Supplementary Material: Static and Sequential Malicious Attacks in the Context of Selective Forgetting

Chenxu Zhao*
Department of Computer Science
Iowa State University
cxzhao@iastate.edu

Wei Qian*
Department of Computer Science
Iowa State University
wqi@iastate.edu

Rex Ying
Department of Computer Science
Yale University
rex.ying@yale.edu

Mengdi Huai
Department of Computer Science
Iowa State University
mdhuai@iastate.edu

1 Broad Impact

In order to study the vulnerabilities of machine learning models to malicious data update requests during the *unlearning* process, we in this paper design the static and sequential selective forgetting attack frameworks, which are different from traditional evasion and data poisoning attacks. Our work shows that machine learning models are highly vulnerable to our proposed selective forgetting attacks, thereby introducing a novel form of security threat. We believe that studying malicious selective forgetting attacks (that happen during the *unlearning* process) has a great impact on the field of computer security and has wide-ranging implications for privacy protection, AI security, policy development, and user trust. Our research contributes to the ongoing enhancement of security measures, fostering innovation, and ensuring the integrity and resilience of the unlearning systems in different real-world applications.

2 Limitations and Future Work

We view our proposed selective forgetting attacks as an initial step towards understanding the vulnerabilities and robustness of machine learning models to malicious data update requests during the *unlearning* process. In the future, we will extend our research by applying our proposed selective forgetting attacks to a broader range of machine learning models, diverse selective forgetting methods, and larger datasets. Additionally, we want to emphasize that unscrupulous individuals may adopt our methods to undermine real-world applications, which inevitably raises new concerns about AI safety. Therefore, in the future, we will develop the robust unlearning systems to defend malicious selective forgetting attacks.

3 Experiments for Fairness-aware Selective Forgetting Attacks

Note that in the main manuscript, we provide a toy example to illustrate the impact of malicious data update requests on fairness using the COMPAS [16] dataset. We consider race (black/white) as the sensitive feature and randomly unlearn varying percentages of samples from the minority group (white). We measure the fairness gap using two well-known metrics, Demographic Parity [9] and

*The first two authors contribute equally to this work.

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

Table 1: Fairness gap with static forgetting attacks on COMPAS.

Evaluation metric	Percentage of unlearning samples		
	4%	12%	20%
Demographic Parity	0.28 ± 0.01	0.30 ± 0.01	0.31 ± 0.02
Equalized Odds	0.39 ± 0.02	0.42 ± 0.02	0.42 ± 0.03

Equalized Odds [12]. Demographic Parity measures equality in the rate of positive predictions, and the fairness gap is defined as follows

$$|Pr(\hat{y} = 1|S = 1) - Pr(\hat{y} = 1|S = 0)|, \tag{1}$$

where $\hat{y} = \{0, 1\}$ is a binary prediction, and $S \in \{0, 1\}$ is the binary sensitive feature. Equalized Odds requires that the protected feature and predicted outcome be conditionally independent given the true label, and the fairness gap is defined as

$$\max_{y \in \{0,1\}} |Pr(\hat{y} \neq y|S = 0, Y = y) - Pr(\hat{y} \neq y|S = 1, Y = y)|, \tag{2}$$

where Y is the true label. These two fairness metrics enable us to quantitatively evaluate the impact of fairness on the COMPAS dataset.

In the main manuscript, we degrade the fairness by randomly removing training samples from the minority group. Here, we explore the application of our proposed static selective forgetting attacks to target fairness. Similarly, we introduce indication parameters for training samples in the minority group, which indicate whether the samples should be removed based on the impact of a fairness loss. To define the fairness loss, [26] proposes a loss function for fair classification that includes a constraint involving the covariance between the sensitive features $\{z\}_{i=1}^{N}$ and the signed distance from feature vectors to the decision boundary $\{d_\theta(x_i)\}_{i=1}^{N}$, formalized as

$$Cov(z, d_\theta(x)) \approx \frac{1}{N} \sum_{i=1}^{N} (z_i - \bar{z}) d_\theta(x_i), \tag{3}$$

where $\bar{z}$ represents the mean value of the sensitive attribute. By leveraging this formulation, we can maximize the fairness loss using the selective forgetting attack strategy.

In Table 1, we unlearn different percentages of training samples from the COMPAS dataset. The unlearning samples are selected from the minority group according to the impact of the fairness loss. We train a logistic regression model on the COMPAS dataset with 100 epochs and a learning rate of 0.01. The initial Demographic Parity is at 0.22, and the initial Equalized Odds is at 0.34. As demonstrated in the table, the fairness gap widens as we unlearn different percentages of samples from the minority group. Notably, by optimizing the fairness loss and assigning importance scores, we achieve a fairness gap of 0.28 for Equalized Odds and 0.39 for Equalized Odds when unlearning 4% of the samples. This implies that our selective forgetting attacks can also be applied to desired attack goals such as fairness.

4 Algorithms for Static Selective Forgetting Attacks

Algorithm 1 for selective forgetting attacks in the second-order unlearning case. Note that in the main manuscript, we take the second-order unlearning strategy proposed in [25] as *an illustrative example* to introduce our proposed selective forgetting attacks. Algorithm 1 illustrates the procedure to solve the proposed optimization framework using the second-order [25] strategy described in the manuscript. In this algorithm, we adopt the f_6 function from [6] as our adversarial loss $\mathcal{L}_{adv}(\cdot; \theta^u(D_f))$ in Eqn. (1) of the main manuscript, and consider the setting where the adversary wants to generate malicious update requests to attack the targeted test samples $\{x_s\}_{s=1}^{S}$ and force the targeted test sample x_s to be assigned as the attack targeted label $\bar{y}_s$. Below, we give our adversarial loss in detail

$$\mathcal{L}_{adv}(\cdot; \theta^u(D_f)) = \max(\max_{c \neq \bar{y}_s} f^c_{D \setminus D_f}(x_s; \theta^u) - f^{\bar{y}_s}_{D \setminus D_f}(x_s; \theta^u), -\beta), \tag{4}$$

where x_s denotes the targeted test sample, $\bar{y}_s$ denotes the attack targeted label, $f^c_{D\setminus D_f}(\cdot;\theta^u)$ represents the logit output of the unlearned model θ^u, and $\max(\cdot,-\beta)$ corresponds the hinge loss. In the above, the adversarial loss enforces the actual prediction of the targeted test sample x_s that aligns with the attack targeted label $\bar{y}_s$. In Algorithm 1, we first initialize the indication parameters from random distributions. In each optimization step, we update the unlearned model with the indication parameters of the targeted samples, the gradients difference between the targeted samples and their unlearned versions, and the inverse Hessian matrix as defined in Eqn. (3) of the main manuscript. After that, we compute the adversarial loss for all the targeted test samples using Eqn. (4) and the gradients of the adversary loss with respect to the indication parameters. Finally, we perform one-step gradient descent using Adam optimizer [14] and further clip the updated indication parameters to be within the range of 0 to 1.

