Machine Unlearning: A Survey

HENG XU, TIANQING ZHU, and LEFENG ZHANG, University of Technology Sydney, Australia
WANLEI ZHOU, City University of Macau, China
PHILIP S. YU, University of Illinois at Chicago, United States

Machine learning has attracted widespread attention and evolved into an enabling technology for a wide range of highly successful applications, such as intelligent computer vision, speech recognition, medical diagnosis, and more. Yet, a special need has arisen where, due to privacy, usability, and/or *the right to be forgotten*, information about some specific samples needs to be removed from a model, called machine unlearning. This emerging technology has drawn significant interest from both academics and industry due to its innovation and practicality. At the same time, this ambitious problem has led to numerous research efforts aimed at confronting its challenges. To the best of our knowledge, no study has analyzed this complex topic or compared the feasibility of existing unlearning solutions in different kinds of scenarios. Accordingly, with this survey, we aim to capture the key concepts of unlearning techniques. The existing solutions are classified and summarized based on their characteristics within an up-to-date and comprehensive review of each category's advantages and limitations. The survey concludes by highlighting some of the outstanding issues with unlearning techniques, along with some feasible directions for new research opportunities.

CCS Concepts: • **Security and privacy → Human and societal aspects of security and privacy;**

Additional Key Words and Phrases: Machine learning, deep learning, machine unlearning, sample removal, data privacy, model usability

ACM Reference format:
Heng Xu, Tianqing Zhu, Lefeng Zhang, Wanlei Zhou, and Philip S. Yu. 2023. Machine Unlearning: A Survey. *ACM Comput. Surv.* 56, 1, Article 9 (August 2023), 36 pages.
https://doi.org/10.1145/3603620

1 INTRODUCTION

In recent years, machine learning has seen remarkable progress and wide exploration across every field of **artificial intelligence (AI)** [1]. However, as AI becomes increasingly data-dependent, more and more factors, such as privacy concerns, regulations, and laws, are leading to a new type of request—to delete information. Specifically, concerned parties are requesting that particular samples be removed from a training dataset and that the impact of those samples be removed

This article is supported in part by the Australian Research Council Discovery DP200100946 and DP230100246, and NSF under grants III-1763325, III-1909323, III-2106758, and SaTC-1930941.
Authors' addresses: H. Xu, T. Zhu, and L. Zhang, University of Technology Sydney, 123 Broadway, Ultimo NSW 2007, Australia; emails: Heng.Xu-2@student.uts.edu.au, Tianqing.Zhu@uts.edu.au, Lefeng.Zhang@student.uts.edu.au; W. Zhou, City University of Macau, Avenida Padre Tomás Pereira Taipa, Macau 999078, China; email: wlzhou@cityu.edu.mo; P. S. Yu, University of Illinois at Chicago, 1200 W Harrison St, Chicago, IL 60607; email: psyu@cs.uic.edu.

0360-0300/2023/08-ART9 $15.00
https://doi.org/10.1145/3603620

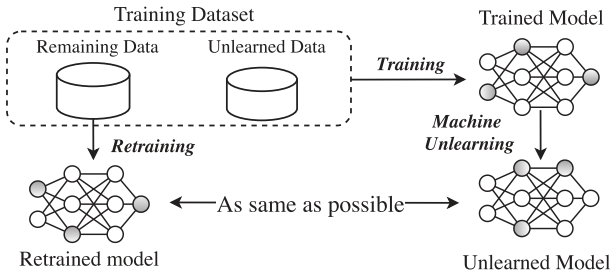

Fig. 1. Illustration of machine unlearning.

from an already-trained model [2–4]. This is because membership inference attacks [5] and model inversion attacks [6] can reveal information about the specific contents of a training dataset. More importantly, legislators around the world have wisely introduced laws that grant users *the right to be forgotten* [7, 8]. These regulations, which include the European Union's **General Data Protection Regulation (GDPR)** [9], the **California Consumer Privacy Act (CCPA)** [10], the **Act on the Protection of Personal Information (APPI)** [11], and Canada's proposed **Consumer Privacy Protection Act (CPPA)** [12], compel the deletion of private information.

