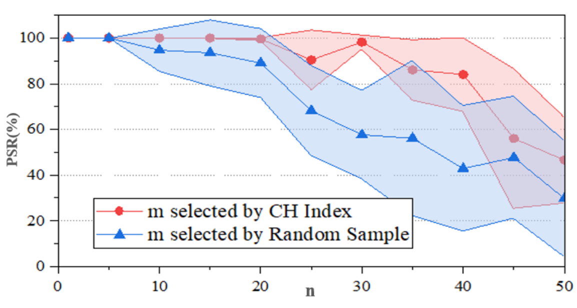
Example of high scalability for dynamic privacy. We provided a simple example to illustrate the high scalability advantage of our plug-and-play UIE module. When the user's privacy requirements ($\mathbf{Y^p}$) change, we can modify the UIE module only. Based on the privacy tree corresponding to the new privacy requirements, we can generate UIE noise that satisfies the privacy demands and directly embeds it into the optimized noise (N_i') generated previously. There is no need to re-optimize and generate new noise. As shown in Figure 15, in the first phase, our privacy task only involves age classification. We optimize and generate noise N_i' based on the task configuration and then apply UIE to generate N_{i-s1}^{UIE}. In the second and third phases, we introduce additional privacy task requirements for gender and race classification. We directly use the N_i' generated in the first phase and efficiently implement privacy protection for the new privacy task by simply modifying the embedding noise ($N_{i-s2} \otimes m, N_{i-s3} \otimes m$) of the UIE module.

Figure 16(a) illustrates the impact of UIE embedding ratios (r) on both the target and privacy tasks. When r exceeds 0.1%, the PSR for the training of the privacy task can be significantly increased by up to 100% without affecting the training of the target task. When r ranges from 0.1% to 2%, a good trade-off between privacy protection and task usability can be achieved. The influence on the target task model training accuracy remains within 1% while achieving a 100% PSR.

As the depth of the privacy tree increases, the number of different initial noises within each privacy label will increase. Consequently, the correlation between noise and privacy labels may weaken, potentially leading to adverse effects on privacy protection. To investigate this, we studied the impact of the number of different initial noises (represented as n) within each privacy label on the effectiveness of protection. As shown in Figure 16(b), employing the CH index for selecting the UIE mask during initialization can significantly enhance the protection effectiveness compared to random se-



(a) Impact of UIE embedding ratios (r) on the training usability.



(b) PSR decreases when the amount of noise increases.

Figure 16: Effectiveness of the UIE Mechanism.

Table 6: Time overhead and memory usage.

Image Size	GPU Time Overhead(s)	GPU Memory Usage(MB)	CPU-only Time Overhead(s)
100×100×3	0.11	1312	0.21
224×224×3	0.15	1332	0.99

lection. However, it is worth noting that as n increases, the PSR tends to decrease and exhibit significant fluctuations, which can have negative implications for privacy protection. Therefore, we recommend setting a limit of no more than 20 different initial noises within each privacy label to mitigate these issues and ensure a satisfactory level of protection.

5.8 Efficiency

We implemented our mechanism in two different computing environments: GPU(RTX 3060 12GB) and CPU-only(11th i5-11500 @ 2.70GHz), which we consider representative of the devices available on the user's end. We primarily evaluated the efficiency from the perspectives of time overhead (per data) and GPU memory usage. The results in Table 6 indicate that our method can perform well on resource-constrained user devices (The primary restriction for training ML models on user devices is GPU memory, which leads to out-of-memory errors for complex models. However, our method can be completed with a fixed and relatively small GPU usage). Below is the analysis of the time and space overhead compared to completely training a model:

The time overhead and space overhead of training a neural network model can be approximately represented as $O(C \cdot T \cdot N)$ and $O(C)$ respectively, where C is the complexity of the model to be trained, N is the scale of the training data, and T is the number of training epochs. Similarly, the time and space complexity of our method can be approximately represented as $O(c \cdot t \cdot N)$ and $O(c)$ where c is the complexity of the feature extractor, t is the number of iterations. Firstly, for complex models, $C >> c$. Moreover, T will significantly increase with the growth of C and N, whereas t remains a fixed value (generally $T > t$). Therefore, the time and space overhead of our method are much lower than those of complete model training, and this advantage becomes more pronounced as the complexity of the target model increases. More importantly, by decoupling training and privacy protection, we only need to perform privacy protection once on user devices (for given tasks). Many repeated operations, such as parameter tuning or model architecture selection, can be implemented on the cloud based on this protected dataset. Even if the privacy task changes, we can modify only the UIE module without needing to re-optimize the entire dataset (Figure 15).