Algorithm 1 Selective forgetting attacks in the second-order unlearning case

Input: Well-trained model θ^*, training dataset D, target data $\{x_s\}_{s=1}^S$, attack targeted label $\bar{y}_s$, targeted training data points $Z=\{z_p\}_{p=1}^P$ and $\tilde{Z}=\{\tilde{z}_p\}_{p=1}^P$, optimization steps O
Output: $\Omega=\{\omega_p\}_{p=1}^P$

1: Randomly initialize $\Omega=\{\omega_p\}_{p=1}^P$
2: **for** $o=1,\ldots,O$ optimization steps **do**
3: Update the unlearned model θ^u with Ω, $\Delta(Z,\tilde{Z})$, and inverse Hessian matrix in Eqn. (3) of the main manuscript
4: Compute adversarial loss $\mathcal{L}=\sum_{s=1}^S \mathcal{L}_{adv}(x_s,\bar{y}_s;\theta^u)$
5: Compute $\nabla_\Omega \mathcal{L}$
6: Update Ω using Adam and project onto $[0,1]$ bound
7: **end for**

Selective forgetting attacks in SISA [4]. Here, we talk about our proposed selective forgetting attacks in SISA [4]. Note that in SISA, the original training dataset D is randomly partitioned into M disjoint shards (i.e., $\{D_m\}_{m=1}^M$). For the m-th shard, we can train a shard model θ^*_m by using D_m, where θ^*_m are the obtained model parameters. After that, the final prediction results are obtained from the aggregation of the M submodels (i.e., $\{\theta^*_m\}_{m=1}^M$). Upon receiving an unlearning request, the model holder only needs to retrain the corresponding shard model. Following the same notations in the main manuscript, we here use $D_t=\{(x_p,y_p)\}_{p=1}^P\subset D$ to denote the targeted training samples, which is accessible to the adversary. Based on the above, we change the constraint in Eqn. (2) of the main manuscript into

$$\theta^u_m=\arg\min_\theta \sum_{x_p\in D_m, x_p\in D_t}(1-\omega_p)*\ell(x_p,y_p;\theta)+\sum_{x_{p'}\in D_m, x_{p'}\notin D_t}\ell(x_{p'},y_{p'};\theta), \tag{5}$$

where $\omega_p\in\{0,1\}$, and ℓ is a training loss (e.g., cross-entropy). Note that $\omega_p=1$ is equivalent to an entire removal of x_p from the training set. That is to say, its loss influence is zero since $(1-\omega_p)=0$. However, it is very difficult to solve the above formulated optimization problem due to the introduced discrete indication parameters (i.e., $\{\omega_p\}_{p=1}^P$) of the constraint in Eqn. (5). To address this challenge, we propose to relax each discrete variable into a continuous one of range $[0,1]$, i.e., $\omega_p\in[0,1]$, and then rewrite the loss in Eqn. (5) as

$$\theta^u_m=\arg\min_\theta \sum_{x_p\in D_m, x_p\in D_t}\frac{1}{2}(1-\mathrm{sgn}(2*\omega_p-1))*\ell(x_i,y_i;\theta)+\sum_{x_{p'}\in D_m, x_{p'}\notin D_t}\ell(x_{p'},y_{p'};\theta), \tag{6}$$

where ω_p is relaxed into a continuous one of range $[0,1]$, i.e., $\omega_p\in[0,1]$. In the above, we rewrite $(1-\omega_p)$ as $\frac{1}{2}(1-\mathrm{sgn}(2*\omega_p-1))$. Note that if $\omega_p\geq 0.5$, we will wholly delete sample x_p, otherwise not. Based on the above, we can approximate the objective in Eqn. (6) by the following one

$$\begin{aligned}\theta^u_m=\arg\min_\theta \sum_{x_p\in D_m, x_p\in D_t}&\left(1-\frac{1}{1+\exp(-\varphi(2*\omega_p-1))}\right)\\ &*\ell(x_i,y_i;\theta)+\sum_{x_{p'}\in D_m, x_{p'}\notin D_t}\ell(x_{p'},y_{p'};\theta).\end{aligned} \tag{7}$$

Based on the fact that function $h_1(x) = \frac{1}{2}(1 - \text{sgn}(x))$ can be approximated by function $h_2(x) = 1 - \frac{1}{1+\exp(-\varphi x)}$, we can obtain the above equation. The parameter φ in h_1 represents the steepness of the curve. Additionally, the continuous property of h_1 allows us to solve the formulated optimization based on the above objective in Eqn. (7). In Algorithm 2, we give the procedure for solving the proposed static selective forgetting attack framework using SISA [4]. For simplicity, we use $\xi(D_m, D_t, \{\omega_p\}_{p=1}^P; \tilde{\theta})$ to denote the model update in Eqn. (7).

Algorithm 2 Selective forgetting attacks in the SISA unlearning case

Input: Training dataset D, target data $\{x_s\}_{s=1}^S$, attack targeted label $\bar{y}_s$, targeted training data points D_t, number of shards M, learning rate η, training epochs E, optimization steps O
Output: $\{\Omega_m\}_{m=1}^M$

for $m = 1, \ldots, \lceil M/2 \rceil + 1$ shards **do**
  Randomly initialize $\Omega_m = \{\omega_p\}_{p=1}^P$
  Initialize K surrogate models and pre-train θ_m^k for $\lfloor kE/K \rfloor$ epochs on shard data D_m
  for $o = 1, \ldots, O$ optimization steps **do**
    for $k = 1, \cdots, K$ models **do**
      Copy $\tilde{\theta} = \theta_m^k$
      Optimize $\tilde{\theta} = \tilde{\theta} - \eta \nabla_{\tilde{\theta}} \xi(D_m, D_t, \{\omega_p\}_{p=1}^P; \tilde{\theta})$
      Compute adversarial loss $\mathcal{L}_k = \sum_{s=1}^S \mathcal{L}_{adv}(x_s, \bar{y}_s; \tilde{\theta})$
    end for
    Average adversarial loss $\mathcal{L} = \sum_{k=1}^K \mathcal{L}_k / K$
    Compute $\nabla_{\Omega_m} \mathcal{L}$
    Update Ω_m using Adam and project onto $[0, 1]$ bound
  end for
end for

