Achieving Zero-Glance Unlearning with Data-Free Inversion and Selective Parameters Suppression

1st Puwei Lian
College of Computer and Data Science
Fuzhou University
Fuzhou, China
lianpuwei@outlook.com

2nd Xiao Ke
College of Computer and Data Science
Fuzhou University
Fuzhou, China
kex@fzu.edu.cn

3rd Zhou Tan
College of Computer and Data Science
Fuzhou University
Fuzhou, China
zhoutan899@gmail.com

4th Jianping Cai
Faculty of Data Science
City University of Macau
Macao, China
jpcai@cityu.edu.mo

5th Ximeng Liu*
College of Computer and Data Science
Fuzhou University
Fuzhou, China
snbnix@gmail.com

***Abstract*—In machine learning (ML), data deletion involves more than just removing data from a dataset. Machine unlearning enables ML models to eliminate the effects of specific data that needs to be deleted. Under zero-glance settings, we may lack the right to utilize the data slated for removal during unlearning, thereby heightening the complexity. To address this challenge, we propose UISPS, which employs data-free inversion to generate replacement data for unavailable forgotten data. Utilizing the generated data, we propose selective parameter suppression to address the issue of catastrophic forgetting during unlearning effectively. Its interpretability improves the reliability of unlearning under zero-glance conditions. The experiments demonstrate that UISPS performs forgotten tasks with commendable results. Meanwhile, UISPS maintains higher accuracy on retained data, improving it by up to 3.34% while reducing the attack success rate of membership inference attacks by 25.85% $\sim$ 39.54% compared to the state-of-the-art.**

***Index Terms*—Machine Unlearning, Model Inversion, Catastrophic Forgetting, Privacy and Security.**

I. Introduction

Regulatory frameworks such as the EU's General Data Protection Regulation (GDPR) and the California Consumer Privacy Act (CCPA) have been established to enforce data protection. These regulations require companies and organizations to implement a "deletion on demand" framework, granting users the "right to be forgotten", which enables them to request the deletion of specific data along with its derivatives. The unique learning mechanisms employed in machine learning make these systems vulnerable to privacy attacks [1], [2]. Such attacks could potentially lead to unauthorized access to the target model and the extraction of private information. Therefore, when users request the removal of their data, merely deleting the corresponding entries from the database is insufficient. Thus, the innovative concept of machine unlearning has been introduced [3].

* is the corresponding author.

Machine unlearning refers to the ability of a trained model to forget a specific set of data. While it is undoubtedly possible to retrain a model using only the retained data, this approach can be computationally intensive. Initially, the focus on unlearning was mainly on simpler machine learning algorithms [4], [5], [6]. However, these methods tended to be effective primarily for small-scale problems. Many researchers have since explored the challenges of unlearning in deep neural networks [7], [8]. Despite their efforts, these methods still struggle with computational efficiency and time consumption, especially when tackling complex deep-learning architectures. Recently, Chundawat et al. [9] proposed a two-teacher model for unlearning. Foster et al. [10] introduced a model pruning technique for fast unlearning. Both approaches have yielded positive results with complex neural networks. However, a critical limitation remains: these algorithms are ineffective if the data to be forgotten is unavailable (zero-glance). The zero-glance conditions closely reflect real-life scenarios where users request the deletion of their data, indicating their desire for this data not to be stored or utilized by the model owner. Graves et al. [11] proposed a method that involves storing past parameter updates and reversing the corresponding data updates during the unlearning process; however, this approach can consume considerable storage space. Tarun et al. [12] introduced a technique that generates error-maximizing noise for each class selected for forgetting (UNSIR). While this method is intuitively appealing, it lacks a certain level of interpretability, as it may inadvertently compromise the parameters linked to the retained data. After the unlearning process, UNSIR typically requires one epoch or more for repair operations, as it may suffer from catastrophic forgetting.

To address the above problem, we propose the Unlearning with Data-Free Inversion and Selective Parameters Suppression framework (UISPS), considering data-free inversion to generate replacement data for unavailable forgotten data. Utilizing the generated data, we propose selective parameters

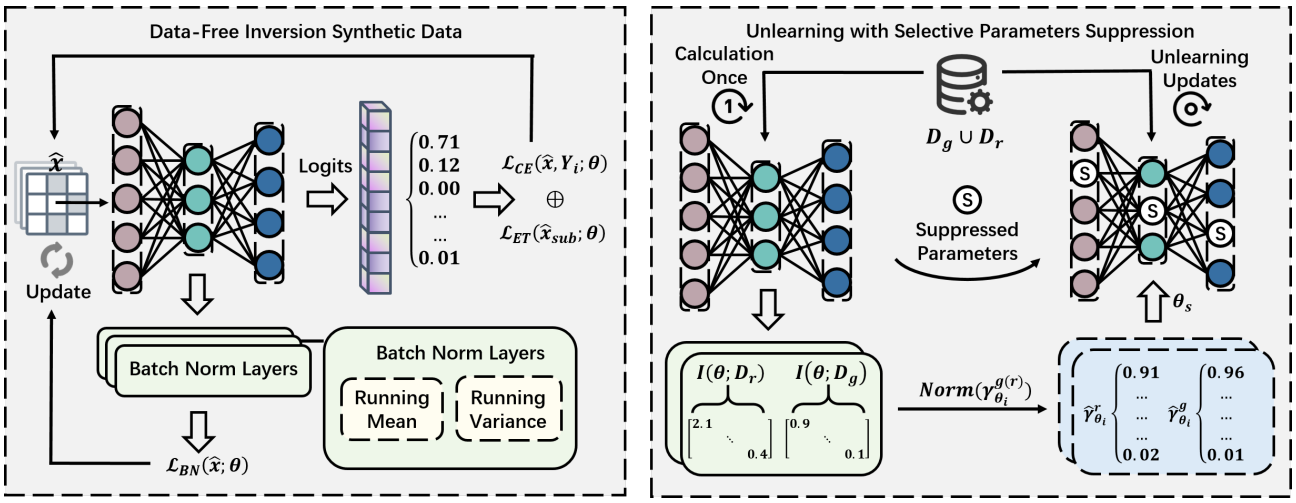


Fig. 1: Overview of UISPS.

suppression (SPS), which identifies the parameters that contribute more to the retained data and less to the forgotten data by the diagonal of the Fisher Information Matrix (FIM) and suppresses their changes, reliably mitigating the oscillations in the model performance and guaranteeing the excellent performance of the model on the retained data. Our main contributions are as follows:

- In this work, we propose the UISPS framework, which employs data-free inversion to generate replacement data for unavailable forgotten data. This approach offers a new perspective for addressing class-level unlearning under zero-glance conditions.
- We propose Selective Parameter Suppression. By analyzing the diagonal of the FIM, we can suppress parameters that contribute more to retained data and less to forgotten data. This added interpretability enhances the reliability of unlearning operations while mitigating the issue of catastrophic forgetting.
- Compared to previous methods, UISPS effectively eliminates memory related to forgotten class from the model while demonstrating superior performance on retained classes. UISPS outperforms state-of-the-art (SOTA) in terms of model performance and forgetfulness, and it performs robustly against member inference attacks.

