A Supplementary Materials

A.1 Related Work

Randomized Smoothing for Certified Robustness. Trustworthy machine learning has attracted active research in recent years [99, 158, 150, 135, 159, 151, 101, 151, 149, 146, 80, 160, 61, 60, 117, 153, 118, 3, 155, 152, 157, 156, 154]. Certificate robustness techniques provide guarantees that an object with certifiable robustness is robust against any admissible attacks [130, 41, 19, 139, 57, 106, 103, 59, 93, 109, 73, 89, 36]. Existing certificate robustness approaches include satisfiability modulo theories-based methods [63, 55, 29, 11] and mixed integer-linear programming-based models [18, 84, 25, 34, 9]. The common characteristics of these methods is that the ever-increasing complexity of deep neural networks has made it difficult to scale these methods meaningfully to high-dimensional datasets like ImageNet. In addition, they are often applicable to certain specific neural networks. Convex relaxation-based algorithms [131, 132, 100, 105, 114, 115, 19, 116, 148], linear relaxation-based models [129, 127], abstract interpretation-based methods [40, 90, 113], interval-bound propagation-based algorithms [26, 47, 54], and Lipschitz constant-constrained approaches [21, 129, 27, 147, 128, 123, 24, 2, 22, 30, 77, 65] rely on over approximation loose too much precision and provide loose guarantees on the worst-case margin, even for networks trained to be amenable to certification. The robust training built upon the loose guarantees limits the improvement of model robustness.

In order to deal with the scalability and loose guarantee issues in certified robustness, a line of work has been introduced based on randomized smoothing [69, 23, 70, 76, 104, 72, 144, 58, 6, 38, 141, 92, 88, 33, 13, 142, 119, 1, 16, 53, 86]. Randomized smoothing-based methods have demonstrated exceptional scalability and applicability when defending against adversarial examples by relaxing exact certificates to high-confidence probabilistic ones [31, 58, 28, 51, 102, 72, 91, 32, 145]. We have witnessed many promising randomized smoothing-based methods for certifying l_0 [70, 72], l_1 [69, 76, 120], and l_2 robustness [75, 23, 104, 144, 74]. Several recent research efforts have studied and identified the inapplicability of randomized smoothing to high-dimensional problems for l_p robustness against high-norm attacks when $p > 2$, especially l_∞ robustness [68, 140, 67, 5, 137].

Machine Unlearning. Machine unlearning, also known as selective forgetting [10, 43, 111] or data removal/deletion [42, 49] in machine learning, aims to eliminate the effect of a subset of training data on the already trained model [39, 50, 134, 97]. Existing techniques on machine unlearning can be broadly classified into two categories below.

(1) Exact machine unlearning algorithms aim to learn an unlearning model with the same performance as the ones obtained with retraining from scratch by completely excluding the forgotten data from the training data. Many of existing methods mainly focus on producing the exact unlearning for simple models or under some specific conditions [107], like leave-one-out cross-validation for SVMs (Support Vector Machines) [12, 62, 17], efficient machine unlearning for linear models [78, 85], data removal-enabled random forests [8, 108], provably efficient data deletion in K-means clustering [42], and fast data deletion for Naïve Bayes based on statistical query learning which assumes the training data is in a decided order [10]. These unlearning models often fail to work on large-scale datasets with complex models. Several research efforts have tried to improve the efficiency of machine unlearning. Linear filtration is a novel algorithm for the sanitization of deep classification models that predict logits, after class-wide deletion requests [4]. SISA is a practical approach for unlearning that relies on data sharding and slicing to reduce the computational overhead of unlearning [7]. It is designed to achieve the largest improvements for stateful algorithms like stochastic gradient descent for deep neural networks. Ullah et al. proposed an efficient unlearning algorithm based on constructing a maximal coupling of Markov chains for the noisy SGD procedure [124]. GraphEraser is a machine unlearning method tailored to graph data with two novel graph partition algorithms and a learning-based aggregation method [15]. RecEraser is a general and efficient machine unlearning framework tailored to recommendation tasks with new data partition methods and an adaptive aggregation method to improve the global model utility. These designs make our RecEraser more suitable for recommendation tasks [14].

(2) Approximate machine unlearning methods try to bring the parameters of the trained model closer to naive ones retrained from scratch through the relaxation of the exact unlearning definitions and requirements [4, 48, 45, 122, 79, 87]. Most of these approaches make use of or reconstruct historical gradients and model weights to quickly eliminate the influence of samples that are requested to be deleted, such that the unlearned model cannot be distinguished from the model that was never

trained on the removed data [136, 56, 94, 64]. O-k-means and DC-k-means first introduced the definition of approximate unlearning based on the distance (or divergence) between distributions of the retrained model and the unlearnt one [42]. Subsequent works follow similar approximate definitions to provide certified unlearning guarantees for strongly-convex learning problems [49, 94, 110]. Several approaches proposed different Newton's methods to approximate retraining for convex models, e.g., linear regression, logistic regression, and the last fully connected layer of a neural network [43, 44, 49]. Some recent works have also studied machine unlearning in optimization-based Bayesian inference [96, 46] and sampling-based Bayesian inference [37]. Recent methods designed effective machine unlearning strategies in the setting of federated learning [79, 81, 83, 126, 133]. Four recent efforts have studied the problem of joint optimization of security or privacy and machine unlearning [45, 15, 143, 82].

certified removal is a certified-removal mechanism that applies a Newton step on the model parameters that largely remove the influence of the deleted data points [49]. GKT is a zero-shot machine unlearning algorithm that imposes the constraint that zero training data is available to the unlearning algorithms [20]. Two recent studies propose online machine unlearning methods for linear regression models [78] and linear support vector machine models [17], for further improving the efficiency of machine unlearning. The former adapts users' requests to delete their data before a specific time bar. The latter conducts only the task of variable support vector machine.

To our best knowledge, the common characteristics of the above machine unlearning methods is that they consist of two sequential operations: (1) Training: train a model on the complete training data and (2) Unlearning: generate an unlearning model from the former. The combination of two operations is computationally expensive when training complex models over large datasets. In addition, they often sequentially redo the unlearning operations one by one, when addressing a series of machine unlearning requests. This work is the first to execute one-time operation of simultaneous training and unlearning in advance for a series of machine unlearning requests, as long as the actual data removals are below the certified budget of data removals, by leveraging the theory of randomized smoothing and gradient quantization.

A.2 Selection of Quantization Threshold

In Section 3, we have introduced the gradient quantization for the randomized smoothing for the certified data removals. Let $\lambda \geq 0$ be the quantization threshold.

$$Q(t) = Softmax([-|t-\lambda|, -|t|, -|t+\lambda|]) \tag{30}$$

where $Q(t)$ maps a gradient dimension t to a three-dimensional vector $[-|t-\lambda|, -|t|, -|t+\lambda|]$, where each component denotes the similarity score between t and $-\lambda$, 0, or λ. Therefore, all T gradient dimensions are partitioned into three intervals: $(-\infty, -\lambda/2]$ that comes near to $-\lambda$, $[-\lambda/2, \lambda/2]$ that is closer to 0, and $[\lambda/2, \infty)$ that approaches λ. Each component in $Q(t)$ with the Softmax function also represents the probability of the gradient dimension t belonging to classes -1, 0, or 1. Namely, the gradient quantization function $Q(t)$ will assign any gradient dimensions t below $-\lambda/2$ to class -1, those above $\lambda/2$ to class 1, and the remaining dimensions to 0. The most probable class $c_A \in \{-1, 0, 1\}$ in $Q(t)$ is assigned to dimension t as a final quantized gradient dimension.