Selective forgetting attacks in the first-order unlearning case. Next, we introduce our proposed malicious selective forgetting attacks against machine learning models in the first-order unlearning case proposed in [25]. The first-order update strategy uses a first-order Taylor Series to change the original model's parameters to obtain the unlearned model [25]. For $D_t = Z = \{z_p\}_{p=1}^P \subset D$, we use $\tilde{D}_t = \tilde{Z} = \{\tilde{z}_p\}_{p=1}^P$ to denote its corresponding unlearned versions, where $\tilde{z}_p = (x_p - \xi_p, y_p)$ and ξ_p is the unlearning modification for x_p. Following [25], we calculate the change $\Delta(Z, \tilde{Z})$ by calculating all the gradients difference between Z and $\tilde{Z}$ as $\Delta(Z, \tilde{Z}) = (\sum_{\tilde{z}_p \in \tilde{Z}} \omega_p * \nabla_\theta \ell(\tilde{z}_p; \theta^*) - \sum_{z_p \in Z} \omega_p * \nabla_\theta \ell(z_p; \theta^*))$, where $\omega_p \in \{0, 1\}$ denotes whether z_p should be completely erased. Then, we can obtain the unlearned model as follows

$$\theta^u \leftarrow \theta^* - \tau[\sum_{p=1}^P \omega_p * (\nabla_\theta \ell(\tilde{z}_p; \theta^*) - \nabla_\theta \ell(z_p; \theta^*))], \tag{8}$$

where $\omega_p \in \{0, 1\}$ denotes whether z_p should be completely erased. Note that it is very difficult to optimize the effective update requests due to the introduced discrete indication parameters (i.e., $\Omega = \{\omega_p \in \{0, 1\}\}_{p=1}^P$) of the constraint in Eqn. (2) of the main manuscript. To address the aforementioned challenge, we propose to relax each discrete variable ω_p into a continuous one of range $[0, 1]$, i.e., $\omega_p \in [0, 1]$, and then approximate the original update objective for this first-order strategy by the following one

$$\theta^u \leftarrow \theta^* - \tau[\sum_{p=1}^P (\frac{1}{1 + \exp(-\varphi(2 * \omega_p - 1))}) * (\nabla_\theta \ell(\tilde{z}_p; \theta^*) - \nabla_\theta \ell(z_p; \theta^*))], \tag{9}$$

where ℓ is a training loss, τ is a small unlearning rate, and $\omega_p \in [0, 1]$. Here, we rewrite ω_p as $\frac{1}{2}(1 + \text{sgn}(2 * \omega_p - 1))$. Based on the fact that function $h_1(x) = \frac{1}{2}(1 - \text{sgn}(x))$ can be approximated by function $h_2(x) = 1 - \frac{1}{1+\exp(-\varphi x)}$, we can obtain the above equation. The parameter φ in h_2 represents the steepness of the curve. Additionally, the continuous property of h_2 allows us to solve the formulated optimization to perform selective forgetting attacks via the first-order unlearning strategy in [25]. Note that from Section IV in [25], we can know that if $\tilde{Z} = \emptyset$, the adversary has the

option to completely remove the targeted sample or retain it, based on whether $\omega_p \geq 0.5$ or $\omega_p < 0.5$. When $\tilde{Z} \neq \emptyset$, the adversary can intentionally and maliciously modify the targeted training samples via partially unlearning some data information based on Eqn. (9). Algorithm 3 sketches the procedure to solve the optimization of static selective forgetting attacks in the first-order case [25].

Algorithm 3 Selective forgetting attacks in the first-order unlearning case

Input: Well-trained model θ^*, training dataset D, target data $\{x_s\}_{s=1}^S$, attack targeted label $\bar{y}_s$, targeted training data points $Z = \{z_p\}_{p=1}^P$ and $\tilde{Z} = \{\tilde{z}_p\}_{p=1}^P$, unlearning rate τ, the number of optimization steps O
Output: $\Omega = \{\omega_p\}_{p=1}^P$

1: Randomly initialize $\Omega = \{\omega_p\}_{p=1}^P$
2: **for** $o = 1, \ldots, O$ optimization steps **do**
3: Update the unlearned model θ^u with Ω, $\Delta(Z, \tilde{Z})$, and τ in Eqn. (9)
4: Compute adversarial loss $\mathcal{L} = \sum_{s=1}^S \mathcal{L}_{adv}(x_s, \bar{y}_s; \theta^u)$
5: Compute $\nabla_\Omega \mathcal{L}$
6: Update Ω using Adam and project onto $[0, 1]$ bound
7: **end for**

5 Proof of Theorem 1

Definition 1 (Strongly convexity)**.** *A function $\psi : \mathbb{R}^{d_3} \to \mathbb{R}^{d_4}$ is said to be M-strongly convex for some $M \geq 0$ if for any $z_1 \in \mathbb{R}^{d_3}, z_2 \in \mathbb{R}^{d_3}$, and any $q \in (0, 1)$, $\psi(qz_1 + (1-q)z_2) \leq q\psi(z_1) + (1-q)\psi(z_1) - \frac{M}{2}q(1-q)||z_1 - z_2||_2^2$. Note that if the above condition is satisfied for $M = 0$, we refer to the function ψ as convex.*

Definition 2 (Lipschitzness continuity)**.** *A general function $\psi : \mathbb{R}^{d_3} \to \mathbb{R}^{d_4}$ is L-globally Lipschitz continuous if for all $z_1 \in \mathbb{R}^{d_3}, z_2 \in \mathbb{R}^{d_3}$, $||\psi(z_1) - \psi(z_2)|| \leq L||z_1 - z_2||_2$.*

Theorem 1. *Let $\theta_D^* = \arg\min_{\theta \in \Theta} \ell_D(\theta)$ for any given dataset D. Suppose that the loss function ℓ_z is L-globally Lipschitz continuous and M-strongly convex for any $z \in \mathcal{Z}$. For any integer N, dataset D of size N, and forget set D_f, we can have that $||\theta_D^* - \theta_{D \setminus D_f}^*||_2 \leq \frac{2L}{MN}|D_f|$, where $|D_f|$ represents the size of the forget set.*

Proof. First, recall the definition of M-strong convexity: for any $\theta_1, \theta_2 \in \Theta$, and any $q \in (0, 1)$,

$$\ell(q\theta_1 + (1-q)\theta_2) \leq q\ell(\theta_1) + (1-q)\ell(\theta_1) - \frac{M}{2}q(1-q)||\theta_1 - \theta_2||_2^2. \tag{10}$$