II. RELATED WORK

A. Machine Unlearning

Bourtoule et al. [3] proposed dividing the training dataset into mutually non-overlapping slices and using multiple models to train disjoint slices separately, resulting in numerous weak learning models. The unlearning effect can be achieved by retraining the weak learning model corresponding to the deleted data. Grounded in Certified Removal Mechanisms, Guo et al. [13] solved the unlearning problem for linear models and convex losses. The data removal problem under the Random Forest algorithm was studied by Brophy et al. [4]. Methods proposed by Ginart et al. [5] and Mirzasoleiman et al. [6] support data deletion in the k-means clustering problem. In [14], the authors proposed an unlearning approach to support a stream of data deletion requests.

B. Deep Machine Unlearning

In deep learning, it's challenging to trace the impact of each data piece accurately, thus adding complexity to unlearning under deep neural networks. Golatkar et al. [7] proposed an information-theoretic method to erase information from intermediate layers of deep networks trained with SGD. Graves et al. [11] stored model updates for each data during training and completed unlearning by deleting historical updates. Model pruning has also been shown to be a way to achieve unlearning [15], [10]. Unlearning based on the teacher-student framework has also been studied [9]. Tarun et al. [12] investigated the unlearning problem with no access to forgotten data for the first time.

III. PRELIMINARIES

A. Unlearning Settings

We assume that the complete dataset is $D_c = \{(x_i, y_i)\}_{i=1}^{n}$, where x_i represents the i^{th} sample, $y_i \in \{1, \ldots, K\}$ denotes the label corresponding to that sample, and n represents the number of data. The samples in the forgotten set are denoted as D_f. In this paper, we study unlearning at the class level, so D_f corresponds to the class data that needs to be forgotten. Also, we set the retained dataset to D_r, representing the retained class data. Thus, D_f and D_r together denote the entire dataset, and they are mutually exclusive, *i.e.*, $D_f \cup D_r = D_c$ and $D_f \cap D_r = \phi$.

B. Zero-glance Settings

Tarun et al. [12] introduced a more stringent privacy setting called zero-glance privacy scenario, *i.e.*, the model doesn't have access to the data that need to be forgotten, wherein the data in question should be promptly removed from the dataset when the user proposes to delete their data. Following [12],

we consider class-level unlearning. When a class is proposed for deletion, we only know its corresponding labels Y_i, yet we lack access to its corresponding dataset D_f. So, we can't use the data from that class. All experiments in this paper will be performed under zero-glance conditions.

IV. METHODOLOGY

A. Overview

UISPS provides a new perspective on the problem of unlearning under zero-glance conditions. It consists of two steps, as shown in Fig.1. i) Data-Free Inversion Synthetic Data: generate replacement data for the class that needs to be forgotten; ii) Unlearning with Selective Parameters Suppression: Suppress changes in parameters that are important for retained data performance during unlearning. *Algorithm 1 in the Appendix provides a detailed process.*

B. Data-Free Inversion Synthetic Data

We assume that the model for which unlearning is to be performed is M and its model parameters are θ. First, we randomly initialize a batch of samples $\hat{x}$. Our objective is to systematically transform this batch of samples into replacement samples of forgotten data. We call the data obtained after inversion D_g. The initial step is to ensure the recognition of these samples by M as belonging to the class to be forgotten. We employ cross-entropy loss to iteratively update the samples $\hat{x}$ in the direction where M outputs labels corresponding to the forgotten class:

$$\mathcal{L}_{CE}(\hat{x}, Y_i; \theta) = CE(M(\hat{x}; \theta), Y_i), \tag{1}$$

where $CE(\cdot)$ represents the cross-entropy.

Following the proposal in [16], we assume that feature statistics adhere to a Gaussian distribution across batches, defined by mean μ and variance σ^2. Subsequently, the regularization term for feature distribution can be formulated as:

$$\begin{aligned}\mathcal{L}_{BN}(\hat{x}; \theta) = &\sum_l \|\mu_l(\hat{x}) - \mathbb{E}\left(\mu_l(x)\right)\|_2 + \\ &\sum_l \left\|\sigma_l^2(\hat{x}) - \mathbb{E}\left(\sigma_l^2(x)\right)\right\|_2,\end{aligned} \tag{2}$$

where $\mu_l(\hat{x})$ and $\sigma_l^2(\hat{x})$ are the batch-wise mean and variance estimates of feature maps corresponding to the l^{th} convolutional layer. The $\mathbb{E}(\cdot)$ and $\|\cdot\|$ operators denote the expected value and ℓ_2 norm calculations, respectively. A BN layer normalizes the feature maps during training to alleviate covariate shifts. Thus, the values of $\mathbb{E}\left(\mu_l(x)\right)$ and $\mathbb{E}\left(\sigma_l^2(x)\right)$ approximate the values of running mean and running variance obtained from the l^{th} BN layer.