In fact, the quantization threshold $\lambda \geq 0$ serves as a tradeoff hyperparameter to well balance the training performance and the certifiable radius regarding the data removals in our PCMU methods. On one hand, a large λ, say ∞, will cause most of the gradient dimensions to be assigned to class 0 and to be updated with zero. This may result in the failure of gradient and parameter updates and fail to train the model until convergence.

$$\int_{\frac{\lambda}{2}}^{\infty} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{z^2}{2\sigma^2}}\, dz = \int_{-\infty}^{-\frac{\lambda}{2}} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{z^2}{2\sigma^2}}\, dz << \int_{-\frac{\lambda}{2}}^{\frac{\lambda}{2}} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{z^2}{2\sigma^2}}\, dz \tag{31}$$

On the other hand, a small λ, say 0, will make $\underline{p_A} = \overline{p_B}$, which bring small certified radius R and thus result in the collapse of certified machine unlearning.

$$\int_{\frac{\lambda}{2}}^{\infty} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{z^2}{2\sigma^2}}\, dz = \int_{-\infty}^{-\frac{\lambda}{2}} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{z^2}{2\sigma^2}}\, dz >> \int_{-\frac{\lambda}{2}}^{\frac{\lambda}{2}} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{z^2}{2\sigma^2}}\, dz \tag{32}$$

$$R = \frac{\sigma}{2}\left(\Phi^{-1}\left(\underline{p_A}\right) - \Phi^{-1}\left(\overline{p_B}\right)\right) \approx 0 \tag{33}$$

Therefore, we choose $\lambda = \sigma^2$ as the quantization threshold in this work, for ensuring the region $[-\lambda/2, \lambda/2]$ owns the largest area but does not dominate other two $(-\infty, -\lambda/2]$ and $[\lambda/2, \infty)$ in the PDF of Gaussian kernel.

A.3 Algorithm

The following are the detailed descriptions of our PCMU method step by step: (1) Train the model in a usual manner with loss function $\mathcal{L}$, e.g., cross-entropy for image classification, and model parameter w; (2) Calculate the gradient $G(x, y) = \frac{\partial \mathcal{L}(x,y;w)}{\partial w}$ in Eq.(5) in the submission; (3) Compute the gradient average $\bar{G}$ in terms of the gradient $G(x_i, y_i)$ of each sample (x_i, y_i) in Eq.(20) in the submission; (4) quantize each dimension t $(t = 1, \cdots, T)$ of the continuous gradient plus Gaussian noise $Q^t(\bar{G} + \varepsilon)$ in Eq.(21) over a discrete three-class space $\{-1, 0, 1\}$, for mimicking the classification in the randomized smoothing for certified robustness based on Eq.(6) in the submission; (5) Perform the randomized gradient smoothing for certified machine unlearning $S^{t\prime}(\bar{G}) = \underset{c\in\{-1,0,1\}}{\arg\max}\, \underset{\varepsilon\sim\mathcal{D}}{\mathbb{P}}(Q^t(\bar{G} + \varepsilon) = c)$ in Eq.(21) in the submission; (6) Derive the certified radius R' in Eq.(24) and the certified budget B' of data removals in Eq.(26) in the submission; (7) Integrate the model training for a specific learning task (e.g., image classification), randomized gradient smoothing, and gradient quantization into a unified framework for directly training a machine unlearning model with the data removal certificates as a guidance, for guaranteeing that the model parameters and gradients keep unchanged against the data removals within the certified budget, in terms of $w = w - \eta[S^{1\prime}(\bar{G}), \cdots, S^{T\prime}(\bar{G})]$ with smoothed and quantized gradients in Eq.(29) in the submission; and (8) Enter the next training round until convergence.

A.4 Proof of Theorems

Lemma 1. *[Neyman-Pearson] Let X and Y be random variables in $\mathbb{R}^d$ with densities μ_x and μ_Y. Let $h : \mathbb{R}^d \to \{0, 1\}$ be a random or deterministic function. Then:*

1. *If $S = \{z \in \mathbb{R}^d : \frac{\mu_Y(z)}{\mu_X(z)} \leq t\}$ for some $t > 0$ and $\mathbb{P}(h(X) = 1) \geq \mathbb{P}(X \in S)$, then $\mathbb{P}(h(Y) = 1) \geq \mathbb{P}(Y = S)$.*
2. *If $S = \{z \in \mathbb{R}^d : \frac{\mu_Y(z)}{\mu_X(z)} \geq t\}$ for some $t > 0$ and $\mathbb{P}(h(X) = 1) \leq \mathbb{P}(X \in S)$, then $\mathbb{P}(h(Y) = 1) \leq \mathbb{P}(Y = S)$.*

Proof. Please refer to the book [23] for detailed proof.

Lemma 2. *[Neyman-Pearson for Gaussians with different means] Let $X \sim \mathcal{N}(x, \sigma^2 I)$ and $Y \sim \mathcal{N}(x + \delta, \sigma^2 I)$. Let $h : \mathbb{R}^d \to \{0, 1\}$ be any random or deterministic function. Then:*

1. *If $S = \{z \in \mathbb{R}^d : \delta^T z \leq \beta\}$ for some β and $\mathbb{P}(h(X) = 1) \geq \mathbb{P}(X \in S)$, then $\mathbb{P}(h(Y) = 1) \geq \mathbb{P}(Y = S)$.*
2. *If $S = \{z \in \mathbb{R}^d : \delta^T z \geq \beta\}$ for some $\beta > 0$ and $\mathbb{P}(h(X) = 1) \leq \mathbb{P}(X \in S)$, then $\mathbb{P}(h(Y) = 1) \leq \mathbb{P}(Y = S)$.*

Proof. Please refer to the paper [23] for detailed proof.

Theorem 2. *The error introduced by the Taylor expansion of $F_c^t(\hat{\bar{x}}, \hat{\bar{y}})$ at $(\bar{x}, \bar{y})$ is*

$$\epsilon \leq \sum_{j=0}^{\infty} || \frac{L_{j+1}}{(j+1)!} || \cdot ||\sigma M||^{j+1} \tag{34}$$

where $L_j = \max_{k=1,\cdots,j} \frac{\partial^j h_i}{\partial x^k \partial y^{j-k}}$ and M is the number of sampled points.

Proof. When we have the Taylor expansion of $F_c^t(\hat{\bar{x}}, \hat{\bar{y}})$ at $(\bar{x}, \bar{y})$

$$\begin{aligned} F_c^t(\hat{\bar{x}}, \hat{\bar{y}}) =& F_c^t(\bar{x}, \bar{y}) + \frac{\partial F_c^t}{\partial x}(\bar{x}, \bar{y})(\hat{\bar{x}} - \bar{x}) + \frac{\partial F_c^t}{\partial y}(\bar{x}, \bar{y})(\hat{\bar{y}} - \bar{y}) + \\ &+ \frac{1}{2}\{\frac{\partial^2 F_c^t(\bar{x}, \bar{y})}{\partial x^2}(\hat{\bar{x}} - \bar{x})^2 + 2\frac{\partial^2 F_c^t(\bar{x}, \bar{y})}{\partial x \partial y}(\hat{\bar{x}} - \bar{x})(\hat{\bar{y}} - \bar{y}) + \frac{\partial^2 F_c^t(\bar{x}, \bar{y})}{\partial y^2}(\hat{\bar{y}} - \bar{y})^2\} \\ &+ \cdots + O_j(\hat{\bar{x}}, \hat{\bar{y}}) \end{aligned} \tag{35}$$

where $O_j(\hat{\bar{x}}, \hat{\bar{y}}) = \frac{1}{(j+1)!}\{\sum \cdots \sum \frac{\partial^k F_c^t(\xi)}{\partial x^k \partial y^{j+1-k}}(\hat{\bar{x}} - \bar{x})^k (\hat{\bar{y}} - \bar{y})^{j+1-k}\}$, $\xi \in ((\hat{\bar{x}}, \bar{x}), (\hat{\bar{y}}, \bar{y})), j = 3, \cdots, \infty$, and $k = 1, 2, 3$.

Then

$$\begin{aligned} \epsilon &= ||F_c^t(\hat{\bar{x}}, \hat{\bar{y}}) - F_c^t(\bar{x}, \bar{y})|| \\ &= || \sum_{j=0}^{\infty} \frac{1}{(j+1)!} \{\sum \cdots \sum \frac{\partial^k F_c^t(\xi)}{\partial x^k \partial y^{j+1-k}}(\hat{\bar{x}} - \bar{x})^k (\hat{\bar{y}} - \bar{y})^{j+1-k}\} || \end{aligned} \tag{36}$$

Suppose that F_c^t is at least second-order differentiable, and there exists Lipschitz constants $L_j = \max_{k=1,\cdots,j} \frac{\partial^j h_i}{\partial x^k \partial y^{j-k}}$ in each function, then

$$\epsilon_2 \leq \sum_{j=0}^{\infty} || \frac{L_{j+1}}{(j+1)!} || \cdot ||\varepsilon_x||^k \cdot ||\varepsilon_y||^{j+1-k} \leq \sum_{j=0}^{\infty} || \frac{L_{j+1}}{(j+1)!} || \cdot ||\sigma M||^{j+1} \tag{37}$$

Theorem 3. *Let $\varepsilon \sim \mathcal{D} = \mathcal{N}\left(0, \sigma^2 I\right)$ and $\tilde{S}^t(\bar{x}) = \underset{c \in \{-1,0,1\}}{\operatorname{argmax}} \underset{\varepsilon \sim \mathcal{D}}{\mathbb{P}}(\tilde{F}^t(\bar{x} + \varepsilon) = c)$. Suppose that for a specific $\bar{x} \in \mathbb{R}^d$, there exist $c_A \in \{-1, 0, 1\}$ and $\underline{p_A}, \overline{p_B} \in [0, 1]$ such that:*

$$\mathbb{P}\left(\tilde{F}^t(\bar{x} + \varepsilon) = c_A\right) \geq \underline{p_A} \geq \overline{p_B} \geq \max_{c \neq c_A} \mathbb{P}(\tilde{F}^t(\bar{x} + \varepsilon) = c) \tag{38}$$

Then $\tilde{S}^t(\bar{x} + \delta) = c_A$ for all $\|\delta\|_2 < R$, where

$$R = \frac{\sigma}{2}\left(\Phi^{-1}\left(\underline{p_A}\right) - \Phi^{-1}\left(\overline{p_B}\right)\right) \tag{39}$$

where Φ^{-1} is the inverse of the standard Gaussian CDF.