1.1 The Motivation of Machine Unlearning

Machine unlearning (a.k.a. selectively forgetting, data deletion, or scrubbing) requires that the samples and their influence can be completely and quickly removed from a training dataset and a trained model [13–15]. Figure 1 illustrates an example of machine unlearning for a trained model.

Machine unlearning is not only motivated by regulations and laws; it also stems from the privacy and security concerns of the data provider, as well as the requirement of model owners themselves. In fact, removing the influence of outlier training samples from a model will lead to higher model performance and robustness [16]. There are existing data protection techniques that are similar to machine unlearning, but they differ in either objectives or rationales.

Here, we briefly discuss the main differences between current techniques and machine unlearning.

- **Differential Privacy.** Differential privacy [17, 18] guarantees that by looking at a model output, one cannot tell whether a sample is in the training dataset or not. This technique ensures a subtle bound on the contribution of *every* sample to the final model [19, 20], but machine unlearning is targeted on the removing of *user-specific* training samples.
- **Data Masking.** Data masking [21] is designed to hide sensitive information in the original dataset. It transforms sensitive data to prevent them from being disclosed in unreliable environments [22]. In comparison, the objective of machine unlearning is to prevent a trained model from leaking sensitive information about its training samples.
- **Online Learning.** Online learning [23] adjusts models quickly according to the data in a feedback process, such that the model can reflect online changes in a timely manner. One major difference between online learning and machine unlearning is that the former requires a merge operation to incorporate updates, while machine unlearning is an inverse operation that eliminates those updates when an unlearning request is received [24].
- **Catastrophic forgetting.** Catastrophic forgetting [25, 26] refers to a significant drop in performance on previously learned tasks when a model is fine-tuned for a new task. Catastrophic forgetting causes a deep network to lose accuracy, but the information of the data it uses may still be accessible by analyzing the weights [27], therefore, it does not satisfy the conditions required by machine unlearning.

When users revoke permissions over some training data, it is not sufficient to merely remove those data from the original training dataset, since the attackers can still reveal user information from the trained models [28]. One straightforward approach to perfectly removing information from the model is to retrain it from scratch (the retraining process in Figure 1). However, many complex models have been built on an enormous set of samples. Retraining is generally a computationally expensive process [29, 30]. Moreover, in some specific learning scenarios, such as federated learning [31, 32], the training dataset may not be accessible, and thus retraining cannot be conducted at all. Therefore, to reduce the computational cost and make machine unlearning possible in all circumstances, new techniques should be proposed (the unlearning process in Figure 1).

1.2 Contributions of This Survey

Machine unlearning has played an essential role in many applications [33, 34]. However, its implementation and verification strategies are still not fully explored. There are various concepts and multiple verification schemes in this field, and the boundary between machine unlearning and other techniques is vague. These phenomena motivate us to compile a comprehensive survey that summarizes, analyzes, and categorizes machine unlearning techniques. In this survey, we aim to find a clear way to present the ideas and concepts in machine unlearning, showing their characteristics and highlighting their advantage. In addition, we propose a novel taxonomy for classifying state-of-the-art literature. We hope this survey provides an in-depth overview to readers who wish to know this field, and it also serves as a stepping-stone for advancing innovations and widening research visions. The main contributions of this article are listed as follows:

- We proposed a novel taxonomy of current machine unlearning techniques based on their rationale and unlearning strategy.
- We comprehensively summarized state-of-the-art unlearning methods based on the proposed taxonomy, showing their benefits and shortcomings.
- We summarized the verification methods of machine unlearning within the taxonomy and reviewed their implementations with related unlearning techniques.
- We provided critical and deep discussions on the open issues in machine unlearning and pointed out possible further research directions.

1.3 Comparison to Existing Surveys in Machine Unlearning

There are some works that have been conducted to summarize machine unlearning. However, few of them provide deep and comprehensive insight into current research. Here, we introduce some relevant works for reference. Table 1 summarizes the comparison of those references.