6 Real World Performance

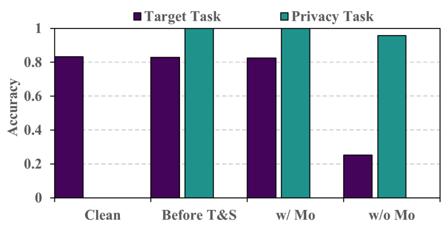
6.1 Model training on cloud server

We utilize rented public cloud servers for model training, a commonly used cloud-based model training paradigm. In this setup, users provide the dataset and training code/model and leverage only the computational power of the cloud to train the model. The data accessible in the cloud consists of T&S images (P') and its overflow matrix (M_o). During model training, the protected data P is recovered based on the overflow mechanism for training. We evaluated the effectiveness of such scenarios from three aspects:

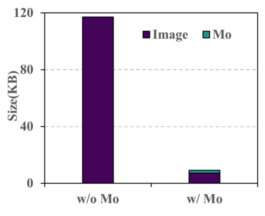
Train usability and PSR. The results in Figure 17(a) indicate that image encoding for transmission/storage has minimal impact on the training for the target task (The accuracy difference between cloud-based and on-premises model training is within 0.3%) with overflow matrix mechanism. Compared to training models without the overflow matrix mechanism, the accuracy improvement is 57%+. For privacy tasks, it can also achieve a 100% PSR after image encoding, transmission to cloud.

T&S Cost. Figure 17(b) illustrates the costs of data transmission or storage. The results show that, compared to directly transmitting tensor data (P) without M_o, the spatial overhead per image is reduced by over 100kb (data size is 3*100*100) when using the overflow matrix mechanism.

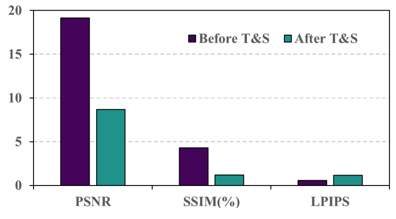
Visual Privacy. Figure 17(c) presents the comparison of the visual privacy protection effects before (P) and after (P')



(a) Train usability and PSR



(b) T & S Cost



(c) Visual Privacy Protection

Figure 17: Effectiveness of the Overflow Matrix (M_o) Mechanism. We evaluate it from three aspects: train usability and PSR (17(a)), transmission and storage costs (17(b)), and visual privacy protection (17(c)).

Table 7: Target task model trained and tested by EasyDL.

	Top-1 Accuracy	Top-2 Accuracy	Accuracy
Clean Data*	90.7%	95.4%	77.0%
Protected Data†	89.2%	94.7%	74.3%
	Precision	**Recall**	**F1-Score**
Clean Data*	71.2%	63.4%	65.8%
Protected Data†	69.4%	62.5%	64.1%

1. *: The model are trained and tested by clean data.
2. †: The model are trained by protected data and tested by clean data

transmission or storage (As shown in Figure 7, P represents the tensor data that is directly optimized for protection. P' refers to the image encoded used for transmission or storage). The experimental results demonstrate that P' has stronger visual privacy protection effects than P.

6.2 Cloud-based model training by Baidu EasyDL API

The Baidu EasyDL API [2] is another more convenient cloud-based model training paradigm. Users only need to upload their training data, and the cloud platform automatically selects the model and hyperparameters for training. Ultimately, the API provides the user with a deployable task model.

We use EasyDL to train the emotion classification task with the protected RAF-DB dataset. We summarize the performance results of the target task model provided by EasyDL in Table 7. The model trained on the protected data demonstrated comparable performance to the model trained on the clean data, with a negligible accuracy difference of approximately 2.7% and an F1-score difference of around 1.7%. This outcome highlights the effective retention of training usability for the target task while leveraging the protected data.

7 Discussion and Limitation

Ethics Statement. This work uses only public datasets and focuses on privacy protection in machine learning.

Noise magnitude constraint. Some black-box cloud-based model training frameworks, like EasyDL, do not support custom data processing, thus preventing the use of overflow matrix. The protected pixel values must be limited to [0, 255] to ensure that images can be transmitted using existing encoding frameworks. We recommend embedding a clipping mechanism during the noise optimization, where after a certain number of iterations. This approach will limit the magnitude of noise N_i, which may compromise visual privacy protection.