We use θ^* to denote the minimizer of ℓ. Now fix some $z \in \mathcal{Z}$. We have that for any $q \in (0, 1)$,

$$\ell(\theta^*) \leq \ell(q\theta + (1-q)\theta^*) \leq q\ell(\theta) + (1-q)\ell(\theta^*) - \frac{M}{2}q(1-q)||\theta - \theta^*||_2^2, \tag{11}$$

where the first inequality follows because θ^* is the minimizer of ℓ, and the second is due to M-strong convexity of ℓ. Based on the above equation, we can obtain the following

$$\begin{aligned} &[\ell(\theta^*) - (1-q)\ell(\theta^*) + \frac{M}{2}q(1-q)||\theta - \theta^*||_2^2] \leq q\ell(\theta), \\ \implies &\frac{M}{2}(1-q)||\theta - \theta^*||_2^2 \leq \ell(\theta) - \ell(\theta^*), \end{aligned} \tag{12}$$

where $q \in (0, 1)$. Then, we can have

$$\begin{aligned} &\sup_{q \in (0,1)} \frac{M}{2}(1-q)||\theta - \theta^*||_2^2 \leq \ell(\theta) - \ell(\theta^*), \\ \implies &\frac{M}{2}||\theta - \theta^*||_2^2 + \ell(\theta^*) \leq \ell(\theta), \end{aligned} \tag{13}$$

where θ^* is the minimizer of ℓ. For the dataset $D = \{z_i\}_{i=1}^N$ that is of size N, we use $D_f = \{\tilde{z}_i\}_{i=1}^{|D_f|}$ to denote the forget set, where $D_f \geq 1$. Next we discuss the case where $|D_f| = 1$. We can easily see

that if $\tilde{z}_1 \notin D$, then the equation immediately follows. Below, we discuss the case where $\tilde{z}_1 \in D$. In this case, given D and D_f, we can obtain the reminding dataset $D_r = D \setminus D_f$. Then we can derive the following

$$\ell_D(\theta^*_{D_r}) = \frac{N-1}{N}\ell_{D_r}(\theta^*_{D_r}) + \frac{1}{N}\ell_z(\theta^*_{D_r}) \leq \frac{N-1}{N}\ell_{D_r}(\theta^*_{D_r}) + \frac{1}{N}\ell_z(\theta^*_{D_r}), \tag{14}$$

where the above inequality is derived based on the optimality of $\theta^*_{D_r}$ for $D_r = D \setminus D_f$. Based on the above, we can obtain the following

$$\begin{aligned} &\frac{N-1}{N}\ell_{D_r}(\theta^*_{D_r}) + \frac{1}{N}\ell_z(\theta^*_{D_r}) \\ &= \ell_D(\theta^*_D) + \frac{1}{N}\ell_z(\theta^*_{D_r}) - \frac{1}{N}\ell_z(\theta^*_D) \\ &\leq \ell_D(\theta^*_D) + \frac{L}{N}||\theta^*_{D_r} - \theta^*_D||_2, \end{aligned} \tag{15}$$

where $D_r = D \setminus D_f$. In the above, the inequality follows by L-Lipschitzness of ℓ_z. Note that ℓ_D is M-strongly convex, based on the above, we can derive

$$\ell_D(\theta^*_{D_r}) \geq \ell_D(\theta^*_D) + \frac{M}{2}||\theta^*_D - \theta^*_{D_r}||_2^2. \tag{16}$$

Combining Eqn. (15) and (16), we can complete the proof for the case when we want to delete D_f (that is of size 1) from dataset D. Next, we discuss the case where the size of $D_f = \{\tilde{z}_i\}_{i=1}^{|D_f|}$ exceeds one (i.e., $D_f > 1$). Here we use $D_f^t = \{\tilde{z}_i\}_{i=1}^t$ to denote a subset of D_f. When $t = 1$, we in the above prove that $||\theta^*_D - \theta^*_{D\setminus D_f^t}||_2 \leq \frac{2L}{MN}$, where $D_f^t = \{\tilde{z}_i\}_{i=1}$. Then, we can obtain $||\theta^*_{D\setminus D_f^{t-1}} - \theta^*_{D\setminus D_f^t}||_2 \leq \frac{2L}{MN}$, where D_f^{t-1} and D_f^t are adjacent datasets that differ by only one sample. Based on the above, we have the following

$$||\theta^*_D - \theta^*_{D\setminus D_f}||_2 \leq \frac{2L}{MN}|D_f|, \tag{17}$$

where $|D_f|$ denotes the size of the forget set, and N is the size of the training dataset.

Therefore, we complete the proof. □

6 Discussions on the Transition Functions

Note that in Section 3.2 of the main manuscript, we consider the modification step where o_t corresponds to the action "*Modify*" to talk about how to calculate the state transition function. Below, we give the computation of the state transition function in other cases where we have the "*Add*" and "*Delete*" update requests.

First, we consider the "*Add*" case, where the update instruction o_t corresponds to the action "*Add*". In this case, the transition function can be derived as

$$\begin{aligned} &\mathcal{T}(s_{t+1}|s_t, a_t) = \mathcal{T}(u_{t+1}, \theta_{t+1}|u_t, \theta_t, a_t) \\ &= P(u_{t+1}|u_t, \theta_t, a_t)Pr(\theta_{t+1} = g(u_t, \theta_t, a_t)) \\ &= P(u_{t+1}|u_t, \theta_t, a_t)Pr(\theta_{t+1} = g'(\widetilde{u}_t, \theta_t)) = P(u_{t+1}|u_t, \theta_t, a_t)Pr(\theta_{t+1} = R_A(D_{t-1}, \\ &\theta_t, \widetilde{u}_t)) = P(u_{t+1}|u_t, \theta_t, a_t) = P(u_{t+1}). \end{aligned} \tag{18}$$

Note that given the attack action a_t, we can obtain $\widetilde{u}_t = (o_t, \emptyset, \tilde{z}_t^{new} = (z_t^{new} + a_t))$.