With the loss function described above, we have been able to generate replacement data; however, we observe that the synthesized data may be far from the decision boundary with high confidence. While the high-confidence data represents the main component of knowledge, it also lacks key features about the model's decision boundary, as shown in the left panel of Fig.2. We know that during class-level unlearning, the decision boundary of the forgotten class will change. The lack of decision boundary features may result in the decision boundary not being adjusted in the most appropriate direction during unlearning. Our ablation experiments have also demonstrated that there may be a risk of residual knowledge. Low-confidence data provide more key features about decision boundaries than high-confidence data because these data have higher entropy [17]. Therefore, we aim to generate some data far from the decision boundary. So, we use entropy-based partial regularization intending to generate both high and low-confidence data as shown in the right panel of Fig.2. We assume that $\hat{x}_{sub} \subseteq \hat{x}$ and take out the data $\hat{x}_{sub}$ for regularization:

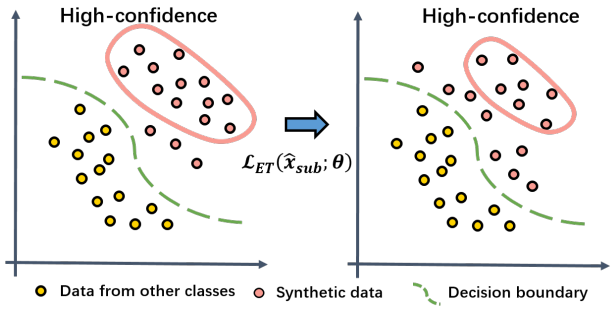


Fig. 2: The illustration of generated data and decision boundary. **Left panel:** Synthetic data are far from the decision boundary. **Right panel:** We can generate more data near the decision boundaries by utilizing entropy-based partial regularization.

$$p(y|\hat{x}_{sub}, \theta) = Softmax(M(\hat{x}_{sub}; \theta)), \tag{3}$$

$$\mathcal{L}_{ET}(\hat{x}_{sub}; \theta) = -\sum_{y=1}^{c} p(y|\hat{x}_{sub}, \theta) \log p(y|\hat{x}_{sub}, \theta), \tag{4}$$

where c denote the class index. By combining the above losses, we can obtain the model inversion loss as follows:

$$\mathcal{L}_G = \mathcal{L}_{CE}(\hat{x}, Y_i; \theta) + \lambda_1 \mathcal{L}_{BN}(\hat{x}; \theta) + \lambda_2 \mathcal{L}_{ET}(\hat{x}_{sub}; \theta), \tag{5}$$

where λ_1, λ_2 denote the weights corresponding to the respective losses. Through continuous updating, we will obtain the generated data D_g.

C. Unlearning with Selective Parameters Suppression

There is no direct mapping relationship between the parameters of the neural network and the data. Suppose we directly destroy the parameters related to the forgotten data during unlearning. Due to the inherent knowledge entanglement in neural networks, the parameters associated with the retained data will also be partially affected, contributing to the phenomenon known as catastrophic forgetting. We hope to minimize updates that harm model performance while ensuring that unlearning proceeds normally. Therefore, we propose selective parameter suppression to achieve this goal. Fisher Information Matrix $I(\theta; D)$ is defined as the negative

TABLE I: Unlearning Results on ResNet18

Method	1-CIFAR-10			3-CIFAR-10			1-SVHN			3-SVHN		
	A_{D_r}	A_{D_f}	MIA	A_{D_r}	A_{D_f}	MIA	A_{D_r}	A_{D_f}	MIA	A_{D_r}	A_{D_f}	MIA
Original	94.26	93.37	99.42	95.37	92.43	99.67	96.07	97.75	99.97	96.80	96.73	99.94
Retrain	94.00	0.00	71.64	96.57	0.00	46.71	96.38	0.00	8.72	97.53	0.00	14.91
Fisher	21.56	0.22	49.21	18.18	0.00	31.58	43.51	0.00	10.17	29.23	0.00	20.12
Fine-Tune	82.49	1.20	73.72	85.61	0.10	49.64	90.90	0.14	44.88	92.33	0.00	60.79
Amnesiac	87.59	0.46	50.66	89.37	0.00	50.61	92.11	0.00	16. 34	93.19	0.00	19.88
UNSIR	87.50	8.43	68.41	90.32	2.99	65.96	93.17	3.00	19.44	93.99	0.72	24.01
UISPS	**89.30**	**0.00**	**33.92**	**93.66**	**0.00**	**26.42**	**94.16**	**0.00**	**4.31**	**95.48**	**0.00**	**9.79**

of the expected value of the second derivative of the log-likelihood function:

$$I(\theta; D) = -\mathbb{E}_D \left[\frac{\partial^2 \ln p(y_i|\theta, x_i)}{\partial \theta^2} \right], \tag{6}$$

where $D = \{(x_i, y_i)\}_{i=1}^N$, N is the number of samples in D. We can express the FIM as the expected value of the outer product of the first derivative:

$$I(\theta; D) = \mathbb{E}_D \left[\frac{\partial \ln p(y_i|\theta, x_i)}{\partial \theta} \right] \left[\frac{\partial \ln p(y_i|\theta, x_i)}{\partial \theta} \right]^T. \tag{7}$$

The likelihood function is very complex, and it isn't easy to calculate the expectation. We can approximate its value with the expectation of the empirical distribution:

$$I(\theta; D) \approx \frac{1}{N} \sum_{i=1}^{N} \left[\frac{\partial \ln p(y_i|\theta, x_i)}{\partial \theta} \right] \left[\frac{\partial \ln p(y_i|\theta, x_i)}{\partial \theta} \right]^T. \tag{8}$$

Even without explicitly computing the second-order derivatives, the cost of computing the FIM is still large. Therefore, we only rely on the diagonal of the FIM to judge the importance of the parameters. So it can be simplified to:

$$I(\theta; D) \approx \frac{1}{N} \sum_{i=1}^{N} \left[\frac{\partial \ln p(y_i|\theta, x_i)}{\partial \theta} \right]^2. \tag{9}$$

It allows us to estimate the uncertainty of a parameter without complex calculations. We then use a small subset of the retained data $D'_r \subseteq D_r$ and generated data D_g to calculate the importance of the model parameters, respectively:

$$\gamma^g_{\theta_i} \leftarrow I(\theta; D_g), \gamma^r_{\theta_i} \leftarrow I(\theta; D'_r), i \in [0, |\theta|], \tag{10}$$

where $\gamma^g_{\theta_i}$ ($\gamma^r_{\theta_i}$) is the i-th element of the diagonal of the FIM, calculated over the set $D_g(D'_r)$. We then normalize separately to get the importance scores of the parameters on the two datasets:

$$\hat{\gamma}^{g(r)}_{\theta_i} = Norm(\gamma^{g(r)}_{\theta_i}) = \frac{\gamma^{g(r)}_{\theta_i} - \min(\gamma^{g(r)}_{\theta})}{\max(\gamma^{g(r)}_{\theta}) - \min(\gamma^{g(r)}_{\theta})}. \tag{11}$$