Proof. Notice that the correlation between the data $\bar{x}$ and its classes $\bar{y}$ is fixed for a given dataset. We denote this correlation as $\bar{y} = H(\bar{x})$ where $H : \mathbb{R}^d \mapsto C$. Thus, the original $F^t(\bar{x}, \bar{y})$ can be rewritten as an equivalent one $\tilde{F}^t(\bar{x})$.

$$\tilde{F}^t(\bar{x}) = F^t(\bar{x}, H(\bar{x})) = F^t(\bar{x}, \bar{y}) \tag{40}$$

Based on the definition of $\tilde{S}^t(\bar{x})$, to prove $\tilde{S}^t(\bar{x}+\delta) = c_A$ for all $\|\delta\|_2 < R$, we first need to demonstrate

$$\mathbb{P}(\tilde{F}^t(\bar{x}+\delta+\varepsilon) = c_A) > \max_{c_B \neq c_A} \mathbb{P}(\tilde{F}^t(\bar{x}+\delta+\varepsilon) = c_B) \tag{41}$$

For ease of presentation, two random variables are defined as follows.

$$u = \bar{x} + \varepsilon = \mathcal{N}(\bar{x}, \sigma^2 I) \tag{42}$$

$$v = \bar{x} + \delta + \varepsilon = \mathcal{N}(\bar{x}+\delta, \sigma^2 I) \tag{43}$$

Based on the given condition in Eq.(38), we know

$$\mathbb{P}(\tilde{F}^t(u)) = c_A) \geq \underline{p_A} \tag{44}$$

$$\mathbb{P}(\tilde{F}^t(v) = c_B) \leq \overline{p_B} \tag{45}$$

We define two half-spaces as follows.

$$X = \{U : \delta^T(U - \bar{x}) \leq \sigma||\delta||\Phi^{-1}(\underline{p_A})\} \tag{46}$$

$$Y = \{U : \delta^T(U - \bar{x}) \geq \sigma||\delta||\Phi^{-1}(1-\overline{p_B})\} \tag{47}$$

Now, we have

$$\begin{aligned}
\mathbb{P}(u \in X) &= \mathbb{P}\left(\delta^T(u-\bar{x}) \leq \sigma\|\delta\|\Phi^{-1}\left(\underline{p_A}\right)\right) = \mathbb{P}\left(\delta^T\mathcal{N}\left(0,\sigma^2 I\right) \leq \sigma\|\delta\|\Phi^{-1}\left(\underline{p_A}\right)\right) \\
&= \mathbb{P}\left(\sigma\|\delta\|z \leq \sigma\|\delta\|\Phi^{-1}\left(\underline{p_A}\right)\right) = \mathbb{P}\left(z \leq \Phi^{-1}\left(\underline{p_A}\right)\right) = \Phi\left(\Phi^{-1}\left(\underline{p_A}\right)\right) \\
&= \underline{p_A}
\end{aligned} \tag{48}$$

where $z \sim \mathcal{N}(0,1)$.

By combining Eq.(44) and Eq.(48), we have

$$\mathbb{P}(\tilde{F}^t(u)) = c_A) \geq \mathbb{P}(u \in X) \tag{49}$$

By applying Lemma 2 with $h(u) = \mathbf{1}[\tilde{F}^t(u) = c_A]$, we obtain

$$\mathbb{P}(\tilde{F}^t(v)) = c_A) \geq \mathbb{P}(v \in X) \tag{50}$$

Similarly, we have

$$\begin{aligned}
\mathbb{P}(u \in Y) &= \mathbb{P}\left(\delta^T(u-\bar{x}) \geq \sigma\|\delta\|\Phi^{-1}\left(1-\overline{p_B}\right)\right) = \mathbb{P}\left(\delta^T\mathcal{N}\left(0,\sigma^2 I\right) \geq \sigma\|\delta\|\Phi^{-1}\left(1-\overline{p_B}\right)\right) \\
&= \mathbb{P}\left(\sigma\|\delta\|z \geq \sigma\|\delta\|\Phi^{-1}\left(1-\overline{p_B}\right)\right) = \mathbb{P}\left(z \geq \Phi^{-1}\left(1-\overline{p_B}\right)\right) = 1 - \Phi\left(\Phi^{-1}\left(1-\overline{p_B}\right)\right) \\
&= \overline{p_B}
\end{aligned} \tag{51}$$

Again, we get

$$\mathbb{P}(\tilde{F}^t(u)) = c_B) \leq \mathbb{P}(u \in Y) \tag{52}$$

and

$$\mathbb{P}(\tilde{F}^t(v)) = c_B) \leq \mathbb{P}(v \in Y) \tag{53}$$

Notice that our proof objective $\tilde{S}^t(\bar{x}+\delta) = c_A$ is equivalent to the following inequality.

$$\mathbb{P}(\tilde{F}^t(v)) = c_A) > \mathbb{P}(\tilde{F}^t(v)) = c_B) \tag{54}$$

If we can prove $\mathbb{P}(v \in X) > \mathbb{P}(v \in Y)$, then we can obtain

$$\mathbb{P}(\tilde{F}^t(v)) = c_A) \geq \mathbb{P}(v \in X) > \mathbb{P}(v \in Y) \geq \mathbb{P}(\tilde{F}^t(v)) = c_B) \tag{55}$$

Now, we calculate the condition satisfying $\mathbb{P}(v \in X) > \mathbb{P}(v \in Y)$.

$$\mathbb{P}(v \in X) = \Phi(\Phi^{-1}(\underline{p_A}) - \frac{||\delta||}{\sigma}) \tag{56}$$

and

$$\mathbb{P}(v \in Y) = \Phi(\Phi^{-1}(\overline{p_B}) + \frac{||\delta||}{\sigma}) \tag{57}$$

$\mathbb{P}(v \in X) > \mathbb{P}(v \in Y)$ is satisfied if and only if

$$||\delta|| < \frac{\sigma}{2}(\Phi^{-1}(\underline{p_A}) - \Phi^{-1}(\overline{p_B})) \tag{58}$$

Therefore, the proof is concluded.

Theorem 4. *Let R be the certified radius of $\bar{x} \in \mathbb{R}^d$ based on $\tilde{S}^t(\bar{x}) = \underset{c \in \{-1,0,1\}}{\operatorname{argmax}} \mathbb{P}_{\varepsilon \sim \mathcal{D}}(\tilde{F}^t(\bar{x}+\varepsilon) = c)$, then the certified budget of data removal is*

$$B \leq N - \frac{9d\sigma^2}{R^2} \tag{59}$$

where N is the number of data samples on the entire training data and B is the maximally allowed number of data samples escaped from the training data.