Then, we present the derivation of the transition function for the "*Delete*", where the update instruction o_t corresponds to the action "*Delete*" for the t-th time step. Here, we can calculate the transition function as follows

$$\begin{aligned} &\mathcal{T}(s_{t+1}|s_t, a_t) = \mathcal{T}(u_{t+1}, \theta_{t+1}|u_t, \theta_t, a_t) \\ &= P(u_{t+1}|u_t, \theta_t, a_t)Pr(\theta_{t+1} = g(u_t, \theta_t, a_t)) \\ &= P(u_{t+1}|u_t, \theta_t, a_t)Pr(\theta_{t+1} = g'(\widetilde{u}_t, \theta_t)) = P(u_{t+1}|u_t, \theta_t, a_t)Pr(\theta_{t+1} = R_A(D_{t-1}, \\ &\theta_t, \widetilde{u}_t)) = P(u_{t+1}|u_t, \theta_t, a_t) = P(u_{t+1}). \end{aligned} \tag{19}$$

Recall that given the attack action a_t, we can obtain $\widetilde{u}_t = (o_t, z_t^{tra}, \tilde{z}_t^{new} = (z_t^{tra} + a_t))$.

7 Algorithm for Sequential Selective Forgetting Attacks

Algorithm 4 Training adversarial policy

Input: Update sequence U, total attack times N, time step T, training episode E
Output: Adversarial policy Φ^{adv}

Initialize network parameters for policy Φ^{adv}
for $e = 0, \cdots, E$ episode **do**
  $attack\_times = 0$
  for $t = 0, \ldots, T$ time steps **do**
    $u_t = U_t = (o_t, z_t^{tra}, z_t^{new})$
    $(p_t, a'_t) = \Phi^{adv}(s_t)$
    if $p_t \geq 0.5$ and $attack\_times < N$ **then**
      if $o_t ==$ *"Add"* **then**
        $\tilde{u}_t = (o_t, z_t^{tra} = \emptyset, \tilde{z}_t^{new} = z_t^{new} + a'_t)$
      else if $o_t ==$ *"Modify"* **then**
        $\tilde{u}_t = (o_t, z_t^{tra}, \tilde{z}_t^{new} = z_t^{new} + a'_t)$
      else
        $\tilde{u}_t = (\tilde{o}_t =$ *"Modify"*$, z_t^{tra}, \tilde{z}_t^{new} = z_t^{tra} + a'_t)$
      end if
      $attack\_times = attack\_times + 1$
      Perform $\tilde{u}_t$ and receive reward r_t and s_{t+1}
    else
      Perform u_t and receive reward r_t and s_{t+1}
    end if
  end for
  Train policy Φ^{adv} and update network parameters
end for

Algorithm 4 describes the training procedure for the adversary policy. The adversary first initializes the network parameters of this policy sampled from random distributions. During each step of an episode, the adversary obtains an attack strategy (p_t, a'_t) from the adversarial policy. If $p_t \geq 0.5$, the adversary designates step t as the critical point and introduces perturbations to mislead the model owner into triggering action a'_t and modifying the corresponding attacked update request. Otherwise, the adversary does not attack the current time step and proceeds with the original update request. The adversary then receives a reward and obtains the next state information from the environment. Subsequently, these data are used to train the adversarial policy (e.g., using deep deterministic policy gradient (DDPG) [18]) and update the network parameters.

8 Proof of Theorem 2

Theorem 2. *Let $\Phi^*_{\mathcal{M}}$ and $\Phi^*_{\hat{\mathcal{M}}}$ be the optimal policies for $\mathcal{M}$ and $\hat{\mathcal{M}}$ respectively, with the same initial state distribution μ_0. We assume that the 1-Wasserstein distance between the estimated distributions $\widehat{P}$ and the true distribution P satisfies $W_1\left(\widehat{P}, P\right) \leq \upsilon$. Additionally, we assume the loss function $\ell : S \times \mathcal{Z} \to \mathbb{R}$ exhibits Lipschitz continuity with respect to both s and z, with a constant L_1, and Lipschitz smoothness with respect to z, with a constant L_2. Moreover, we assume the loss function to possess strong convexity, strong smoothness, and twice continuous differentiability with respect to s. Let $\mathcal{J}_{\mathcal{M}}(\Phi_{\mathcal{M}})$ be the cost in Eqn. 5. Then there exist two constants Ψ and Ω such that:*

$$\left|\mathcal{J}_{\mathcal{M}}\left(\Phi^*_{\mathcal{M}}\right) - \mathcal{J}_{\mathcal{M}}\left(\Phi^*_{\hat{\mathcal{M}}}\right)\right| \leq \upsilon\Omega(L_1(L_2\zeta + 2) + \Psi L_2\zeta), \tag{20}$$

where ζ is a constant related with updating the model.

Proof. We use $W_1\left(\widehat{P}, P\right)$ to denote the 1-Wassertein distance to measure the distance between the estimated distributions $\widehat{P}$ and the true distribution P. Then the transition distributions are $\widehat{T}$ and T respectively. Let $\mathcal{O}_l$ be the expected return when $\Phi^*_{\mathcal{M}}$ is applied to $\widehat{\mathcal{M}}$ for the first l steps, then

switched to $\mathcal{M}$ for l to $H-1$ [17]. That is,

$$\mathcal{O}_l = \mathop{\mathbb{E}}_{\substack{a^t \sim \Phi^*_{\mathcal{M}}(s^t) \\ t<l: s^{t+1} \sim \widehat{T}(s^t, a^t)}} \left[\sum_{t=0}^{H-1} R\left(s^t, a^t\right) \right] \tag{21}$$