We hope to suppress the parameters that contribute less to the forgotten data and more to the retained data. This allows us to suppress the performance oscillation of the model while updating some parameters to achieve the purpose of unlearning. Therefore, we filter out the suppressed parameters based on the normalized scores:

$$\theta_s = \begin{cases} \theta_s \cup \theta_i, & \text{if } \hat{\gamma}^r_{\theta_i} > \alpha \hat{\gamma}^g_{\theta_i}, \\ \theta_s, & \text{if } \hat{\gamma}^r_{\theta_i} < \alpha \hat{\gamma}^g_{\theta_i}. \end{cases} \quad \forall i \in [0, |\theta|], \tag{12}$$

where θ_s is initialized to empty. Then, during the unlearning process, these parameters will not be updated. We relabelled the generated data D_g using the untrained model M_t with parameter θ_t, thus obtaining the new soft label:

$$D'_g := \{(x, y') : (x, y) \in D_g, y' \leftarrow M_t(x; \theta_t)\}. \tag{13}$$

Such a labeling approach is intended to make the model M perform as well as the untrained model on the forgotten data, achieving the purpose of unlearning. We then use cross-entropy loss $\mathcal{L}_{unlearn}$ to train the model with one epoch on the generated data and the retained data:

$$\mathcal{L}_{unlearn}(\bar{x}, \bar{y}; \theta) = CE(M(\bar{x}; \theta), \bar{y}), \tag{14}$$

where $(\bar{x}, \bar{y}) \subseteq (D_r \cup D'_g)$. Finally, we will complete the unlearning by updating the model as follows:

$$\theta'_i = \begin{cases} \theta_i & \text{if } \theta_i \in \theta_s, \\ \theta_i + \lambda \nabla_{\theta_i} \mathcal{L}_{unlearn}(\bar{x}, \bar{y}; \theta) & otherwise. \end{cases} \tag{15}$$

V. Experiments

We demonstrate the effectiveness of our method in forgetting single class and multiple classes across diverse settings. For our experiments and evaluations, we employed various deep neural networks, namely ResNet18 [18] and AllCNN [19], to showcase the superiority of the UISPS. The effectiveness of UISPS is validated on different datasets, including CIFAR-10 [20], CIFAR-100 [20], and SVHN [21].

A. Experimental Settings

All models were trained on the complete dataset for 50 epochs until convergence initially. The models retrained on the retained data were trained for 50 epochs. The experiments were executed using an NVIDIA Tesla-V100 (16GB) GPU.

Implementation details of model inversion: We synthesize 32 samples for each forgotten class, with the learning rate of 0.01, λ_1 is set to 0.01, λ_2 to 0.05. We perform 1000 iterations for each batch of synthetic data.

TABLE II: Ablation Experiments on ResNet18.

ENT	SPS	Metrics	SVHN				CIFAR-10				CIFAR-100			
			1	3	5	7	1	3	5	7	20	40	60	80
×	×	A_{D_r}	89.56	92.10	92.83	93.02	87.11	88.47	91.19	91.57	66.99	69.71	70.02	79.55
		A_{D_f}	0.48	0.33	0.02	0.00	3.59	1.52	0.99	0.84	2.10	0.98	1.01	0.33
×	✓	A_{D_r}	93.41	95.18	95.67	96.34	88.19	93.14	95.36	95.88	69.56	72.69	75.41	81.65
		A_{D_f}	0.99	0.56	0.00	0.00	4.53	2.39	2.44	1.90	1.52	1.58	0.46	0.00
✓	×	A_{D_r}	91.12	92.69	93.18	93.28	87.92	89.14	92.05	92.27	67.43	69.19	72.31	77.80
		A_{D_f}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
✓	✓	A_{D_r}	**94.16**	**95.48**	**95.93**	**96.38**	**89.30**	**93.66**	**95.58**	**95.68**	**69.84**	**72.81**	**75.42**	**81.79**
		A_{D_f}	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**

Implementation details of unlearning: The model was trained using synthetic and retained data with a batch size of 128. On CIFAR-10, the learning rate for ResNet18 is 0.002, and AllCNN is 0.03. For CIFAR-100, ResNet18 uses a learning rate of 0.002. On SVHN, the learning rates are 0.003 for ResNet18 and 0.03 for AllCNN. We set α to 1.5 and trained the model for one epoch to complete the unlearning.

Compared methods: We compare our method with various methods applicable to zero-glance conditions. 1) UNSIR [12]: unlearning based on error-maximizing noise. 2) Fine Tune: the fine-tuned model on D_r. 3) Amnesiac [11]: unlearning by canceling the parameters update. 4) Fisher Forgetting [7]. 5) Retrain: the model trained on retain set D_r.

Evaluation Metrics: In our experimental analysis, we used the following metrics to determine the effectiveness of the unlearning method. 1) **Accuracy on forget set** (A_{D_f}): Should be similar to the retrained model, the accuracy of the forgotten class of the retrained model is usually 0%. This is because after the model removes the knowledge of a class, it tends to be more inclined to other classes that have been learned when making predictions. 2) **Accuracy on retain set** (A_{D_r}): Should be similar to the retrained model or original model. 3) **Membership inference attack** (MIA): The attack is performed to check if any information about the forgotten data remains in the model. The attack success rate should be lower on the forget set in the unlearned model.

TABLE III: Results of ResNet18 on CIFAR-100.