Proof. Without loss of generality, suppose that $x \sim \mathcal{N}(\bar{x}, \sigma^2 I)$. The complete training data D is partitioned into two subsets: the forgotten data $D_f \subseteq D$ and the remembered data $D_r \subseteq D$ ($D = D_f \cup D_r$, $D_f \cap D_r = \emptyset$). Let $\{x_1, \cdots, x_B\}$ be the data in D_f, $\{x_{B+1}, \cdots, x_N\}$ be the data in D_r, and $\bar{x}_r$ be the average of all data samples in D_r.

$$\bar{x}_r = \frac{1}{N-B} \sum_{x_i \in D_r} x_i \tag{60}$$

We calculate the expectation and variance of $\bar{x}_r$ about possible escape situations.

$$\mathbf{E}(\bar{x}_r) = \frac{1}{N-B} \sum_{i=B+1}^{N} x_i = \bar{x}_r \tag{61}$$

$$\mathbf{Var}(\bar{x}_r) = \mathbf{Var}(\frac{\sum_{i=B+1}^{N} x_i}{N-B}) = \frac{1}{(N-B)^2} \mathbf{Var}(\sum_{i=B+1}^{N} x_i) = \frac{\sigma^2 I}{N-B} \tag{62}$$

Thus, the data samples $x_{B+1}, \cdots, x_N$ in D_r follow $\mathcal{N}(\bar{x}_r, \frac{\sigma^2 I}{N-B})$.

If we want to guarantee the forgotten data (i.e., the data removals) within the certified radius R, then we need to ensure

$$\mathbb{P}\{||\bar{x} - \bar{x}_r|| \leq R\} = 99.73\% \approx 1, \tag{63}$$

in terms of the three-sigma rule. $\bar{x}$ is the average of all data samples in the entire training data.

Thus, we have

$$R \geq 3||\frac{\sigma I}{\sqrt{N-B}}|| \tag{64}$$

Therefore, we obtain

$$B \leq N - \frac{9d\sigma^2}{R^2} \tag{65}$$

By combining Eq.(39) and Eq.(65) together, we further get

$$B \leq N - \frac{36d}{(\Phi^{-1}(\underline{p_A}) - \Phi^{-1}(\overline{p_B}))^2} \tag{66}$$

Theorem 5. *Let R and R' be the certified radii of the above two algorithms respectively and L be the Lipschitz constant of gradient $G(x, y) \in \mathbb{R}^T$, then*

$$R \geq \frac{\sqrt{T}}{L} R' \tag{67}$$

By combining Theorems 4 and 5 together, we derive the certified budget B' of data removal from R'.

$$B' \leq N - \frac{36dL^2}{T(\Phi^{-1}(\underline{p_A}') - \Phi^{-1}(\overline{p_B}'))^2} \tag{68}$$

Proof. Let $G(x, y) \in \mathbb{R}^T$ be the gradient of a machine learning model.

$$G(x, y) = \frac{\partial \mathcal{L}(x, y; w)}{\partial w} \tag{69}$$

Let $G^t(x, y)$ be the t^{th} $(t = 1, \cdots, T)$ dimension of the gradient $G(x, y)$, $\tilde{G}^t(\bar{x}) = G^t(\bar{x}, H(\bar{x})) = G^t(\bar{x}, \bar{y})$, and $\tilde{Q}^t(\tilde{G}^t(\bar{x})) = Q^t(G^t(\bar{x}, \bar{y}))$. We use $\tilde{Q}_c^t(\tilde{G}^t(\bar{x}))$ to represent the c^{th} $(c \in \{-1, 0, 1\})$ component of $\tilde{Q}^t(\tilde{G}^t(\bar{x}))$.

For the randomized data smoothing and gradient quantization method, we have

$$\underline{p_A} = \int_{\bar{x}} \mathbb{P}(\tilde{F}_c^t(\bar{x}))d\bar{x} = \int_{x_i} \mathbb{P}(\tilde{F}_c^t(\mathbf{E}(x_i)))dx_i = \int_{x_i} \mathbb{P}(\tilde{Q}_c^t(\tilde{G}^t(\mathbf{E}(x_i))))dx_i,\ c = c_A \tag{70}$$

$$\overline{p_B} = \int_{\bar{x}} \mathbb{P}(\tilde{F}_c^t(\bar{x}))d\bar{x} = \int_{x_i} \mathbb{P}(\tilde{F}_c^t(\mathbf{E}(x_i)))dx_i = \int_{x_i} \mathbb{P}(\tilde{Q}_c^t(\tilde{G}^t(\mathbf{E}(x_i))))dx_i,\ c \neq c_A \tag{71}$$

$$R = \frac{\sigma}{2}\left(\Phi^{-1}\left(\underline{p_A}\right) - \Phi^{-1}\left(\overline{p_B}\right)\right) \tag{72}$$

For the randomized gradient smoothing and quantization approach, we have

$$\begin{aligned}\underline{p'_A} &= \int_{\bar{G}^t} \mathbb{P}(Q_c^t(\bar{G}^t))d\bar{G}^t = \int_{x_i}\int_{y_i} \mathbb{P}(Q_c^t(\mathbf{E}(G^t(x_i,y_i))))dx_i dy_i \\ &= \int_{x_i} \mathbb{P}(\tilde{Q}_c^t(\mathbf{E}(\tilde{G}^t(x_i))))dx_i,\ c = c_A\end{aligned} \tag{73}$$

$$\begin{aligned}\overline{p'_B} &= \int_{\bar{G}^t} \mathbb{P}(Q_c^t(\bar{G}^t))d\bar{G}^t = \int_{x_i}\int_{y_i} \mathbb{P}(Q_c^t(\mathbf{E}(G^t(x_i,y_i))))dx_i dy_i \\ &= \int_{x_i} \mathbb{P}(\tilde{Q}_c^t(\mathbf{E}(\tilde{G}^t(x_i))))dx_i,\ c \neq c_A\end{aligned} \tag{74}$$

$$R' = \frac{\sigma}{2}\left(\Phi^{-1}\left(\underline{p'_A}\right) - \Phi^{-1}\left(\overline{p'_B}\right)\right) \tag{75}$$

Let L be the Lipschitz constant of gradient $G(x,y)$, for any $\delta > 0$, we have

$$L \cdot ||\delta|| \geq ||\mathbf{E}(\tilde{G}^t(x_i+\delta)) - \mathbf{E}(\tilde{G}^t(x_i))|| = \sqrt{T}R' \tag{76}$$

Then the minimum change in $\bar{x}$ is

$$\min ||\delta|| = \frac{\sqrt{T}}{L}R' \tag{77}$$

This implies

$$R \geq \frac{\sqrt{T}}{L}R' \tag{78}$$

By combining Eq.(65) and Eq.(78), we obtain the certified budget B' of data removals in the randomized gradient smoothing and quantization approach.

$$\begin{aligned}B' &\leq N - \frac{9d\sigma^2}{(\frac{\sqrt{T}}{L}R')^2} \\ &\leq N - \frac{36dL^2}{T(\Phi^{-1}(\underline{p_A}') - \Phi^{-1}(\overline{p_B}'))^2}\end{aligned} \tag{79}$$

Theorem 6. *Let $S^{t\prime}(\bar{G})$ be the randomized gradient smoothing for certified machine unlearning on gradient quantization, L, L_1, and L_2 be the Lipschitz constants of G, Q^t, and $S^{t\prime}$ respectively, i.e.,*

$$||\nabla S^{t\prime}(a) - \nabla S^{t\prime}(b)||_2 \leq L_2 L_1 L||a-b||_2 \ for\ any\ a, b \tag{80}$$

If we run gradient descent for k iterations with a fixed step size $s \leq \frac{1}{L_2 L_1 L}$, it will yield a solution $S^{t\prime(k)}$ which satisfies

$$S^{t\prime}(q^{(k)}) - S^{t\prime}(q^*) \leq \frac{||q^{(0)} - q^*||_2^2}{2sk} \tag{81}$$

where $S^{t\prime}(q^{(0)})$ is the initial solution and $S^{t\prime}(q^*)$ is the local optimal solution.

This means that gradient descent is guaranteed to converge and that it converges with rate $\mathcal{O}(1/k)$.

Proof. Suppose that $S^{t\prime}$ is local convex and differentiable. Let $q^{(s)}$ be the gradient $S^{t\prime}(\bar{G})$ with the randomized gradient smoothing and gradient quantization at the s^{th} training iteration and q^* be the local optimal solution of $q^{(s)}$.