Other Privacy Attacks. *1) Attribute inference.* The training data includes attribute labels, so attribute inference attacks are not our primary focus. Although the CS mechanism can provide some degree of protection against attribute inference, it cannot guarantee a 100% PSR. Further protection measures are necessary if the privacy goal is to prevent attribute inference. One feasible solution is the adversarial perturbation approach, disrupting the attribute inference attack model's accuracy by applying subtle perturbations to the protected image. *2) Membership inference.* Member inference attacks, aiming to infer the training data privacy from well-trained models, are orthogonal to our work [56]. Our method does not involve the model training process to get well-trained models. Interestingly, training models with protected data can partially mitigate membership inference attacks (Appendix D). This is because the model overfits the protected training data, which differs from the original one. Besides, strategies to mitigate membership inference during model training are equally applicable.

8 Conclusion

In this paper, we introduce a unique privacy-utility goal for cloud-based model training and propose a practical mechanism to achieve this goal. With only one-time protection (for given tasks), we can enable computationally intensive and repetitive training processes to be performed entirely on the cloud without changing the existing training pipeline. Extensive experiments conducted in real-world applications demonstrate that the protected data not only fulfills the privacy requirements (including visual privacy and prevention of unauthorized model training) but also allows for training the target task with minimal performance degradation. In the future, we will extend our method to other types of data, such as audio, text, and more.

9 Acknowledgments

We thank the anonymous reviewers for their valuable and constructive feedback. This research is partially supported by National Key R&D Program of China under Grant No.2021ZD0110400, Innovation Program for Quantum Science and Technology 2021ZD0302900 and China National Natural Science Foundation with No. 62132018, 62231015, "Pioneer" and "Leading Goose" R&D Program of Zhejiang, 2023C01029 and 2023C01143, and NIO Inc.

References

[1] Azure. Azure databricks. `https://azure.microsoft.com/en-us/products/databricks/`.

[2] Baidu. Baidu easydl api. `https://console.bce.baidu.com/easydl/imgcls/overview`.

[3] Xincheng Li, Zhaoyi Wang, Yanjun Huang, and Hong Chen. A survey on self-evolving autonomous driving: a perspective on data closed-loop technology. *IEEE Transactions on Intelligent Vehicles*, 2023.

[4] Kashmir Hill. The secretive company that might end privacy as we know it. In *Ethics of Data and Analytics*, pages 170–177. Auerbach Publications, 2022.

[5] Maya Shwayder. Clearview ai's facial-recognition app is a nightmare for stalking victims. *Digital Trends*, 1, 2020.

[6] Angelica Mari. Brazilian retailer quizzed over facial recognition tech. *ZDNet, March*, 2019.

[7] Hao Wu, Xuejin Tian, Minghao Li, Yunxin Liu, Ganesh Ananthanarayanan, Fengyuan Xu, and Sheng Zhong. Pecam: privacy-enhanced video streaming and analytics via securely-reversible transformation. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*, pages 229–241, 2021.

[8] Shulan Wang, Qin Liu, Yang Xu, Hongbo Jiang, Jie Wu, Tian Wang, Tao Peng, and Guojun Wang. Protecting inference privacy with accuracy improvement in mobile-cloud deep learning. *IEEE Transactions on Mobile Computing*, 2023.

[9] Abhishek Singh, Ayush Chopra, Ethan Garza, Emily Zhang, Praneeth Vepakomma, Vivek Sharma, and Ramesh Raskar. Disco: Dynamic and invariant sensitive channel obfuscation for deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12125–12135, 2021.

[10] Yuxi Mi, Yuge Huang, Jiazhen Ji, Hongquan Liu, Xingkun Xu, Shouhong Ding, and Shuigeng Zhou. Duetface: Collaborative privacy-preserving face recognition via channel splitting in the frequency domain. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 6755–6764, 2022.

[11] Runhua Xu, Nathalie Baracaldo, and James Joshi. Privacy-preserving machine learning: Methods, challenges and directions. *arXiv preprint arXiv:2108.04417*, 2021.

[12] Jingjing Yang, Jiaxing Liu, Runkai Han, and Jinzhao Wu. Transferable face image privacy protection based on federated learning and ensemble models. *Complex & Intelligent Systems*, 7(5):2299–2315, 2021.

[13] Dong Chen, Siliang Tang, Zijin Shen, Guoming Wang, Jun Xiao, Yueting Zhuang, and Carl Yang. Fedaa: Using non-sensitive modalities to improve federated learning while preserving image privacy. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 3796–3806, 2023.