Method	40-CIFAR-100		60-CIFAR-100		80-CIFAR-100	
	A_{D_r}	A_{D_f}	A_{D_r}	A_{D_f}	A_{D_r}	A_{D_f}
Original	75.21	74.27	75.12	74.90	78.16	74.13
Retrain	76.09	0.00	78.35	0.00	83.72	0.00
Fine-Tune	66.78	1.24	70.11	0.16	75.63	0.03
Amnesiac	71.09	3.47	73.99	6.97	80.65	0.77
UNSIR	70.77	1.24	74.34	0.28	80.51	0.00
UISPS	**72.81**	**0.44**	**75.42**	**0.11**	**81.79**	**0.00**

B. Model Performance Analysis

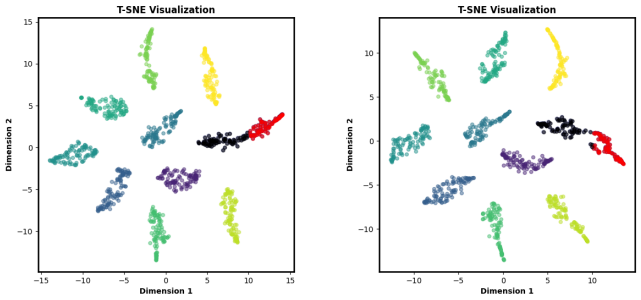
We show the 1 and 3 class(es) unlearning results using ResNet18 in Table I. UISPS effectively eliminates information related to the forgotten data in the model, resulting in a 0% accuracy on A_{D_f}. This is attributed to low-entropy data guiding changes in decision boundaries. UNSIR and Fine-Tune typically suffer from memory residue in forgetting tasks. Amnesiac performs well on forgetting tasks, but it requires many historical updates to support it. Simultaneously, while ensuring forgetting, we achieve high accuracy on retained data (D_r). Our method showed better results than UNSIR and Amnesiac. The A_{D_r} of ResNet18 can be improved by up to 3.34%, compared to the state-of-the-art. This is due to SPS's suppression of essential parameters, so the model's performance in retained data will not experience significant drift. To demonstrate the superiority of UISPS, we conducted a large-scale unlearning experiment on CIFAR-100 and performed experiments with forgetting 40, 60, and 80 classes. According to Table III, our method enables the model to maintain the best performance, improving model performance by up to 2.04%. It is worth noting that UISPS doesn't need to fine-tune or repair the model compared with UNSIR, and it does not take a lot of space to store a large amount of updated data as [11]. *Further results on AllCNN are provided in Appendix Tables IV and V.*

C. Membership Inference Attacks

We report the attack success rate (ASR) of various unlearning methods under MIA in Table I. Experiments show that UISPS has excellent defense capabilities against membership inference attacks among all methods. Compared with the original model, we successfully reduced ASR by 81.09% on average and up to 95.66%. At the same time, compared to SOTA, we reduce ASR by 25.85% on average and up to 39.54%. This shows that UISPS removes the influence of data from the model more efficiently to show the probability distribution of the data on which it has not been trained. Our method is better at making the model forget knowledge rather than covering up traces of past learning. This fully reflects the robustness of UISPS and reduces the risk of privacy exposure to forgotten data.

D. Ablation Experiments

As depicted in Table II, we present the results of the ablation experiments on the ResNet18 when forgetting different numbers of classes. Experiments show that when entropy-based regularization is added, the model can better adjust



(a) CIFAR-10 (b) SVHN

Fig. 3: T-SNE visualization of ResNet18 on CIFAR10 and SVHN. The red part is the data that needs to be forgotten, and the black part is the generated data. The other parts are data of other classes.

the decision boundary, reducing A_{D_f} by up to 4.53% and maintaining it stably at 0%. By adding SPS, it can effectively prevent parameter adjustments that affect model performance, increasing A_{D_f} by 2.19% $\sim$ 5.40%. Experiments have shown that our method can effectively improve the completion quality of unlearning tasks. In addition, we tested the transferability of SPS using other methods. We employed a two-teacher distillation model [9] for unlearning. *The transferability of SPS is provided relevant analysis in the Appendix.*

E. Analysis of Model Inversion Results

We employ T-SNE to visualize the features extracted by the network, demonstrating that the generated data can serve as replacement data for the forgotten class data. As illustrated in Fig. 3, the features of the generated data closely align with those of the data that need to be replaced. With the incorporation of entropy-based regularization, the generated images become closer to data from other classes and exhibit higher entropy. This observation supports our hypothesis and confirms the effectiveness of data-free model inversion.

VI. Conclusion

In this work, we propose UISPS framework which generates data of forgotten classes through data-free inversion and unlearning with selective parameters suppression to mitigate the deterioration of model performance. It provides a more interpretable unlearning perspective and a simple and efficient solution to catastrophic forgetting. Experiments in different settings verify the effectiveness of our method. Compared with SOTA, we show better performance in both model performance and anti-privacy attack capabilities.

References

[1] Xin Liu, Yue Xu, and Kun He, "Improving the sar image adversarial transferability through dual-loop ensemble gradient attack," in *2024 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 2024, pp. 1–6.

[2] Yuchen Wang, Xiaoguang Li, Li Yang, Lu Zhou, Jianfeng Ma, and Hui Li, "Adaptive oriented adversarial attacks on visible and infrared image fusion models," in *2024 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 2024, pp. 1–6.

[3] Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot, "Machine unlearning," in *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2021, pp. 141–159.

[4] Jonathan Brophy and Daniel Lowd, "Machine unlearning for random forests," in *International Conference on Machine Learning*. PMLR, 2021, pp. 1092–1104.

[5] Antonio Ginart, Melody Guan, Gregory Valiant, and James Y Zou, "Making ai forget you: Data deletion in machine learning," *Advances in neural information processing systems*, vol. 32, 2019.

[6] Baharan Mirzasoleiman, Amin Karbasi, and Andreas Krause, "Deletion-robust submodular maximization: Data summarization with "the right to be forgotten"," in *International Conference on Machine Learning*. PMLR, 2017, pp. 2449–2458.

[7] Aditya Golatkar, Alessandro Achille, and Stefano Soatto, "Eternal sunshine of the spotless net: Selective forgetting in deep networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9304–9312.

[8] Aditya Golatkar, Alessandro Achille, Avinash Ravichandran, Marzia Polito, and Stefano Soatto, "Mixed-privacy forgetting in deep networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognitionn*, 2021, pp. 792–801.

[9] Vikram S Chundawat, Ayush K Tarun, Murari Mandal, and Mohan Kankanhalli, "Can bad teaching induce forgetting? unlearning in deep networks using an incompetent teacher," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023, vol. 37, pp. 7210–7217.