- Thanh et al. [35] summarized the definitions of machine unlearning, the unlearning request types, and different designing requirements. They also provided a taxonomy of the existing unlearning schemes based on available models and data.
- Saurabh et al. [36] analyzed the problem of privacy leakage in machine learning and briefly described how the "right-to-be-forgotten" can be implemented with the potential approaches.
- Anvith et al. [37] discussed the semantics behind unlearning and reviewed existing unlearning schemes based on logits, weights, and weight distributions. They also briefly described partial validation schemes of machine unlearning.

In addition to the difference in Table 1, this survey also differs from the above references in several aspects. First, we provide a comprehensive analysis of each unlearning scheme together with corresponding verification strategies, since the verification problem is an important metric

Table 1. Comparison between Existing Machine Unlearning Surveys

Survey	Targets				Desiderata			Unlearning Request						Verification Methods						Open Questions			
	Exact	Approximate	Strong	Weak	Consistency	Accuracy	Verifiability	Sample	Class	Feature	Sequence	Graph	Client	Retraining-based	Attack-based	Accuracy-based	Relearn Time-based	Theory-based	Information bound-based	Universality	Security	Verification	Applications
Thanh et al. [35]	✓	✓	×	×	✓	✓	✓	✓	✓	✓	✓	✓	×	×	✓	✓	✓	×	×	×	×	✓	×
Saurabh et al. [36]	×	×	×	×	×	×	×	✓	✓	×	×	×	×	×	×	×	×	×	×	×	×	×	×
Anvith et al. [37]	✓	✓	×	×	✓	×	×	×	×	×	×	×	×	×	✓	×	×	×	✓	×	×	×	×
Ours	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

in future studies. This is the significant difference between the above reference, as existing works have only reviewed the unlearning schemes used in each work. Second, each unlearning scheme is reviewed and compared through several dimensions, such as whether original training data is required, whether intermediate data needs to be cached, which classes and models are supported for unlearning requests, and so on. In addition, we analyze the commonalities and problems within each category in our taxonomy scheme, summarizing the trends, shortcomings, and potential solutions, which have not been fully discussed in the above works [35–37].

Our work also involves multiple key areas of privacy preserving and optimization, covering topics of differential privacy, data masking, convex optimization, and so on. In contrast, existing surveys mainly focus on summarizing the methods employed in machine unlearning, ignoring the relationship between unlearning strategy and verification technique. The most similar work to ours is Reference [35], however, it elaborates more on the unlearning framework and its application scenario, while we particularly emphasize unlearning strategy and verification. Moreover, we explore the possible trends of machine unlearning and summarize the latest research progress and possible techniques involved, including universality, security, and so on, and suggest several specific research directions. Those are also not provided in the above references [35–37] in Table 1.

2 PRELIMINARIES

2.1 Definition of Machine Unlearning

Vectors are denoted as bold lowercase, e.g., $\mathbf{x}_i$, and space or set as italics in uppercase, e.g., $\mathcal{X}$. A general definition of machine learning is given based on a supervised learning setting. The instance space is defined as $\mathcal{X} \subseteq \mathbb{R}^d$, with the label space defined as $\mathcal{Y} \subseteq \mathbb{R}$. $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n \subseteq \mathbb{R}^d \times \mathbb{R}$ represents a training dataset, in which each sample $\mathbf{x}_i \in \mathcal{X}$ is a d-dimensional vector $(x_{i,j})_{j=1}^d$, $y_i \in \mathcal{Y}$ is the corresponding label, and n is the size of $\mathcal{D}$. Let d be the dimension of $\mathbf{x}_i$ and let $\mathbf{x}_{i,j}$ denote the jth feature in the sample $\mathbf{x}_i$.