[14] Fahad Ahmed KhoKhar, Jamal Hussain Shah, Muhammad Attique Khan, Muhammad Sharif, Usman Tariq, and Seifedine Kadry. A review on federated learning towards image processing. *Computers and Electrical Engineering*, 99:107818, 2022.

[15] Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS'16. ACM, October 2016.

[16] Natalia Ponomareva, Hussein Hazimeh, Alex Kurakin, Zheng Xu, Carson Denison, H. Brendan McMahan, Sergei Vassilvitskii, Steve Chien, and Abhradeep Guha Thakurta. How to dp-fy ml: A practical guide to machine learning with differential privacy. *Journal of Artificial Intelligence Research*, 77:1113–1201, July 2023.

[17] Seyedeh Maryam Hosseini, Milad Sikaroudi, Morteza Babaei, and Hamid R Tizhoosh. Cluster based secure multi-party computation in federated learning for histopathology images. In *International Workshop on Distributed, Collaborative, and Federated Learning*, pages 110–118. Springer, 2022.

[18] Oleksandr Lytvyn and Giang Nguyen. Secure multi-party computation for magnetic resonance imaging classification. *Procedia Computer Science*, 220:24–31, 2023.

[19] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[20] Dingfan Chen, Raouf Kerkouche, and Mario Fritz. Private set generation with discriminative information, 2022.

[21] Nicholas Carlini, Vitaly Feldman, and Milad Nasr. No free lunch in "privacy for free: How does dataset condensation help privacy", 2022.

[22] Hanxun Huang, Xingjun Ma, Sarah Monazam Erfani, James Bailey, and Yisen Wang. Unlearnable examples: Making personal data unexploitable. *arXiv preprint arXiv:2101.04898*, 2021.

[23] Shawn Shan, Emily Wenger, Jiayun Zhang, Huiying Li, Haitao Zheng, and Ben Y Zhao. Fawkes: Protecting privacy against unauthorized deep learning models. In *29th USENIX security symposium (USENIX Security 20)*, pages 1589–1604, 2020.

[24] Qi Tian, Kun Kuang, Kelu Jiang, Furui Liu, Zhihua Wang, and Fei Wu. Confoundergan: Protecting image data privacy with causal confounder. *Advances in Neural Information Processing Systems*, 35:32789–32800, 2022.

[25] Jiaming Zhang, Xingjun Ma, Qi Yi, Jitao Sang, Yu-Gang Jiang, Yaowei Wang, and Changsheng Xu. Unlearnable clusters: Towards label-agnostic unlearnable examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3984–3993, 2023.

[26] Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A. Efros. Dataset distillation, 2018.

[27] Ahmad Sajedi, Samir Khaki, Ehsan Amjadian, Lucy Z Liu, Yuri A Lawryshyn, and Konstantinos N Plataniotis. Datadam: Efficient dataset distillation with attention matching. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17097–17107, 2023.

[28] George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A. Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2022.

[29] Bo Zhao and Hakan Bilen. Dataset condensation with differentiable siamese augmentation. In *International Conference on Machine Learning*, pages 12674–12685. PMLR, 2021.

[30] Zhibo Wang, He Wang, Shuaifan Jin, Wenwen Zhang, Jiahui Hu, Yan Wang, Peng Sun, Wei Yuan, Kaixin Liu, and Kui Ren. Privacy-preserving adversarial facial features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8212–8221, 2023.

[31] Vahid Mirjalili, Sebastian Raschka, and Arun Ross. Privacynet: Semi-adversarial networks for multi-attribute face privacy. *IEEE Transactions on Image Processing*, 29:9400–9412, 2020.

[32] Haohan Wang, Xindi Wu, Zeyi Huang, and Eric P Xing. High-frequency component helps explain the generalization of convolutional neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8684–8694, 2020.

[33] Yinggui Wang, Jian Liu, Man Luo, Le Yang, and Li Wang. Privacy-preserving face recognition in the frequency domain. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 2558–2566, 2022.

[34] Mohammed Aledhari, Rehma Razzak, Reza M Parizi, and Fahad Saeed. Federated learning: A survey on enabling technologies, protocols, and applications. *IEEE Access*, 8:140699–140725, 2020.

[35] Kang Wei, Jun Li, Chuan Ma, Ming Ding, Sha Wei, Fan Wu, Guihai Chen, and Thilina Ranbaduge. Vertical federated learning: Challenges, methodologies and experiments. *arXiv preprint arXiv:2202.04309*, 2022.