[10] Jack Foster, Stefan Schoepf, and Alexandra Brintrup, "Fast machine unlearning without retraining through selective synaptic dampening," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024, vol. 38, pp. 12043–12051.

[11] Laura Graves, Vineel Nagisetty, and Vijay Ganesh, "Amnesiac machine learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021, vol. 35, pp. 11516–11524.

[12] Ayush K Tarun, Vikram S Chundawat, Murari Mandal, and Mohan Kankanhalli, "Fast yet effective machine unlearning," *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[13] Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten, "Certified data removal from machine learning models," *arXiv preprint arXiv:1911.03030*, 2019.

[14] Varun Gupta, Christopher Jung, Seth Neel, Aaron Roth, Saeed Sharifi-Malvajerdi, and Chris Waites, "Adaptive machine unlearning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 16319–16330, 2021.

[15] Jiancheng Liu, Parikshit Ram, Yuguang Yao, Gaowen Liu, Yang Liu, PRANAY SHARMA, Sijia Liu, et al., "Model sparsity can simplify machine unlearning," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[16] Hongxu Yin, Pavlo Molchanov, Jose M Alvarez, Zhizhong Li, Arun Mallya, Derek Hoiem, Niraj K Jha, and Jan Kautz, "Dreaming to distill: Data-free knowledge transfer via deepinversion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8715–8724.

[17] Huan Liu, Li Gu, Zhixiang Chi, Yang Wang, Yuanhao Yu, Jun Chen, and Jin Tang, "Few-shot class-incremental learning via entropy-regularized data-free replay," in *European Conference on Computer Vision*. Springer, 2022, pp. 146–162.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[19] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller, "Striving for simplicity: The all convolutional net," *arXiv preprint arXiv:1412.6806*, 2014.

[20] Alex Krizhevsky, Geoffrey Hinton, et al., "Learning multiple layers of features from tiny images," 2009.

[21] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, et al., "Reading digits in natural images with unsupervised feature learning," in *NIPS workshop on deep learning and unsupervised feature learning*. Granada, Spain, 2011, vol. 2011, p. 7.

VII. Appendix

A. Detailed Process of the Algorithm

Algorithm 1 UISPS

Input: Training iterations E, trained model M, synthetic class Y_i, retained data D_r, learning rate λ.

Output: Unlearned model M_u with parameter θ_u.

Randomly initialize model M_t and initialize model M with pretrained parameters θ. Randomly initialized $\hat{x}$ to the same size as the input of M. Initialize $D_g \in \phi, \theta_s \in \phi$.

// Data-Free Inversion Synthetic Data

for $l = 1...L$ **do**

  $\mathbb{E}(\mu_l(x)) \leftarrow$ running mean of l^{th} BN layer of M

  $\mathbb{E}(\sigma_l^2(x)) \leftarrow$ running variance of l^{th} BN layer of M

end for

for $e = 1...E$ **do**

  Calculate cross-entropy loss $\mathcal{L}_{CE}$ ▷$Eq.(1)$

  for $l = 1...L$ **do**

    Calculate feature distribution loss $\mathcal{L}_{BN}$ ▷$Eq.(2)$

  end for

  Calculate probability distributions p ▷$Eq.(3)$

  Calculate the entropy of distributions $\mathcal{L}_{ET}$ ▷$Eq.(4)$

  Update $\hat{x}$ via total loss $\mathcal{L}_G$ ▷$Eq.(5)$

end for

Obtaining synthetic data $D_g = D_g \cup \hat{x}$

// Unlearning with Selective Parameters Suppression

$D'_r \leftarrow$ Take a small portion from the retained data D_r

$\gamma^g_{\theta_i} \leftarrow$ Calculate the FIM diagonal on data D_g ▷$Eq.(9)$

$\gamma^r_{\theta_i} \leftarrow$ Calculate the FIM diagonal on data D'_r ▷$Eq.(9)$

$\hat{\gamma}^r_{\theta_i} \leftarrow Norm(\gamma^r_{\theta_i}), \hat{\gamma}^g_{\theta_i} \leftarrow Norm(\gamma^g_{\theta_i})$ ▷$Eq.(11)$

for $i = 0...|\theta|$ **do**

$$\theta_s = \begin{cases} \theta_s \cup \theta_i, & \text{if } \hat{\gamma}^r_{\theta_i} > \alpha \hat{\gamma}^g_{\theta_i} \\ \theta_s, & \text{if } \hat{\gamma}^r_{\theta_i} < \alpha \hat{\gamma}^g_{\theta_i} \end{cases}$$

end for

$D'_g \leftarrow$ Relabel D_g using M_t ▷$Eq.(13)$

for $(\bar{x}, \bar{y})$ in $(D'_g \cup D_r)$ **do**

  Calculate cross entropy loss $\mathcal{L}_{unlearn}$ ▷$Eq.(14)$

  Update model parameters θ ▷$Eq.(15)$

end for

return model M

B. Additional Experimental Results

TABLE IV: Results of AllCNN on CIFAR-10.

Method	1-CIFAR-10		2-CIFAR-10		3-CIFAR-10	
	A_{D_r}	A_{D_f}	A_{D_r}	A_{D_f}	A_{D_r}	A_{D_f}
Original	86.31	93.51	88.16	89.90	89.00	82.18
Retrain	86.57	0.00	89.56	0.00	91.96	0.00
Fine-Tune	81.15	5.03	84.20	0.19	88.18	2.72
Amnesiac	79.91	1.04	85.49	0.00	88.15	0.00
UNSIR	78.90	9.70	84.67	4.28	87.86	3.00
UISPS	**81.26**	**0.00**	**86.44**	**0.00**	**89.16**	**0.00**

TABLE V: Results of AllCNN on SVHN.