The purpose of machine learning is to build a model M with the parameters $\mathbf{w} \in \mathcal{H}$ based on a specific training algorithm $\mathcal{A}(\cdot)$, where $\mathcal{H}$ is the hypothesis space for $\mathbf{w}$. In machine unlearning, let $\mathcal{D}_u \subset \mathcal{D}$ be a subset of the training dataset, whose influence we want to remove from the trained model. Let its complement $\mathcal{D}_r = \mathcal{D}_u^{\complement} = \mathcal{D}/\mathcal{D}_u$ be the dataset that we want to retain, and let $\mathcal{R}(\cdot)$ and $\mathcal{U}(\cdot)$ represent the retraining process and unlearning process, respectively. $\mathbf{w}_r$ and $\mathbf{w}_u$ donate the parameters of the built models from those two processes. $P(a)$ represents the distribution of a variable a, and $\mathcal{K}(\cdot)$ represents a measurement of the similarity of two distributions. When considering $\mathcal{K}(\cdot)$ as a **Kullback-Leibler (KL)** divergence, $\mathcal{K}(\cdot)$ is defined by $\mathrm{KL}(P(a)\|P(b)) :=$

Table 2. Notations

Notations	Explanation	Notations	Explanation
$\mathcal{X}$	The instance space	$\mathcal{Y}$	The label space
$\mathcal{D}$	The training dataset	$\mathcal{D}_r$	The remaining dataset
$\mathcal{D}_u$	The unlearning dataset	$\mathbf{x}_i$	One sample in $\mathcal{D}$
y_i	The label of sample $\mathbf{x}_i$	n	The size of $\mathcal{D}$
$\mathbf{x}_{i,j}$	The jth feature in $\mathbf{x}_i$	d	The dimension of $\mathbf{x}_i$
$\mathcal{A}(\cdot)$	The learning process	$\mathcal{U}(\cdot)$	The unlearning process
$\mathcal{R}(\cdot)$	The retraining process	$\mathbf{w}$	The parameters of learned model
$\mathbf{w}_u$	The parameters of unlearned model	$\mathbf{w}_r$	The parameters of retrained model
$P(\cdot)$	The distribution function	$\mathcal{K}(\cdot)$	The distribution measurement
$\mathcal{I}(\cdot)$	The Shannon Mutual Information	$\mathcal{H}$	The hypothesis space for $\mathbf{w}$

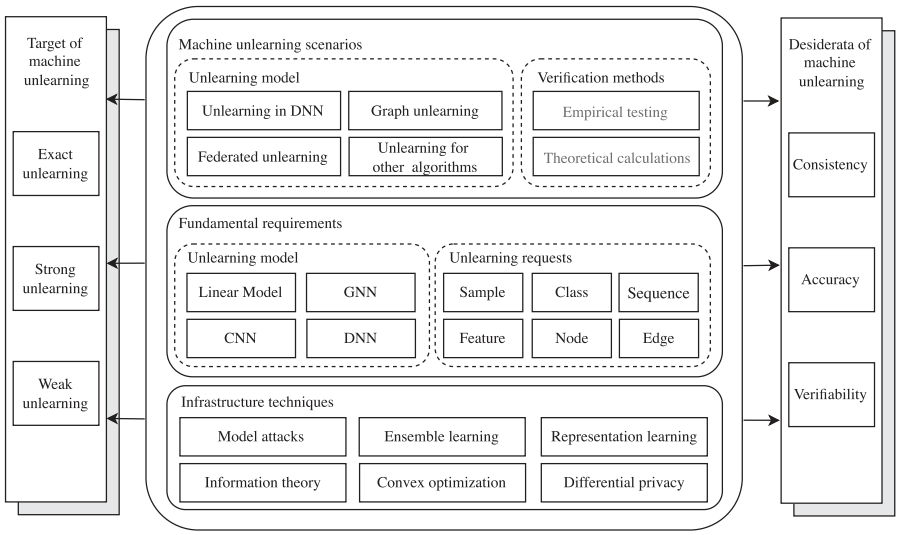

Fig. 2. Machine unlearning and its ecosystem.

$\mathbb{E}_{a\sim P(a)}[\log(P(a)/P(b))]$. Given two random variables a and b, the amount of Shannon Mutual Information that a has about b is defined as $I(a;b)$. The main notations are summarized in Table 2.

Now, we give the definition of machine unlearning.