[36] Ilia Sucholutsky and Matthias Schonlau. Secdd: Efficient and secure method for remotely training neural networks (student abstract). In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 15897–15898, 2021.

[37] Margarita Vinaroz and Mi Jung Park. Differentially private kernel inducing points using features from scatternets (dp-kip-scatternet) for privacy preserving data distillation, 2023.

[38] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[39] Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Towards causal representation learning, 2021.

[40] Bernhard Schölkopf. *Causality for Machine Learning*, page 765–804. ACM, February 2022.

[41] Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Toward causal representation learning. *Proceedings of the IEEE*, 109(5):612–634, 2021.

[42] Bernhard Schölkopf. Causality for machine learning. In *Probabilistic and Causal Inference: The Works of Judea Pearl*, pages 765–804. 2022.

[43] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27, 2014.

[44] Xiu-Shen Wei, Jian-Hao Luo, Jianxin Wu, and Zhi-Hua Zhou. Selective convolutional descriptor aggregation for fine-grained image retrieval. *IEEE transactions on image processing*, 26(6):2868–2881, 2017.

[45] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE international conference on computer vision*, pages 2736–2744, 2017.

[46] Tadeusz Caliński and Jerzy Harabasz. A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods*, 3(1):1–27, 1974.

[47] Michael W Marcellin, Michael J Gormish, Ali Bilgin, and Martin P Boliek. An overview of jpeg-2000. In *Proceedings DCC 2000. Data Compression Conference*, pages 523–541. IEEE, 2000.

[48] Shan Li, Weihong Deng, and JunPing Du. Reliable crowdsourcing and deep locality-preserving learning for expression recognition in the wild. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2584–2593. IEEE, 2017.

[49] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[50] Neeraj Kumar, Alexander C Berg, Peter N Belhumeur, and Shree K Nayar. Attribute and simile classifiers for face verification. In *2009 IEEE 12th international conference on computer vision*, pages 365–372. IEEE, 2009.

[51] Anna Montoya. State farm distracted driver detection. https://kaggle.com/competitions/state-farm-distracted-driver-detection.

[52] Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *2009 IEEE conference on computer vision and pattern recognition*, pages 413–420. IEEE, 2009.

[53] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[54] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2921–2929, 2016.

[55] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer, 2017.

[56] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017.

[57] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE transactions on image processing*, 26(7):3142–3155, 2017.

[58] Shi Guo, Zifei Yan, Kai Zhang, Wangmeng Zuo, and Lei Zhang. Toward convolutional blind denoising of real photographs. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1712–1722, 2019.

[59] Bhumika Gupta and Shailendra Singh Negi. Image denoising with linear and non-linear filters: a review. *International Journal of Computer Science Issues (IJCSI)*, 10(6):149, 2013.

[60] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8):2080–2095, 2007.

A Class-wise Noise & Class-agnostic Noise

This section provides a more intuitive explanation of Class-wise Noise and Class-agnostic Noise and their impact on model training. Noise was applied to 0.4% of all the pixels. The dataset used is RAF-DB, the model trained is ResNet34, and the task is 7-class Emotion Classification.

- *Class-wise Noise*: Each class label is assigned a unique noise, with 7 classes corresponding to 7 different noises.
- *Class-agnostic Noise*: Each sample is randomly assigned one of 7 different noises, with each class containing all 7 different noises.

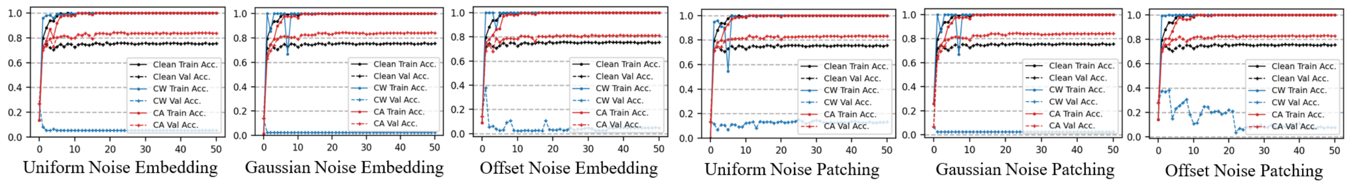


Figure 18: Training and Testing Accuracy Curves with Different Class-wise and Class-agnostic Noise. The X-axis represents Epochs; the Y-axis represents Accuracy. Class-wise Noise (CW); Class-agnostic Noise (CA). Embedding refers to adding noise by randomly masking, as described in our paper; Patching refers to adding noise by randomly selecting a continuous region.