For any a, b in the local convex domain of $S^{t\prime}$, we have

$$\begin{aligned} S^{t\prime}(b) &\leq S^{t\prime}(a) + \nabla S^{t\prime}(a)^T(b-a) + \frac{1}{2}S^{t\prime}(a)||b-a||_2^2 \\ &\leq S^{t\prime}(a) + \nabla S^{t\prime}(a)^T(b-a) + \frac{1}{2}L_2L_1L||b-a||_2^2 \end{aligned} \tag{82}$$

We plug in the gradient descent update by letting $b = a^{(+)} = a - t\nabla F(a)$.

$$\begin{aligned} S^{t\prime}(a^{(+)}) &\leq S^{t\prime}(a) + \nabla S^{t\prime}(a)^T(a^{(+)} - a) + \frac{1}{2}L_2L_1L||a^{(+)} - a||_2^2 \\ &= S^{t\prime}(a) + \nabla S^{t\prime}(a)^T(a - s\nabla S^{t\prime}(a) - a) + \frac{1}{2}L_2L_1L||a - s\nabla S^{t\prime}(a) - a||_2^2 \\ &= S^{t\prime}(a) - \nabla S^{t\prime}(a)^T s\nabla S^{t\prime}(a) + \frac{1}{2}L_2L_1L||s\nabla S^{t\prime}(a)||_2^2 \\ &= S^{t\prime}(a) - s||\nabla S^{t\prime}(a)||_2^2 + \frac{1}{2}L_2L_1Ls^2||\nabla S^{t\prime}(a)||_2^2 \\ &= S^{t\prime}(a) - (1 - \frac{1}{2}L_2L_1Ls)s||\nabla S^{t\prime}(a)||_2^2. \end{aligned} \tag{83}$$

Based on $s \leq \frac{1}{L_2L_1L}$, we have

$$-(1 - \frac{1}{2}L_2L_1Ls) = \frac{1}{2}L_2L_1Ls - 1 \leq -\frac{1}{2} \tag{84}$$

By plugging Eq.(84) into Eq.(83)*, we obtain*

$$S^{t\prime}(a^{(+)}) \leq S^{t\prime}(a) - \frac{1}{2}s||\nabla S^{t\prime}(a)||_2^2 \tag{85}$$

Since $\frac{1}{2}s||\nabla S^{t\prime}(a)||_2^2$ is always non-negative, the inequality in Eq.(85) *implies that the objective function value strictly decreases with the iteration of gradient descent until it reaches the local optimal value $S^{t\prime}(a) = S^{t\prime}(a^*)$.*

Now we need to bound the objective value at the next iteration, $S^{t\prime}(a^{(+)})$, in terms of the local optimal objective value $S^{t\prime}(a^*)$.

Since $S^{t\prime}$ is local convex, we have

$$S^{t\prime}(a^*) \geq S^{t\prime}(a) + \nabla S^{t\prime}(a)^T(a^* - a) \tag{86}$$

$$S^{t\prime}(a) \geq S^{t\prime}(a^*) + \nabla S^{t\prime}(a)^T(a - a^*) \tag{87}$$

According to (84)*, we obtain*

$$S^{t\prime}(a^{(+)}) \leq S^{t\prime}(a^*) + \nabla S^{t\prime}(a)^T(a - a^*) - \frac{s}{2}||\nabla S^{t\prime}(a)||_2^2 \tag{88}$$

Notice that

$$||a - s\nabla S^{t\prime}(a) - a^*||_2^2 = ||a - a^*||_2^2 - 2s\nabla S^{t\prime}(a)^T(a - a^*) + s^2||\nabla S^{t\prime}(a)||_2^2 \tag{89}$$

Thus, we have

$$S^{t\prime}(a^{(+)}) - S^{t\prime}(a^*) \leq \frac{1}{2s}(||a - a^*||_2^2 - ||a - s\nabla S^{t\prime}(a) - a^*||_2^2) \tag{90}$$

Notice that $a^{(+)} = a - s\nabla S^{t\prime}(a)$*, by plugging this into* (90)*, we get*

$$S^{t\prime}(a^{(+)}) - S^{t\prime}(a^*) \leq \frac{1}{2s}(||a - a^*||_2^2 - ||a^{(+)} - a^*||_2^2) \tag{91}$$

By aggregating the terms at all iterations, we have

$$\begin{aligned}\sum_{i=1}^{k} S^{t\prime}(a^{(k)} - S^{t\prime}(a^*)) &\leq \sum_{i=1}^{k} \frac{1}{2s}(||a^{(i-1)} - a^*||_2^2 - ||a^{(i)} - a^*||_2^2) \\ &= \frac{1}{2s}(||a^{(0)} - a^*||_2^2 - ||a^{(k)} - a^*||_2^2) \\ &\leq \frac{1}{2s}(||a^{(0)} - a^*||_2^2)\end{aligned} \tag{92}$$

Finally, since the function $S^{t\prime}$ *keeps decreasing at each iteration, we can conclude*

$$S^{t\prime}(a^{(k)}) - S^{t\prime}(a^*) \leq \frac{1}{k}\sum_{i=1}^{k} S^{t\prime}(a^{(i)}) - S^{t\prime}(a^*) \leq \frac{||a^{(0)} - a^*||_2^2}{2sk} \tag{93}$$

By replacing a *with* q*, the proof is concluded.*

A.5 Additional Experiments

Machine unlearning performance and running time with varying ratios of data removal. Tables 5-15 exhibit the classification accuracy, errors, training time, and unlearning time obtained by eleven machine unlearning approaches by varying the ratio of unlearning request / data removal between 2% and 20% on three datasets of Fashion-MNIST, CIFAR-10, and SVHN respectively. Similar trends are observed for the comparison of machine unlearning effectiveness and efficiency in these figures: our PCMU method achieves the smallest absolute performance difference with the Retrain model, regarding $Accuracy$ (<1%), $Error_t$ (<1%), $Error_r$ (<4%), and $Error_f$ (<1%) on three datasets respectively. Our PCMU method achieves better efficiency than most baseline methods, except DeltaGrad and Unrolling SGD. Our PCMU method performs one-time operation of simultaneous training and unlearning when addressing a series of machine unlearning requests. Thus, our PCMU method has only one running time for multiple unlearning requests, e.g., 2,566 seconds on SVHN for all five unlearning results (2%, 3%, 5%, 10%, and 20%). However, other baselines need to sequentially handle these machine unlearning requests one by one. Therefore, they have multiple running time for varying ratios of data removals, e.g., 1,176, 1,195, 1,231, 1,280, and 1,371 seconds achieved by Unrolling SGD on SVHN for five unlearning requests (2%, 3%, 5%, 10%, and 20%) respectively. The above experiment results demonstrate that PCMU is effective as well as efficient for addressing the machine unlearning problem. This advantage is very important for entitling data owners to the right to have their private data removed from trained complex models at their requests in a timely and cost-efficient manner in privacy-critical applications that usually require near-zero tolerance of data leaking.

Table 5: Performance with 5% data removal and CNN on Fashion-MNIST

	Performance				Runtime (s)		
Metric	$Accuracy$	$Error_t$	$Error_r$	$Error_f$	Training	Unlearning	Total
Retrain	88.18	11.82	10.12	11.53	687	646	1,333
Fisher	86.02	13.98	13.04	13.57	672	550	1,222
certified removal	77.01	22.99	90.06	90.83	720	81	801
DeltaGrad	83.33	16.67	15.52	15.83	563	148	711
NTK	85.95	14.05	13.05	13.53	672	416	1,088
Unrolling SGD	84.74	15.26	39.64	38.34	**362**	63	**425**
SISA	84.79	15.21	13.82	13.73	1,419	1,397	2,816
Adaptive Unlearning	79.93	30.07	18.83	19.43	1,537	1,505	3,042
FedEraser	70.32	29.68	27.73	27.66	669	575	1,244
MCMC unlearning	83.04	16.96	15.22	28.93	965	419	1,384
PCMU	**88.34**	**11.66**	**11.09**	**11.08**	802	**0**	802

Table 6: Performance with 8% data removal and CNN on Fashion-MNIST

	Performance				Runtime (s)		
Metric	$Accuracy$	$Error_t$	$Error_r$	$Error_f$	Training	Unlearning	Total
Retrain	88.21	11.79	9.46	11.31	687	618	1,305
Fisher	86.02	13.98	12.80	13.25	647	1,942	2,589
certified removal	77.57	22.43	89.79	89.85	730	147	877
DeltaGrad	86.07	13.93	12.89	12.60	555	137	692
NTK	86.02	13.98	12.81	13.25	647	1,810	2,457
Unrolling SGD	83.45	16.55	39.04	41.34	**371**	63	**434**
SISA	84.77	15.23	14.33	14.50	1,419	1,384	2,803
Adaptive Unlearning	76.18	23.82	24.56	24.13	1,537	1,495	3,032
FedEraser	66.75	33.25	32.97	35.06	673	593	1,266
MCMC unlearning	86.83	13.17	7.91	34.15	619	651	1,270
PCMU	**88.34**	**11.66**	**10.86**	**10.93**	802	**0**	802