Definition 2.1 (Machine Unlearning [29]). Consider a cluster of samples that we want to remove from the training dataset and the trained model, denoted as $\mathcal{D}_u$. An unlearning process $\mathcal{U}(\mathcal{A}(\mathcal{D}), \mathcal{D}, \mathcal{D}_u)$ is defined as a function from an trained model $\mathcal{A}(\mathcal{D})$, a training dataset $\mathcal{D}$, and an unlearning dataset $\mathcal{D}_u$ to a model $\mathbf{w}_u$, which ensures that the unlearned model $\mathbf{w}_u$ performs as though it had never seen the unlearning dataset $\mathcal{D}_u$.

Figure 2 presents the typical concept, unlearning targets, and desiderata associated with machine unlearning. The infrastructure techniques involved in machine unlearning include several aspects, such as ensemble learning, convex optimization, and so on [38]. These technologies provide robust guarantees for different foundational unlearning requirements that consist of various types of models and unlearning requests, resulting in diverse unlearning scenarios and corresponding verification methods. Additionally, to ensure effectiveness, the unlearning process requires different targets, such as exact unlearning or strong unlearning. Each unlearning target ensures

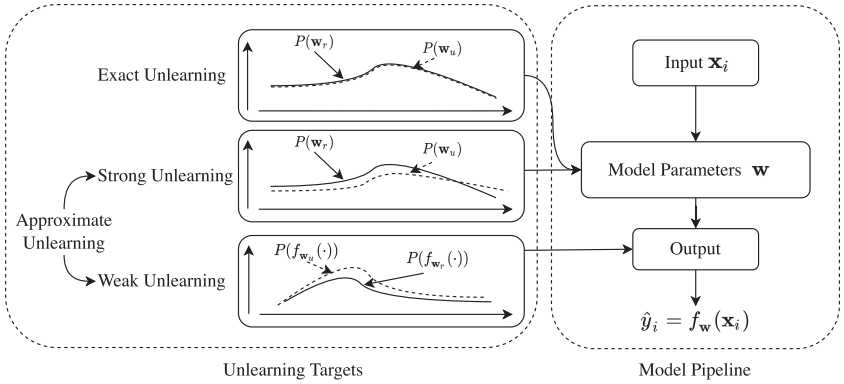

Fig. 3. Targets of machine unlearning.

different similarities in the distribution of the parameters between the unlearned model and that of the retrained model. Machine unlearning also involves several unlearning desiderata, including consistency, accuracy, and verifiability. Those desiderata, with the target constraint, simultaneously guarantee the validity and feasibility of each unlearning scheme.

2.2 Targets of Machine Unlearning

The ultimate target of machine unlearning is to reproduce a model that (1) behaves as if trained without seeing the unlearned data and (2) consumes as less time as possible. The performance baseline of an unlearned model is that of the model retrained from scratch (a.k.a., native retraining).

Definition 2.2 (Native Retraining [29]). Supposing the learning process, $\mathcal{A}(\cdot)$, never sees the unlearning dataset $\mathcal{D}_u$, and thereby performs a retraining process on the remaining dataset, denoted as $\mathcal{D}_r = \mathcal{D} \backslash \mathcal{D}_u$. In this manner, the retraining process is defined as:

$$\mathbf{w}_r = \mathcal{A}(\mathcal{D} \backslash \mathcal{D}_u). \tag{1}$$

The naive retraining naturally ensures that any information about samples can be unlearned from both the training dataset and the already-trained model. However, the computational and time overhead associated with the retraining process could be significantly expensive. Further, a retraining process is not always possible if the training dataset is inaccessible, such as federated learning [39]. Therefore, two alternative unlearning targets have been proposed: exact unlearning and approximate unlearning.

Exact unlearning guarantees that the distribution of an unlearned model and a retrained model are indistinguishable. In comparison, approximate unlearning mitigates the indistinguishability in weights and final activation, respectively. In practice, approximate unlearning further evolves to strong and weak unlearning strategies. Figure 3 illustrates the targets of machine unlearning and their relationship with a trained model. The different targets, in essence, correspond to the requirement of unlearning results.