$$t \geq l : s^{t+1} \sim T\left(s^t, a^t\right). \tag{22}$$

Based on the above definition of $\mathcal{O}_l$, we have that $\mathcal{J}_M\left(\Phi^*_{\mathcal{M}}\right) = \mathcal{O}_0$ and $\mathcal{J}_{\widehat{\mathcal{M}}}\left(\Phi^*_{\mathcal{M}}\right) = \mathcal{O}_H$, which implies that $\mathcal{J}_{\mathcal{M}}\left(\Phi^*_{\mathcal{M}}\right) - \mathcal{J}_{\widehat{\mathcal{M}}}\left(\Phi^*_{\mathcal{M}}\right) = \sum_{l=0}^{H-1}\left(\mathcal{O}_l - \mathcal{O}_{l+1}\right)$. In the above, without loss of generality, we first assume the time step is H, and half of them are non-attack steps. Moreover, we adopt the attack-pair setting, wherein two update requests are received simultaneously during each time step. In this setting, the attack is conducted on the second request ($k = 2$), while the first request ($k = 1$) is processed in the non-attack mode. Note that our result here can be easily generalized to other settings. We denote the optimal value function for $\Phi^*_{\mathcal{M}}$ and $\Phi^*_{\hat{\mathcal{M}}}$ at time l as $V^*_{\mathcal{M},l}(s)$ and $V^*_{\hat{\mathcal{M}},l}(s)$. We then assume $V^*_{\mathcal{M},l}(s)$ and $V^*_{\hat{\mathcal{M}},l}(s)$ are Lipschitz continuous with Ψ. Additionally, we assume the loss function $\ell : S \times \mathcal{Z} \to \mathbb{R}$ exhibits Lipschitz continuity with respect to both s and z, with a constant L_1 and Lipschitz smoothness with respect to z, with a constant L_2. We also assume the loss function to possess strong convexity, strong smoothness, and twice continuous differentiability with respect to s. We define $\ell_k(s) := \mathbb{E}_{z_k \sim P}\left[\ell\left(s, z_k\right)\right], \ell'_k(s) := \mathbb{E}_{z_k \sim \widehat{P}}\left[\ell\left(s, z_k\right)\right]$, and $G^*_{\widehat{\mathcal{M}},l}\left(s^l, a^l\right) := \mathbb{E}_{s^{l+1} \sim T(s^l, a^l)}\left[V^*_{\mathcal{M},l}\left(s^{l+1}\right)\right] - \mathbb{E}_{s^{l+1} \sim \widehat{T}(s^l, a^l)}\left[V^*_{\mathcal{M},l}\left(s^{l+1}\right)\right]$. Based on the above, we can derive the following

$$\mathcal{J}_{\mathcal{M}}\left(\Phi^*_{\mathcal{M}}\right) - \mathcal{J}_{\widehat{\mathcal{M}}}\left(\Phi^*_{\mathcal{M}}\right) = \sum_{l=0}^{H-1}\left(\mathcal{O}_l - \mathcal{O}_{l+1}\right) \tag{23}$$

$$= \sum_{l=0}^{H-1} \mathbb{E}_{s^l, a^l \sim \widehat{T}, \Phi^*_{\mathcal{M}}} \left(\mathbb{E}_{s^{l+1} \sim T(s^l, a^l)} \left[\frac{1}{2} \sum_{k=1}^{2} \left(\ell_k\left(s^{l+1}\right) - \ell_k\left(s^l\right) \right) \right] \right. \tag{24}$$

$$\left. - \mathbb{E}_{s^{l+1} \sim \widehat{T}(s^l, a^l)} \left[\frac{1}{2} \sum_{k=1}^{2} \ell'_k\left(s^{l+1}\right) - \ell'_k\left(s^l\right) \right) \right] \right) + \sum_{l=0}^{H-1} \mathbb{E}_{s^l, a^l \sim \widehat{T}, \Phi^*_{\mathcal{M}}} \left[G^*_{\widehat{\mathcal{M}},l}\left(s^l, a^l\right) \right] \tag{25}$$

$$\leq H\Psi W_1\left(T(\cdot \mid s, a), \widehat{T}(\cdot \mid s, a)\right) + \frac{H}{2} \sum_{k=1}^{2} \left| \mathbb{E}_{z_k \sim \widehat{P}} \ell_k\left(s, z_k\right) - \mathbb{E}_{z_k \sim P} \ell_k\left(s, z_k\right) \right| \tag{26}$$

$$+ \frac{H}{2} \sum_{k=1}^{2} \left| \mathbb{E}_{s' \sim T(s,a), z_k \sim P}\left[\ell_k\left(s', z_k\right)\right] - \mathbb{E}_{s' \sim \widehat{T}(s,a), z_k \sim \widehat{P}}\left[\ell_k\left(s', z_k\right)\right] \right| \tag{27}$$

$$\leq H\Psi W_1\left(T(\cdot \mid s, a), \widehat{T}(\cdot \mid s, a)\right) + \frac{H}{2} L_1 \upsilon + H L_1 W_1\left(T(\cdot \mid s, a), \widehat{T}(\cdot \mid s, a)\right) + \frac{H}{2} L_1 \upsilon \tag{28}$$

$$= H\left(\Psi + L_1\right) W_1\left(T(\cdot \mid s, a), \widehat{T}(\cdot \mid s, a)\right) + H L_1 \upsilon, \tag{29}$$

where L_1 is the Lipschitz continuity constant of the loss function ℓ. Similarly, we can get the bound for $\mathcal{J}_{\widehat{\mathcal{M}}}\left(\Phi^*_{\hat{\mathcal{M}}}\right) - \mathcal{J}_{\mathcal{M}}\left(\Phi^*_{\hat{\mathcal{M}}}\right)$. Thus, we have the following

$$\left|\mathcal{J}_{\mathcal{M}}\left(\Phi^*_{\mathcal{M}}\right) - \mathcal{J}_{\mathcal{M}}\left(\Phi^*_{\hat{\mathcal{M}}}\right)\right| \leq H\left(\Psi + L_1\right) W_1\left(T(\cdot \mid s, a), \widehat{T}(\cdot \mid s, a)\right) + H L_1 \upsilon. \tag{30}$$

For the 1-Wassertein distance between the estimated distributions $\widehat{P}$ and the true distribution P, we consider that $W_1\left(\widehat{P}, P\right)$ belongs to the interval $[0, \upsilon]$. Denoting the update rate as ζ, we can then derive an upper bound for the 1-Wasserstein distance between T and $\widehat{T}$ [3, 17]. Specifically, the 1-Wasserstein distance between $T(\cdot \mid s, a)$ and $\widehat{T}(\cdot \mid s, a)$ generated from the true and the estimated environments, respectively, is bounded by $\zeta L_2 \upsilon$, that is, $W_1\left(T(\cdot \mid s, a), \widehat{T}(\cdot \mid s, a)\right) \leq \frac{1}{2}\zeta L_2 \upsilon$.

Finally, by combining all the above, we have

$$\left|\mathcal{J}_{\mathcal{M}}\left(\Phi^*_{\mathcal{M}}\right) - \mathcal{J}_{\mathcal{M}}\left(\Phi^*_{\hat{\mathcal{M}}}\right)\right| \leq \upsilon\Omega(L_1(L_2\zeta + 2) + \Psi L_2\zeta). \tag{31}$$

In the above, L_2 denotes the Lipschitz smoothness constant of the loss function ℓ with respect to z, L_1 is the Lipschitz continuity constant of the loss function ℓ and $\Omega = \frac{1}{2}H$. □

9 More Experimental Results

In this section, we first delve into further experimental details, providing a deeper understanding of our research methodology. Then, we present additional experimental results for our proposed static selective forgetting attacks. Next, we showcase additional experimental results pertaining to our proposed sequential selective forgetting attacks.