Table 7: Performance with 15% data removal and CNN on Fashion-MNIST

	Performance				Runtime (s)		
Metric	$Accuracy$	$Error_t$	$Error_r$	$Error_f$	Training	Unlearning	Total
Retrain	87.78	12.22	9.35	11.57	687	552	1,239
Fisher	86.21	13.79	12.49	13.04	693	2,006	2,699
certified removal	76.18	23.82	89.99	90.10	736	266	1,002
DeltaGrad	85.36	14.64	13.49	13.51	557	146	703
NTK	86.39	13.61	12.51	13.13	693	1,875	2,568
Unrolling SGD	86.18	13.82	38.52	39.00	**359**	64	**423**
SISA	84.18	15.82	14.78	14.55	1,419	1,366	2,785
Adaptive Unlearning	86.02	13.98	12.80	13.25	1,537	1,453	2,990
FedEraser	73.11	26.89	26.38	26.52	654	576	1,230
MCMC unlearning	83.22	16.78	7.95	77.03	961	803	1,764
PCMU	**88.40**	**11.60**	**10.08**	**10.97**	802	**0**	802

Table 8: Performance with 5% data removal and LeNet on CIFAR-10

	Performance				Runtime (s)		
Metric	$Accuracy$	$Error_t$	$Error_r$	$Error_f$	Training	Unlearning	Total
Retrain	64.24	35.76	26.67	37.04	846	826	1,672
Fisher	62.37	37.63	33.78	33.53	655	1,165	1,820
certified removal	40.11	59.89	89.90	90.44	749	88	837
DeltaGrad	62.79	37.21	22.31	21.56	901	503	1,404
NTK	62.72	37.28	33.07	34.17	655	899	1,554
Unrolling SGD	58.17	41.83	43.06	40.40	**511**	238	**749**
SISA	58.23	41.77	34.65	33.46	1,594	1,585	3,179
Adaptive Unlearning	43.20	56.80	56.68	62.21	1,176	317	1,493
FedEraser	51.76	48.24	47.81	48.67	1,190	972	2,162
MCMC unlearning	61.02	38.98	5.70	10.32	1,322	405	2,330
PCMU	**64.33**	**35.67**	**24.57**	**36.32**	903	**0**	903

Table 9: Performance with 8% data removal and LeNet on CIFAR-10

	Performance				Runtime (s)		
Metric	$Accuracy$	$Error_t$	$Error_r$	$Error_f$	Training	Unlearning	Total
Retrain	64.36	35.64	28.14	37.43	846	751	1,597
Fisher	62.38	37.62	33.74	33.88	655	1,478	2,133
certified removal	39.33	60.67	89.78	90.28	749	144	893
DeltaGrad	61.66	38.34	22.45	22.95	867	486	1,353
NTK	62.71	37.29	33.28	34.83	655	1,149	1,804
Unrolling SGD	56.92	43.08	44.22	45.40	**511**	202	**713**
SISA	58.21	41.79	34.63	33.37	1,594	1,591	3,185
Adaptive Unlearning	42.88	57.12	55.21	55.97	1,176	316	1,492
FedEraser	49.50	50.50	43.36	48.76	1,190	965	2,155
MCMC unlearning	62.00	38.00	5.12	15.40	1,322	1,138	2,460
PCMU	**64.33**	**35.67**	**24.04**	**37.43**	903	**0**	903

Table 10: Performance with 15% data removal and LeNet on CIFAR-10

	Performance				Runtime (s)		
Metric	$Accuracy$	$Error_t$	$Error_r$	$Error_f$	Training	Unlearning	Total
Retrain	64.12	35.88	26.97	36.24	846	708	1,554
Fisher	62.40	37.60	33.72	33.88	655	5,268	5,923
certified removal	37.01	62.99	89.86	90.19	749	304	1,053
DeltaGrad	60.73	39.27	23.38	21.63	909	499	1,408
NTK	61.87	38.13	33.72	36.16	655	5,144	5,799
Unrolling SGD	58.94	41.06	41.07	44.40	**511**	353	**864**
SISA	57.73	42.27	35.30	34.08	1,594	1,499	3,093
Adaptive Unlearning	43.51	56.49	56.14	57.93	1,176	295	1,471
FedEraser	51.37	48.63	49.95	48.91	1,190	1,000	2,190
MCMC unlearning	61.99	38.01	4.64	34.42	1,322	1,515	2,837
PCMU	**64.65**	**35.35**	**25.42**	**37.60**	903	**0**	903

Table 11: Performance with 2% data removal and ResNet-18 on SVHN

	Performance				Runtime (s)		
Metric	$Accuracy$	$Error_t$	$Error_r$	$Error_f$	Training	Unlearning	Total
Retrain	93.34	6.66	3.38	6.48	1,433	1,401	2,834
Fisher	83.13	16.87	87.58	87.30	1,191	2,085	3,276
certified removal	92.25	7.75	0.34	0.41	1,746	42	1,788
DeltaGrad	91.65	8.35	0.61	0.56	1,236	774	2,010
NTK	90.01	9.99	88.34	87.24	1,191	1,854	3,045
Unrolling SGD	87.69	12.31	10.30	10.85	**787**	389	**1,176**
SISA	90.65	9.35	8.05	8.23	2,021	1,983	4,004
Adaptive Unlearning	86.35	13.65	11.32	13.70	728	728	1,456
FedEraser	88.61	11.39	10.92	12.97	2,464	1,136	3,600
MCMC unlearning	91.35	8.65	0.28	7.17	1,854	2,053	3,907
PCMU	**93.41**	**6.59**	**3.50**	**6.95**	2,566	**0**	2,566

Table 12: Performance with 3% data removal and ResNet-18 on SVHN

	Performance				Runtime (s)		
Metric	$Accuracy$	$Error_t$	$Error_r$	$Error_f$	Training	Unlearning	Total
Retrain	93.74	6.26	3.61	6.96	1,433	1,418	2,851
Fisher	83.82	16.18	87.77	87.03	1,022	2,298	3,320
certified removal	92.00	8.00	0.36	0.27	1,719	56	1,775
DeltaGrad	90.91	9.09	0.87	0.79	1,261	760	2,021
NTK	91.91	8.09	88.46	86.99	1,022	2,053	3,075
Unrolling SGD	85.22	14.78	10.90	10.78	**776**	419	**1,195**
SISA	90.50	9.50	8.08	7.84	2,021	1,957	3,978
Adaptive Unlearning	87.22	12.78	11.98	12.65	740	724	1,464
FedEraser	89.11	10.89	11.30	11.21	2,470	1,146	3,616
MCMC unlearning	91.31	8.69	0.45	10.69	1,855	3,123	4,978
PCMU	**93.41**	**6.59**	**3.93**	**7.29**	2,566	**0**	2,566

Table 13: Performance with 5% data removal and ResNet-18 on SVHN

	Performance				Runtime (s)		
Metric	$Accuracy$	$Error_t$	$Error_r$	$Error_f$	Training	Unlearning	Total
Retrain	93.40	6.60	4.63	8.11	1,433	1,386	2,819
Fisher	83.06	16.94	88.05	89.19	1,109	3,404	4,513
certified removal	91.28	8.72	0.58	0.68	1,738	90	1,828
DeltaGrad	90.98	9.02	1.18	1.17	1,244	759	2,003
NTK	90.55	9.45	88.66	89.87	1,109	3,152	4,261
Unrolling SGD	85.93	14.07	11.64	11.30	**778**	453	**1,231**
SISA	90.53	9.47	8.09	9.20	2,021	1,922	3,943
Adaptive Unlearning	86.11	13.89	12.32	14.03	726	731	1,457
FedEraser	89.01	10.99	10.62	12.03	2,455	1,151	3,606
MCMC unlearning	90.33	9.67	0.89	20.31	2,017	5,149	7,166
PCMU	**93.41**	**6.59**	**4.28**	**7.88**	2,566	**0**	2,566

Table 14: Performance with 10% data removal and ResNet-18 on SVHN

	Performance				Runtime (s)		
Metric	$Accuracy$	$Error_t$	$Error_r$	$Error_f$	Training	Unlearning	Total
Retrain	92.73	7.27	4.58	7.73	1,433	1,351	2,784
Fisher	Failed due to out of memory				Failed due to out of memory		
certified removal	91.08	8.92	0.39	0.42	1,755	171	1,926
DeltaGrad	88.13	11.87	4.02	3.88	1,257	755	2,012
NTK	Failed due to out of memory				Failed due to out of memory		
Unrolling SGD	87.04	12.96	10.43	10.41	**794**	486	**1,280**
SISA	89.98	10.02	8.26	8.35	2,021	1,811	3,832
Adaptive Unlearning	84.29	15.71	14.23	15.75	735	720	1,455
FedEraser	89.72	10.28	10.40	10.98	2,402	1,116	3,518
MCMC unlearning	88.33	11.67	1.23	32.61	1,860	10,194	12,054
PCMU	**93.41**	**6.59**	**4.44**	**7.62**	2,566	**0**	2,566