Definition 2.3 (Exact Unlearning [40]). Given a distribution measurement $\mathcal{K}(\cdot)$, such as KL-divergence, the unlearning process $\mathcal{U}(\cdot)$ will provide an *exact unlearning* target if

$$\mathcal{K}(P(\mathcal{U}(\mathcal{A}(\mathcal{D}), \mathcal{D}, \mathcal{D}_u)), P(\mathcal{A}(\mathcal{D} \backslash \mathcal{D}_u))) = 0, \tag{2}$$

where $P(\cdot)$ denotes the distribution of the weights.

Table 3. Summary and Comparison of Difference between Targets

Tartgets	Aims	Advantages	Limitations
Exact Unlearning	To make the distributions of a natively retrained model and an unlearned model indistinguishable	Ensures that attackers cannot recover any information from the unlearned model	Difficult to implement
Strong Unlearning	To ensure that the distributions of two models are approximately indistinguishable	Easier to implement than exact unlearning	Attackers can still recover some information from the unlearned model
Weak Unlearning	To only ensure that the distributions of two final activations are indistinguishable	The easiest target for machine unlearning	Cannot guarantee whether the internal parameters of the model are successfully unlearned

Exact unlearning guarantees the two output distributions are indistinguishable, thus preventing an observer (e.g., attacker) to exact any information about $\mathcal{D}_u$.

However, a less strict unlearning target is necessary, because exact unlearning can only be achieved for simple and well-structured models [24]. As a result, approximate unlearning, which is suitable to complex machine learning models, is proposed.

Definition 2.4 (Approximate Unlearning [37]). If $\mathcal{K}(P(\mathcal{U}(\mathcal{A}(\mathcal{D}), \mathcal{D}, \mathcal{D}_u)), P(\mathcal{A}(\mathcal{D}\backslash\mathcal{D}_u)))$ is limited within a tolerable threshold, then the unlearning process $\mathcal{U}(\cdot)$ is defined as strong unlearning.

Approximate unlearning ensures that the distribution of the unlearned model and that of a retrained model are approximately indistinguishable. This approximation is usually guaranteed by differential privacy techniques, such as (ε, δ)-certified unlearning [41, 42].

Depending on how the distribution is estimated, approximate unlearning can be further classified into *strong unlearning* and *weak unlearning*. Strong unlearning is established based on the similarity between the internal parameter distributions of the models, while weak unlearning is based on the distribution of the model's final activation results [42, 43].

Table 3 summarizes the main differences between each unlearning target.

2.3 Desiderata of Machine Unlearning

To fairly and accurately assess the efficiency and effectiveness of unlearning approaches, there are some mathematical properties that can be used for evaluation.

Definition 2.5 (Consistency). Assume there is a set of samples X_e, with the true labels $Y_e : \{y_1^e, y_2^e, \ldots, y_n^e\}$. Let $Y_n : \{y_1^n, y_2^n, \ldots, y_n^n\}$ and $Y_u : \{y_1^u, y_2^u, \ldots, y_n^u\}$ be the predicted labels produced from a retrained model and an unlearned model, respectively. If all $y_i^n = y_i^u, 1 \leq i \leq n$, then the unlearning process $\mathcal{U}(\mathcal{A}(\mathcal{D}), \mathcal{D}, \mathcal{D}_u)$ is considered to provide the consistency property.

Consistency denotes how similar the behavior of a retrained model and an unlearned model is. It represents whether the unlearning strategy can effectively remove all the information of the unlearning dataset $\mathcal{D}_u$. If, for every sample, the unlearned model gives the same prediction result as the retrained model, then an attacker has no way to infer information about the unlearned data.

Definition 2.6 (Accuracy). Given a set of samples X_e in remaining dataset, where their true labels are $Y_e : \{y_1^e, y_2^e, \ldots, y_n^e\}$. Let $Y_u : \{y_1^u, y_2^u, \ldots, y_n^u\}$ to denote the predicted labels produced by the model after the unlearning process, $\mathbf{w}_u = \mathcal{U}(\mathcal{A}(\mathcal{D}), \mathcal{D}, \mathcal{D}_u)$. The unlearning process is considered to provide the accuracy property if all $y_i^u = y_i^e, 1 \leq i \leq n$.