Method	1-SVHN		2-SVHN		3-SVHN	
	A_{D_r}	A_{D_f}	A_{D_r}	A_{D_f}	A_{D_r}	A_{D_f}
Original	95.22	96.72	95.31	95.90	95.27	95.81
Retrain	95.34	0.00	96.90	0.00	96.90	0.00
Fine-Tune	93.05	1.84	94.14	0.36	94.13	0.03
Amnesiac	94.12	0.00	95.19	0.00	95.65	0.00
UNSIR	93.11	1.42	94.35	2.80	94.92	0.00
UISPS	**94.46**	**0.00**	**95.64**	**0.00**	**96.70**	**0.00**

C. Using Different Proportions of Retain Data

In some cases, we may not have access to all of the retained data, or the volume of retained data is so extensive that utilizing all of it becomes prohibitively expensive. Therefore, we explored the model's performance with varying proportions of retained data. The detailed results are in Tables VI and VII. Experiments were conducted on the SVHN dataset using 80%, 60%, 40%, and 20% retained data. Our method maintains good performance with lower percentages of retained data; even if only 20% of the data is retained, the A_{D_r} only dropped by 1.83% $\sim$ 3.38%. This shows that UISPS is still effective even under more demanding conditions.

TABLE VI: Multiple class unlearning (3 classes) in AllCNN with different proportions of retained data on SVHN.

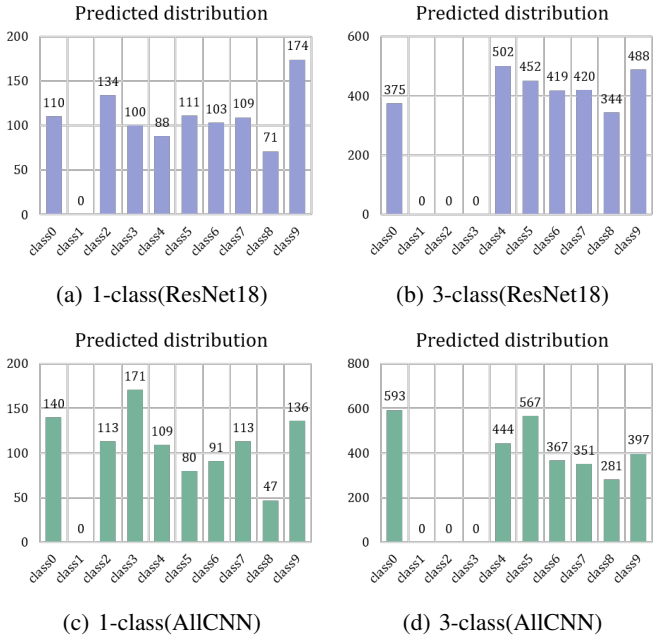
Percentage	Metrics	Original	Retrain	UNSIR	Fine-Tune	**UISPS**
80%	A_{D_r}	95.27	96.62	94.47	92.31	**95.41**
	A_{D_f}	95.81	0.00	0.42	0.13	**0.00**
60%	A_{D_r}	95.27	96.23	93.88	90.82	**95.02**
	A_{D_f}	95.81	0.00	0.74	0.07	**0.00**
40%	A_{D_r}	95.27	96.02	93.55	90.11	**94.75**
	A_{D_f}	95.81	0.00	1.03	0.00	**0.00**
20%	A_{D_r}	95.27	94.00	92.61	89.53	**93.42**
	A_{D_f}	95.81	0.00	0.16	0.00	**0.00**

TABLE VII: Multiple class unlearning (3 classes) in ResNet18 with different proportions of retained data on SVHN.

Percentage	Metrics	Original	Retrain	UNSIR	Fine-Tune	**UISPS**
80%	A_{D_r}	96.80	97.48	93.91	92.03	**95.42**
	A_{D_f}	96.73	0.00	0.46	0.43	**0.00**
60%	A_{D_r}	96.80	97.38	93.51	91.61	**94.92**
	A_{D_f}	96.73	0.00	0.88	0.16	**0.00**
40%	A_{D_r}	96.80	96.71	93.16	89.28	**94.53**
	A_{D_f}	96.73	0.00	1.89	0.05	**0.00**
20%	A_{D_r}	96.80	93.65	92.28	89.23	**93.65**
	A_{D_f}	96.73	0.00	2.08	0.00	**0.00**

TABLE VIII: Transferability of SPS.

Method	Metrics	SVHN				CIFAR-10				CIFAR-100			
		1	3	5	7	1	3	5	7	20	40	60	80
TTM	A_{D_r}	92.80	95.02	94.94	95.05	76.49	81.61	87.69	92.17	68.19	69.42	72.99	76.52
	A_{D_f}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
+SPS	A_{D_r}	94.05	95.12	96.03	96.46	76.98	82.36	89.58	92.71	69.46	69.98	73.67	76.65
	A_{D_f}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00



(a) 1-class(ResNet18) (b) 3-class(ResNet18)

(c) 1-class(AllCNN) (d) 3-class(AllCNN)

Fig. 4: Prediction distribution of the unlearned model on forgotten class(es) of data.

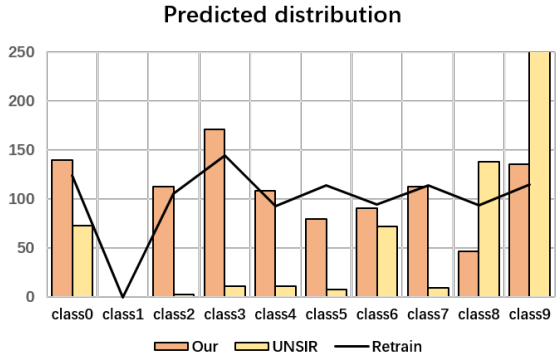


Fig. 5: comparison of the distribution of predictions of our method with SOTA and retrain model in the forgotten class. Compared to SOTA, our approach brings the distribution of model predictions closer to the retrained model.

TABLE IX: Time Overhead.

Metrics	UNSIR	Amnesiac	Fine-Tune	Retrain	**UISPS**
Time	173.08s	192.30s	221.15s	1355.75s	**194.10s**

D. Transferability of SPS:

To showcase the transferability of SPS, we employed a two-teacher distillation model for the forgetting task, comparing the model's performance before and after using selective parameters suppression. We use AllCNN to test single-class and multi-class unlearning on three datasets. Table VIII shows that SPS can improve the model's performance, demonstrating high transferable to other methods. This is because in most unlearning approaches, ensuring model performance needs to be attended to alongside forgetting, whereas SPS measures a good balance between the two.