Table 15: Performance with 20% data removal and ResNet-18 on SVHN

	Performance				Runtime (s)		
Metric	$Accuracy$	$Error_t$	$Error_r$	$Error_f$	Training	Unlearning	Total
Retrain	92.59	7.41	4.80	8.16	1,433	1,128	2,561
Fisher	Failed due to out of memory				Failed due to out of memory		
certified removal	90.98	9.02	0.94	0.90	1,733	329	2,062
DeltaGrad	87.20	12.80	3.98	4.20	1,248	724	1,972
NTK	Failed due to out of memory				Failed due to out of memory		
Unrolling SGD	84.65	15.35	11.10	11.20	**779**	592	**1,371**
SISA	90.20	9.80	8.15	9.22	2,021	1,632	3,653
Adaptive Unlearning	81.19	18.81	18.34	18.83	732	726	1,458
FedEraser	89.28	10.72	10.86	12.05	2,397	1,100	3,497
MCMC unlearning	84.55	15.45	3.24	32.71	1,857	20,331	22,188
PCMU	**93.41**	**6.59**	**4.41**	**7.76**	2,566	**0**	2,566

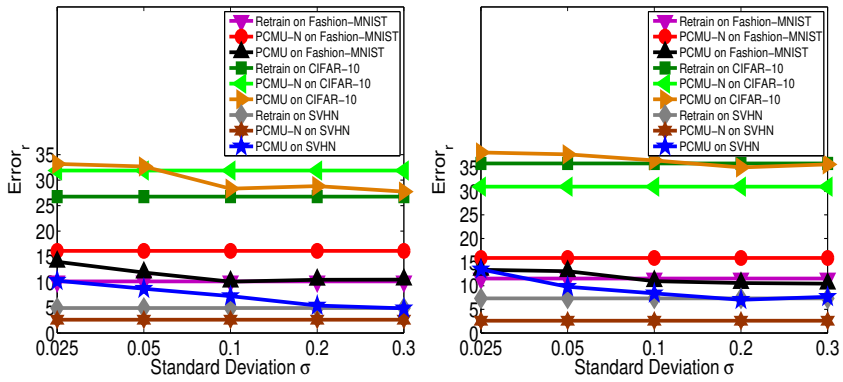
Machine unlearning performance and running time with different smoothing techniques. Tables 16 and 17 evaluate the impact of different smoothing techniques on our proposed prompt certified machine unlearning method, PCMU, over two popular image classification datasets: Fashion-MNIST and CIFAR-10. We replace the randomized smoothing component in the original PCMU model with Laplacian smoothing and uniform smoothing respectively. It is observed that two PCMU variants with Laplacian smoothing and uniform smoothing achieve the close performance to the original PCMU model with randomized smoothing, showing the generality of PCMU to the machine unlearning. Compared with the results achieved by the nine state-of-the-art baselines in Tables 2 and 4, two PCMU variants still substantially outperform the performance of other baselines in most experiments. We will include all the experiment results in this rebuttal into the submission.

Table 16: Performance with 20% data removal and CNN on Fashion-MNIST

	Performance				Runtime (s)		
Metric	$Accuracy$	$Error_t$	$Error_r$	$Error_f$	Training	Unlearning	Total
Retrain	88.21	11.79	9.75	11.76	687	561	1,248
PCMU	**88.34**	**11.66**	**10.25**	**11.47**	802	0	802
PCMU+Laplacian Smoothing	86.16	13.84	12.29	13.20	773	0	773
PCMU+Uniform Smoothing	86.86	13.14	10.90	12.77	804	0	804

Table 17: Performance with 20% data removal and LeNet on CIFAR-10

	Performance				Runtime (s)		
Metric	$Accuracy$	$Error_t$	$Error_r$	$Error_f$	Training	Unlearning	Total
Retrain	63.29	36.71	24.59	36.89	846	673	1,519
PCMU	**64.33**	**35.67**	**25.18**	**35.32**	903	**0**	903
PCMU+Laplacian Smoothing	61.77	38.23	29.41	38.30	928	0	928
PCMU+Uniform Smoothing	62.17	37.83	28.06	38.60	936	0	936



(a) $Error_r$ (b) $Error_f$

Figure 3: Errors with varying standard deviation on three datasets

Machine unlearning performance and running time based on removals of class samples. In the submission, all the formulae, methods, and theories do not have assumptions or restrictions regarding the distribution of data removals. Thus, our proposed prompt certified machine unlearning method, PCMU, is able to work on the case of removals of class samples, as long as the actual data removals are below the certified budget of data removals. In fact, the removals of arbitrary data samples in our experiments is more general than the removals of class samples.

In order to validate the performance of our PCMU for this special machine unlearning problem, we randomly choose one class and remove all samples from this class over CIFAR-10. Table 18 exhibits the corresponding experiment results, which demonstrate that PCMU is also effective as well as efficient for addressing this special machine unlearning problem.

Table 18: Performance with data removal of samples from one class and LeNet on CIFAR-10

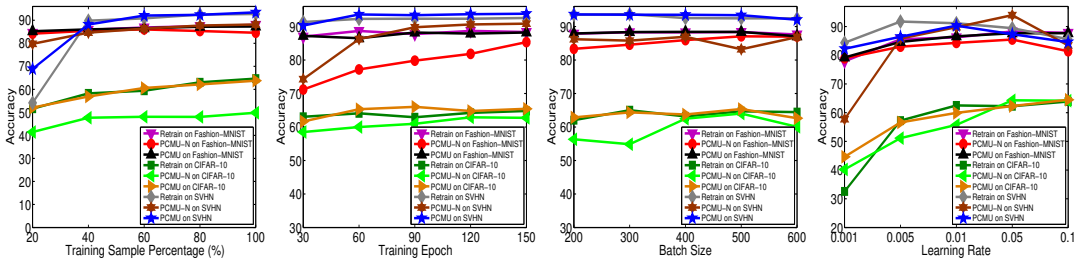
	Performance				Runtime (s)		
Metric	$Accuracy$	$Error_t$	$Error_r$	$Error_f$	Training	Unlearning	Total
Retrain	58.91	41.09	28.23	100	859	738	1,597
Fisher	62.28	37.72	34.28	29.51	1,459	1,490	2,949
certified removal	38.40	61.61	61.73	60.32	**886**	218	1,104
NTK	62.71	37.29	32.47	42.20	1,459	1,353	2,812
MCMC unlearning	57.13	42.87	16.44	49.52	1,565	734	2,299
PCMU	**60.15**	**39.85**	**26.51**	**72.18**	926	**0**	**926**

A.6 Parameter Sensitivity

In this section, we conduct more experiments to validate the sensitivity of various parameters in our PCMU method for the certified machine unlearning task.

Impact of standard deviation. Figure 3 (a) and (b) measure the effect of standard deviation of the Gaussian distribution in the randomized gradient smoothing for machine unlearning on $Error_r$ and $Error_f$ by varying σ from 0.025 to 0.3. The error scores achieved by the Retrain and PCMU-N models keep unchanged with varying σ. We have observed similar results in these two figures: The error curves by PCMU initially decrease quickly and then become stable when σ continuously increases. A suitable σ can help utilize the randomized gradient smoothing and quantization for directly training a certfied machine unlearning model in advance. A too large σ beyond some thresholds does not affect the performance of machine unlearning any more.

Influence of training sample percentage. Figure 4 (a) shows the influence of training sample percentage in our PCMU model by varying it from 20% to 100%. We make the observations on the quality by three machine unlearning methods. (1) The accuracy by our PCMU model is very close to that of the Retrain method in most experiments. (2) The performance curves keep increasing when the number of training samples increases. (3) PCMU outperforms PCMU-N in most tests with the smallest accuracy difference with the Retrain method. When there are many training samples available ($\geq 40\%$), the quality improvement by PCMU is obvious. A reasonable explanation is more



(a) Training Sample Percentage (b) Training Epoch (c) Batch Size (d) Learning Rate

Figure 4: Performance with varying Parameters on three datasets

training data makes PCMU be more resilient to machine unlearning under suitable ratios of data removals.