Machine configuration. The experiments are implemented using the PyTorch [2] and run on a GPU machine with Intel Xeon Gold 5218 2.30GHz CPUs and NVIDIA Quadro RTX 8000 GPUs.

Experimental setup and parameter settings for static forgetting attacks. In our experiments, we evaluate the impact of static forgetting attacks on different datasets and models. Specifically, we consider ResNet-18 [13], VGG-16 [22], and MobileNetV2 [21] on the CIFAR-10 [15] dataset and a neural network with two fully connected layers on the Diabetes [1] dataset. Prior to the attacks, we pre-train the models for 20 epochs with a learning rate of 0.01 and a batch size of 128 and then perform selective forgetting attacks. The cross-entropy loss is used to optimize the models. In the proposed optimization framework, we set the optimization steps to 30 and the steepness of the curve to 100. We update the indication parameters of the training samples using the Adam optimizer [14] with a learning rate of 0.01. For the first-order based [25] unlearning method, we initialize the unlearning rate to 0.04; for the unrolling SGD [23] unlearning method, we initialize the fine-tuning epoch to 1 and the learning rate to 0.01; for the amnesiac [11] unlearning method, we initialize the fine-tuning epoch to 1 and the learning rate to 0.02; for SISA [4], we divide the training dataset into 5 disjoint shards, and we attack all the shards. Each shard is trained (retrained) for 20 epochs. We use different random seeds for each experiment to sample the target class, the adversarial class, and the test samples in the target class. We repeat the experiments 10 times and report the means and standard errors of the results.

Experimental setup and parameter settings for sequential forgetting attacks. In experiments, we consider a logistic regression model on the MNIST [7] dataset (digits 1 and 7), the Adult [8] dataset, and the synthetic dataset. We pre-train the models for 200 epochs with a learning rate of 0.01 and a batch size of 128. The target model is generated from static forgetting attacks in the untargeted setting using the first-order based unlearning method with an unlearning rate of 0.008. In terms of policy learning, we create a simulated environment using OpenAI Gym [5] and implement the deep deterministic policy gradient (DDPG) algorithm using Stable Baseline3 [20]. For our experiments, we set the length of pre-attack data (time step) to 300 and randomly sample update requests for *add*, *delete*, and *update*. The DDPG algorithm is trained for 300 episodes, with a policy learning rate of 0.001, a batch size of 100, a discount factor of 0.995, and a normal action noise.

9.1 More Experimental Results for Static Selective Forgetting Attacks

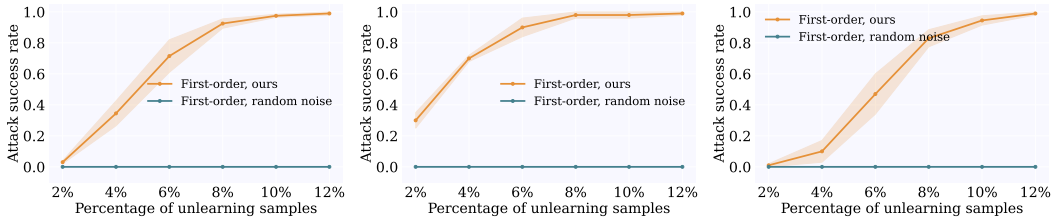
Attack performance in the untargeted setting. Table 2 summarizes the attack success rate of malicious selective forgetting attacks in the untargeted setting, where the adversary's goal is to manipulate the model into predicting any incorrect class. To calculate the adversarial loss in the untargeted setting, we modify the loss function in Eqn. (4) to $\max(f^{y_s}_{D\setminus D_f}(x_s;\theta^u) - \max_{c\neq y_s} f^{c}_{D\setminus D_f}(x_s;\theta^u), -\beta)$, where y_s denotes the true label and $f^{c}_{D\setminus D_f}(\cdot;\theta^u)$ represents the logit output of the unlearned model θ^u. This loss function ensures that the targeted test sample x_s is predicted as a label different from its true label y_s. In this experiment, we adopt first-order, second-order, unrolling SGD, amnesiac, and SISA unlearning methods. The experimental results demonstrate that our proposed methods achieve high attack success rates across various datasets and unlearning procedures. For instance, our proposed methods achieve attack success rates of 1.0 on both the Diabetes dataset and the CIFAR-10 dataset using first-order based, second-order based, unrolling SGD, and amnesiac unlearning methods. Our proposed methods can assign importance scores to training samples based on the impact of the untargeted loss and unlearn optimal samples, resulting in significant attack performance. In contrast, the RandSearch baseline shows negligible effort to misclassify targeted test samples in an untargeted manner. These experimental results also highlight the applicability and effectiveness of our optimization framework to malicious selective forgetting attacks via various unlearning methods to achieve untargeted attack goals.

Attack performance in the partial removal case against adversarially robust models. Here, we investigate the attack performance of our proposed malicious selective forgetting attacks in the

Table 2: Attack success rate of our proposed static forgetting attacks in the untargeted setting.

Dataset	Unlearning method	RandSearch	**Ours**
Diabetes	First-order	0.06 ± 0.03	$\mathbf{1.00 \pm 0.00}$
	Second-order	0.06 ± 0.03	$\mathbf{1.00 \pm 0.00}$
	Unrolling SGD	0.06 ± 0.06	$\mathbf{1.00 \pm 0.00}$
	Amnesiac	0.04 ± 0.03	$\mathbf{1.00 \pm 0.00}$
	SISA	0.33 ± 0.11	$\mathbf{0.87 \pm 0.05}$
CIFAR-10	First-order	0.28 ± 0.08	$\mathbf{1.00 \pm 0.00}$
	Second-order	0.32 ± 0.10	$\mathbf{1.00 \pm 0.00}$
	Unrolling SGD	0.22 ± 0.08	$\mathbf{1.00 \pm 0.00}$
	Amnesiac	0.12 ± 0.04	$\mathbf{1.00 \pm 0.00}$
	SISA	0.33 ± 0.10	$\mathbf{0.75 \pm 0.08}$

partial removal case, where the adversary aims to partially forget some data information on a batch of targeted training samples and mislead the model into making wrong predictions on targeted test samples. Specifically, we conduct experiments to verify the effectiveness of our proposed selective forgetting attacks against adversarially robust models, which are trained in an adversarial way to improve the models' ability to resist being attacked. In experiments, we consider three widely adopted adversarial training methods, FGSM (Fast Gradient Sign Method) [10], PGD (Projected Gradient Descent) [19], and TRADES adversarial training [27]. We train the adversarially robust models against L_∞ norm-bounded perturbations of size $8/255$. We perform selective forgetting attacks via the first-order unlearning method with a modification bound of $8/255$. As a baseline comparison, we introduce uniform random noise within the same modification bound. In Figure 1, we present the experimental results obtained on the MNIST dataset using the adversarially trained robust models. The figure demonstrates that our proposed malicious selective forgetting attacks yield significantly higher attack success rates compared to the baseline. For instance, when unlearning training samples of 8%, our malicious selective forgetting attacks achieve an attack success rate of approximately 0.92 on the adversarially robust model with PGD adversarial training, whereas the baseline has an attack success rate of 0. Similarly, we conduct experiments on the CIFAR-10 dataset, and the results are illustrated in Figure 2. These results further demonstrate the high attack success rates achieved by our selective forgetting attacks. From these experimental outcomes, we can conclude that our proposed malicious selective forgetting attacks exhibit strong attacking performance in the partial removal case, even against adversarially robust models.