E. Prediction Distribution for the Forget Class of Data

In Fig.4, we depict the distribution of predictions by the unlearned model on CIFAR-10 for the forgetting class(es). For single-class forgetting, we illustrate the predictions of ResNet18 and AllCNN for the forgetting class (class1). For multi-class forgetting, we showcase the prediction distribution of ResNet18 and AllCNN for the forgetting classes (class1, class2, class3). The results demonstrate that our method does not induce the model to predict a clear pattern on the forgotten data; instead, it is closer to a random distribution. Such a distribution ensures that the forgotten classes are unrelated to the data of other retained classes, thereby reducing the possibility of countering attacks. Furthermore, we compare the distribution of predictions in the forgetting class with the SOTA and retrained model, as shown in Fig. 5. The experiment reveals that the predictions of the unlearned model closely resemble those of the model that has not encountered the forgotten class(es). UNSIR can induce a specific pattern in the model predictions, and the displayed prediction distribution may deviate significantly from that of the retrained model. This further underscores, by comparison with SOTA, that our model successfully forgets the data that needs to be forgotten and exhibits the effect of never being trained on these data.

F. Computational Overhead

We supplemented the time-overhead experiments of the various algorithms. We calculate the time of single-class forgetting of ResNet18 on CIFAR-10. As shown in Table IX, our approach is similar to Amnesiac in terms of time overhead, although our time-overhead was not the least, but it is only about 20s slower than UNSIR. Compared to UNSIR,

it took only a small amount of time to achieve better results. Therefore, such a time overhead is acceptable.

TABLE X: ResNet18 uses different α on CIFAR-10.

Metrics	$\alpha = 0.5$	$\alpha = 1.0$	$\alpha = 1.5$	$\alpha = 2.0$	$\alpha = 2.5$	$\alpha = 3.0$
A_{D_r}	90.88	89.20	89.30	88.65	88.01	87.88
A_{D_f}	39.28	1.66	0.00	0.00	0.00	0.00

TABLE XI: ResNet18 uses different α on SVHN.

Metrics	$\alpha = 0.5$	$\alpha = 1.0$	$\alpha = 1.5$	$\alpha = 2.0$	$\alpha = 2.5$	$\alpha = 3.0$
A_{D_r}	94.91	94.78	94.16	94.05	93.15	92.92
A_{D_f}	9.79	0.78	0.00	0.00	0.00	0.00

TABLE XII: ResNet18 uses different λ_1 and λ_2 on CIFAR-10

Metrics	λ_1/λ_2	$\lambda_1 = 0.005$	$\lambda_1 = 0.01$	$\lambda_1 = 0.05$	$\lambda_1 = 0.1$	$\lambda_1 = 0.15$
A_{D_r}	$\lambda_2 = 0.01$	88.75	88.54	88.62	89.71	89.57
	$\lambda_2 = 0.05$	88.87	89.30	89.29	88.85	88.89
	$\lambda_2 = 0.10$	88.83	88.61	89.36	88.78	89.11
	$\lambda_2 = 0.50$	89.15	88.71	89.88	88.96	88.30
	$\lambda_2 = 1.00$	88.63	89.32	88.14	88.21	88.88
A_{D_f}	$\lambda_2 = 0.01$	0.51	1.02	0.44	0.22	0.00
	$\lambda_2 = 0.05$	0.00	0.00	0.00	0.87	0.71
	$\lambda_2 = 0.10$	0.61	0.00	0.00	0.00	0.86
	$\lambda_2 = 0.50$	0.00	0.00	0.00	0.00	0.00
	$\lambda_2 = 1.00$	0.00	0.00	1.23	0.00	0.00

TABLE XIII: ResNet18 uses different λ_1 and λ_2 on SVHN.

Metrics	λ_1/λ_2	$\lambda_1 = 0.005$	$\lambda_1 = 0.01$	$\lambda_1 = 0.05$	$\lambda_1 = 0.1$	$\lambda_1 = 0.15$
A_{D_r}	$\lambda_2 = 0.01$	93.78	93.16	93.90	94.19	93.70
	$\lambda_2 = 0.05$	94.06	94.16	93.51	93.20	93.29
	$\lambda_2 = 0.10$	93.32	94.03	93.80	93.10	93.76
	$\lambda_2 = 0.50$	93.71	93.99	93.40	94.22	93.40
	$\lambda_2 = 1.00$	93.69	93.20	93.29	93.38	94.03
A_{D_f}	$\lambda_2 = 0.01$	0.42	0.00	1.11	0.05	1.01
	$\lambda_2 = 0.05$	0.00	0.00	0.12	0.00	0.00
	$\lambda_2 = 0.10$	0.00	0.00	0.74	0.42	0.00
	$\lambda_2 = 0.50$	0.00	0.00	0.00	0.00	0.00
	$\lambda_2 = 1.00$	0.00	0.00	0.00	0.00	0.00

G. Hyperparameter Sensitivity Analysis

We include hyperparameter experiments in our study. Due to space constraints, we added analysis using ResNet18 on CIFAR-10 and SVHN datasets. In Tables 1 and 2, we use α with values of [0.5,1.0,1.5,2.0,2.5,3.0] for the experiments. The results indicate that SPS is not sensitive to hyperparameter variations. It is important to note that α setting too low, such as 0.5, suppresses a significant number of model parameters, leading to a decrease in the model's ability to forget. Additionally, we added experiments on λ_1 with values of [0.005,0.01,0.05,0.1,0.15] and λ_2 with values of [0.01,0.05,0.10,0.50,1.00] in the model inversion. Whether it is A_{D_r} or A_{D_f}, the fluctuation of accuracy does not exceed 2%. The findings demonstrate that UISPS is also not sensitive to hyperparameters during model inversion.

H. Detail of Membership Inference Attacks

We adopt the member inference attack framework used in [9]. We first obtain the predicted probability distributions of the unlearned model for both the training and test sets. Subsequently, we employ a logistic regression model for binary classification training and evaluate the member inference attack success rate with the prediction results of the unlearned model on the forgotten data.

I. Limitations and Future Work

The purpose of our method is to unlearn when the data of the class to be forgotten cannot be used. It can essentially preserve the model's performance on the retained class. However, there is still no good solution for sample-level unlearning under zero-glance conditions. Although our method performs excellently among the current methods supporting class-level forgetting, the forgetting of random data or a subset of a class is beyond the scope of this work. Future work may require the development of more general methods under zero-glance conditions that can meet the needs of sample-level and class-level unlearning.