Impact of training epochs. Figure 4 (b) exhibits the sensitivity of training epochs of our PCMU model by varying them from 30 and 150. As we can see, the performance curves continuously increase with increasing training epochs. This is consistent with the fact that more training epochs makes the image classification models be resilient to machine unlearning under suitable ratios of data removals. It is observed that the accuracy scores oscillate within the range of 3.1% on three datasets.

Sensitivity of batch size. Figure 4 (c) exhibits the sensitivity of batch size of machine unlearning models in our PCMU model by varying them from 200 and 600. It is observed that the performance curves keep relatively stable when we continuously change the batch size. This demonstrates that our PCMU method is insensitive to the batch size of machine unlearning. No matter what the batch size is, our PCMU method can always achieve the superior performance in all tests, showing the effectiveness of our PCMU method to the machine unlearning.

Influence of learning rates. Figure 4 (d) shows the influence of learning rate in our PCMU model by varying it from 0.001 to 0.1. We have observed that the accuracy initially raises when the learning rate increases. Intuitively, a large learning rate can help the algorithm quickly find the optimal solution and thus help improve the quality of machine unlearning. Later on, the performance curves decrease quickly when the learning rate continuously increases. A reasonable explanation is that a too large learning rate may miss the optimal solution with large step size in the search process. Thus, it is important to determine the optimal learning rate for the machine unlearning.

A.7 Experimental Details

Environment. The experiments were conducted on a compute server running on Red Hat Enterprise Linux 7.2 with 2 CPUs of Intel Xeon E5-2650 v4 (at 2.66 GHz) and 8 GPUs of NVIDIA GeForce GTX 2080 Ti (with 11GB of GDDR6 on a 352-bit memory bus and memory bandwidth in the neighborhood of 620GB/s), 256GB of RAM, and 1TB of HDD. Overall, the experiments took about 3 days in a shared resource setting. We expect that a consumer-grade single-GPU machine (e.g., with a 2080 Ti GPU) could complete the full set of experiments in around 4-5 days, if its full resources were dedicated. The codes were implemented in Python 3.7.3 and PyTorch 1.0.14. We also employ Numpy 1.16.4 and Scipy 1.3.0 in the implementation. Since the datasets used are all public datasets and our methodologies and the hyperparameter settings are explicitly described in Section 3, 4, 5, and A.7, our codes and experiments can be easily reproduced on top of a GPU server.

Training. We study image classification networks on three standard image datasets: Fashion-MNIST [2], CIFAR-10 [3], and SVHN [4]. The above three image datasets are all public datasets, which allow researchers to use for non-commercial research and educational purposes. We use 60,000 examples as training data and 10,000 examples as test data for Fashion-MNIST. We train the machine unlearning model on the CIFAR-10 training set and test it on the CIFAR-10 test set. We use 73,257 digits as training data and 26,032 digits as test data for SVHN. We train a convolutional neural network (CNN) on Fashion-MNIST for clothing classification. We train LeNet over CIFAR-10 for

[2] https://github.com/zalandoresearch/fashion-mnist

[3] https://www.cs.toronto.edu/∼kriz/cifar.html

[4] http://ufldl.stanford.edu/housenumbers/

image classification. We apply the ResNet-18 architecture on SVHN for street view house number identification. The neural networks are trained with Kaiming initialization [52] using SGD for 120 epochs with an initial learning rate of 0.05 and batch size 500. The learning rate is decayed by a factor of 0.1 at 1/2 and 3/4 of the total number of epochs. In addition, we run each experiment for 3 trials for obtaining more stable results.

Implementation. For nine state-of-the-art machine unlearning models of Fisher [5], certified removal [6], DeltaGrad [7], NTK [8], Unrolling SGD [9], SISA [10], Adaptive Unlearning [11], FedEraser [12], and MCMC unlearning [13], we utilized the same model architecture as the official open-source implementation and default parameter settings provided by the original authors for machine unlearning in all experiments. All hyperparameters are standard values from reference codes or prior works. We validate the performance of different machine unlearning methods with a range of ratio of data removals $\{5\%, 8\%, 10\%, 15\%, 20\%\}$. All models were trained for 120 epochs, with a batch size of 500, and a learning rate of 0.05. The above open-source codes from the GitHub are licensed under the MIT License, which only requires preservation of copyright and license notices and includes the permissions of commercial use, modification, distribution, and private use.

For our PCMU model, we performed hyperparameter selection by performing a parameter sweep on standard deviation $\sigma \in \{0.025, 0.05, 0.1, 0.2, 0.3, 0.5, 1\}$ in the Gaussian distribution, quantization threshold $\lambda \in \{\sigma^2/4, \sigma^2/2, \sigma^2, 2\sigma^2, 4\sigma^2\}$, ratio of data removals $\{5\%, 8\%, 10\%, 15\%, 20\%\}$, training epochs of the machine unlearning model $\in \{30, 60, 90, 120, 150\}$, batch size for training the model $\in \{200, 300, 400, 500, 600\}$, and learning rate $\in \{0.001, 0.005, 0.01, 0.05, 0.1, 0.5\}$. We select the best parameters over 50 epochs of training and evaluate the model at test time.

Hyperparameter settings.

Unless otherwise explicitly stated, we used the following default parameter settings in the experiments.

Table 19: Hyperparameter Settings

Parameter	Value
Training data on Fashion-MNIST	60,000
Test data ratio on Fashion-MNIST	10,000
Training data on CIFAR-10	50,000
Test data on CIFAR-10	10,000
Training data on SVHN	73,257
Test data on SVHN	26,032
Standard deviation σ in the Gaussian distribution	0.1
Quantization threshold λ	σ^2
Ratio of data removals	20%
Training epochs of the machine unlearning model	120
Batch size for training the model	500
Learning rate	0.05

A.8 Potential Negative Societal Impacts and Limitations

In this work, the three image datasets are all open-released datasets [138, 66, 95], which allow researchers to use for non-commercial research and educational purposes. These three datasets are widely used in training/evaluating the image classification. All baseline codes are open-accessed resources that are from the GitHub and licensed under the MIT License, which only requires

[5] https://github.com/AdityaGolatkar/SelectiveForgetting

[6] https://github.com/facebookresearch/certified-removal

[7] https://github.com/thuwuyinjun/DeltaGrad

[8] https://github.com/AdityaGolatkar/SelectiveForgetting

[9] https://github.com/cleverhans-lab/unrolling-sgd

[10] https://github.com/cleverhans-lab/machine-unlearning

[11] https://github.com/ChrisWaites/adaptive-machine-unlearning

[12] https://www.dropbox.com/s/1lhx962axovbbom/FedEraser-Code.zip?dl=0

[13] https://github.com/fshp971/mcmc-unlearning

preservation of copyright and license notices and includes the permissions of commercial use, modification, distribution, and private use.

To our best knowledge, this work is the first to execute one-time operation of simultaneous training and unlearning in advance for a series of machine unlearning requests, as long as the actual data removals are below the certified budget of data removals, while there is no need to know the forgotten data, by leveraging the theory of randomized smoothing and gradient quantization. Many machine learning applications often need to collect massive amount of data from third parties for model training. This raises a legitimate privacy risk: training data can be practically reconstructed from models [35, 112, 125, 7, 85, 87, 14]. In addition, modern privacy regulations, such as the European Union's General Data Protection Regulation (GDPR) [98] and the California Consumer Privacy Act (CCPA) [71], enforce the right to be forgotten, i.e., entitle data owners to the right to have their private data removed at their requests [87, 83, 20]. Our framework is able to resolve the requests of data removal in a timely and cost-efficient manner. Our framework can play an important building block for a wide variety of privacy-critical applications that usually require near-zero tolerance of data leaking, such as financial and health data analyses. This paper is primarily of a theoretical nature. We expect our findings to produce positive impact, i.e, significantly improve the efficiency of machine unlearning models by simultaneously training and unlearning in advance. To our best knowledge, we do not envision any immediate negative societal impacts of our results, such as security, privacy, and fairness issues.

An important product of this paper is to explore the possibility of simultaneous training and unlearning in advance as well as one-time unlearning. Due to high-dimensional double integrals or non-integrable mapping between samples and labels in the randomized data smoothing and gradient quantization method, the randomized gradient smoothing and quantization approach is designed to produce high confidence certificates for the certified machine unlearning. Our theoretical framework can inspire further improved development and implementations on certified machine unlearning with better applicability and efficiency from the academic institutions and industrial research labs.