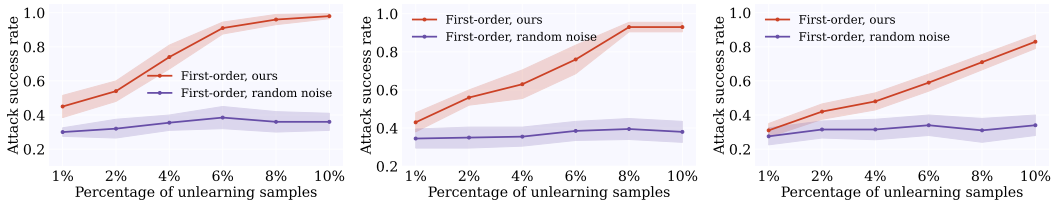


(a) PGD adversarial training (b) FGSM adversarial training (c) TRADES adversarial training

Figure 1: Attack success rate of our proposed static forgetting attacks on the adversarially robust models (which are trained using different adversarial training methods) on MNIST.

9.2 More Experimental Results for Sequential Selective Forgetting Attacks

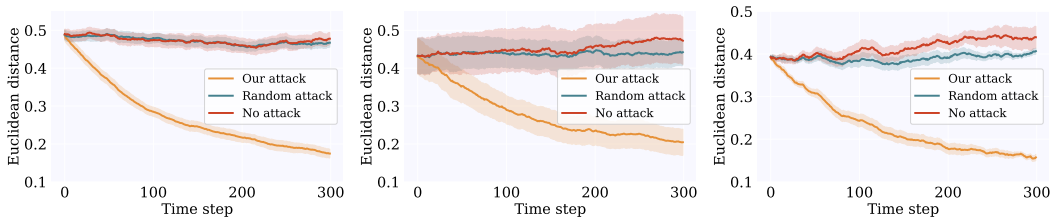
Black-box experiments. In the black-box setting, we propose training a local substitute model using a synthetic dataset. The adversary generates inputs from the synthetic dataset and observes the corresponding outputs from the target black-box model [24]. By leveraging this information, the adversary can gain insights into the vulnerabilities and decision boundary of the target black-box model. The substitute model is then trained using pre-attack data to learn an optimal policy, which is subsequently employed to attack the update requests on the target black-box model. In this experiment, we adopt the MNIST (digits 1 and 7), Adult, and synthetic datasets. For each dataset,



(a) PGD adversarial training (b) FGSM adversarial training (c) TRADES adversarial training

Figure 2: Attack success rate of our proposed static forgetting attacks on the adversarially robust models (which are trained using different adversarial training methods) on CIFAR-10.

we train a substitute logistic regression model by creating a synthetic dataset and conducting 5000 queries on the black-box model. The resulting substitute model achieves a high equivalence of 99% with the black-box model. Then, we train the optimal policy and employ it to attack the sequential update requests, following a similar approach to the white-box setting. The experimental results are shown in Figure 3. As we can see, the learned optimal policy from a substitute model also reduces the Euclidean distance between the victim model and the target model over time steps on each adopted dataset. In contrast, both the no attack baseline and the random attack baseline fail to approach the target model. Therefore, our proposed malicious selective forgetting attacks are also applicable in a sequential manner in the black-box setting and demonstrate efficient attack performance.



(a) MNIST (b) Adult (c) Synthetic

Figure 3: Euclidean distance of the victim model to the target model in the black-box setting.

References

[1] Behavioral risk factor surveillance system survey data, 2015.

[2] Pytorch: An imperative style, high-performance deep learning library, 2019.

[3] Kavosh Asadi, Dipendra Misra, and Michael Littman. Lipschitz continuity in model-based reinforcement learning. In *International Conference on Machine Learning*, pages 264–273. PMLR, 2018.

[4] Lucas Bourtoule, Varun Chandrasekaran, Christopher Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *Proceedings of the 42nd IEEE Symposium on Security and Privacy*, May 2021.

[5] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. OpenAI Technical Report, 2016.

[6] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57. IEEE Computer Society, May 2017.

[7] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[8] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[9] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226, 2012.

[10] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

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

[12] Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29, 2016.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[14] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[15] Alex Krizhevsky and Geoff Hinton. Convolutional deep belief networks on cifar-10. *Unpublished manuscript*, 40(7):1–9, 2010.

[16] Jeff Larson, Surya Mattu, Lauren Kirchner, and Julia Angwin. Compas dataset. 2017.

[17] Henger Li, Xiaolin Sun, and Zizhan Zheng. Learning to attack federated learning: A model-based reinforcement learning attack framework. In *Advances in Neural Information Processing Systems*, 2022.

[18] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *International Conference on Learning Representations*, 2016.

[19] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

[20] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.

[21] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[22] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[23] Anvith Thudi, Gabriel Deza, Varun Chandrasekaran, and Nicolas Papernot. Unrolling sgd: Understanding factors influencing machine unlearning. In *2022 IEEE 7th European Symposium on Security and Privacy (EuroS&P)*, pages 303–319. IEEE, 2022.

[24] Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. Stealing machine learning models via prediction apis. In *USENIX security symposium*, volume 16, pages 601–618, 2016.

[25] Alexander Warnecke, Lukas Pirch, Christian Wressnegger, and Konrad Rieck. Machine unlearning of features and labels. *Network and Distributed System Security Symposium*, 2023.

[26] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rogriguez, and Krishna P Gummadi. Fairness constraints: Mechanisms for fair classification. In *Artificial intelligence and statistics*, pages 962–970. PMLR, 2017.

[27] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P. Xing, Laurent El Ghaoui, and Michael I. Jordan. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*, 2019.