As shown in Figure 18, the CA training/testing accuracy curves are almost identical to the Clean training/testing accuracy curves. In fact, the test accuracy of the model trained with class-agnostic noise is even higher than that of the model trained with clean data. On the other hand, for class-wise noise (CW), the model's training accuracy quickly reaches 100%, while the test accuracy is very poor, even worse than random classification. This indicates that when the noise is related to the class, the model tends to fit the simpler patterns of noise and class relationships, leading to overfitting to the noise.

B Details of Baseline Experiments

PPFR-FD. The block size is 8 for the DCT transform, and we only remove the DC component.

Fawkes. We reproduced Fawkes under our task setup using a ResNet-18 feature extractor ($\Phi(\cdot)$ in Eq.(18)). We set $\varepsilon = 0.0001$, $\lambda = 20$, and the number of iterations to 1500 per image for a better trade-off between privacy and usability.

$$\min_{\delta} \quad \varepsilon \cdot Dist(\Phi(x_T), \Phi(x \oplus \delta)) + \lambda \cdot DSSIM(x, x \oplus \delta). \quad (18)$$

Mixup. Mixup ensures model accuracy by increasing the amount of training data. To maintain comparability, we select N training samples from the randomly augmented space, where N is the number of samples in the original dataset.

DataDAM. DataDAM is the SOTA distribution-based distillation method. Specifically, we set the size of the synthetic (protected) dataset to be $1/3$ of the original dataset, using ConvNet as the feature extractor, and perform 10,000 iterations.

DataDAM+DPSGD. The parameter settings for data distillation are consistent with DataDAM. As outlined in [20], we add gaussian noise to the gradients to satisfy differential privacy constraints. Specifically, the clipping norm C is set to 1.0, and the noise multiplier is set to 0.02.

C Denoising Attacks

To validate the robustness against existing denoising techniques, we evaluated four commonly used denoising methods: the CNN-based method (DnCNN [57], CBDNet [58]) and the Filter-based method (Mean Filter [59], BM3D [60]). Figure 19 shows that our privacy protection method exhibits good resistance against denoising attacks. Even though ❹ and ❺ may recover some contour information, this is insufficient to constitute a privacy breach, as most of the privacy-related detailed information remains protected.

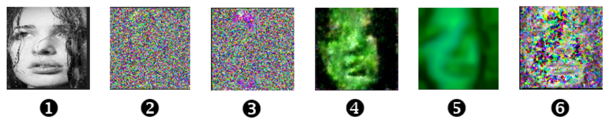


Figure 19: Recovered results. ❶: Clean, ❷: Protected, ❸: Recovered by DnCNN, ❹: Recovered by CBDNet, ❺: Recovered by BM3D, ❻: Recovered by Mean Filtering.

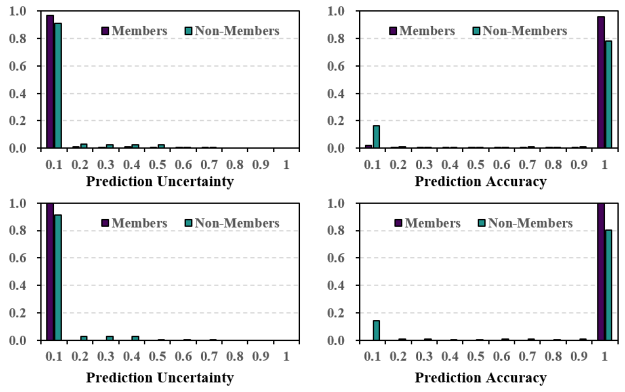


Figure 20: Prediction uncertainty (PU) and prediction accuracy (PA) for the members (clean training data) vs. non-members. The target model in the first row is trained on protected data, while the second row is trained on clean data.

D Membership Inference Attacks

Figure 20 uses PU and PA to evaluate the difficulty of membership inference attacks. The more similar the distributions of members and non-members are, the lower the success rate of the attack [56]. More intuitively, we use the Jensen-Shannon divergence to evaluate this similarity. PU and PA are 0.094 vs. 0.176 and 0.2 vs. 0.271, respectively (trained on protected data vs. trained on clean data). For the model trained on protected data, the distributions of members and non-members are more similar, making membership inference more difficult.