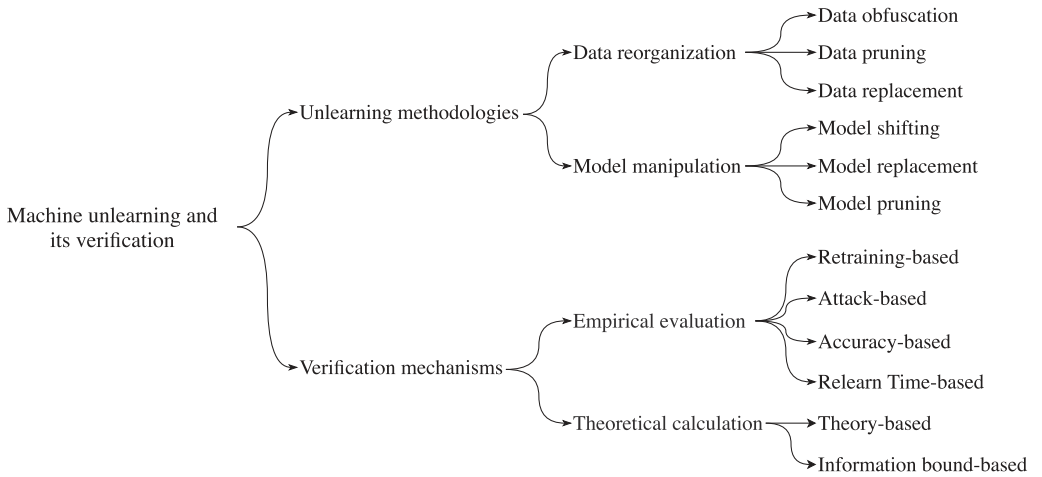

Fig. 4. Taxonomy of unlearning and verification mechanisms.

Accuracy refers to the ability of the unlearned model to predict samples correctly. It reveals the usability of a model after the unlearning process, given that a model with low accuracy is useless in practice. Accuracy is a key component of any unlearning mechanism, as we claim the unlearning mechanism is ineffective if the process significantly undermines the original model's accuracy.

Definition 2.7 (Verifiability). After the unlearning process, a verification function $\mathcal{V}(\cdot)$ can make a distinguishable check, that is, $\mathcal{V}(\mathcal{A}(\mathcal{D})) \neq \mathcal{V}(\mathcal{U}(\mathcal{A}(\mathcal{D}), \mathcal{D}, \mathcal{D}_u))$. The unlearning process $\mathcal{U}(\mathcal{A}(\mathcal{D}), \mathcal{D}, \mathcal{D}_u)$ can then provide a verifiability property.

Verifiability can be used to measure whether a model provider has successfully unlearned the requested unlearning dataset $\mathcal{D}_u$. Taking the following backdoor verification method as an example [44], if the pre-injected backdoor for an unlearned sample $\mathbf{x}_d$ is verified as existing in $\mathcal{A}(\mathcal{D})$ but not $\mathcal{U}(\mathcal{A}(\mathcal{D}), \mathcal{D}, \mathcal{D}_u)$, that is $\mathcal{V}(\mathcal{A}(\mathcal{D})) = true$ and $\mathcal{V}(\mathcal{U}(\mathcal{A}(\mathcal{D}), \mathcal{D}, \mathcal{D}_u)) = false$, then the unlearning method $\mathcal{U}(\mathcal{A}(\mathcal{D}), \mathcal{D}, \mathcal{D}_u)$ can be deemed to provide verifiability property.

3 TAXONOMY OF UNLEARNING AND VERIFICATION MECHANISMS

Figure 4 summarizes the general taxonomy of machine unlearning and its verification used in this article. The taxonomy is inspired by the design details of the unlearning strategy. Unlearning approaches that concentrate on modifying the training data are classified in data reorganization, while methods that directly manipulate the weights of a trained model are denoted as model manipulation. As for verification methods, initially, we categorize those schemes as either experimental or theoretical; subsequently, we summarize these methods based on the metrics they use.

3.1 Unlearning Taxonomy

3.1.1 Data Reorganization. Data reorganization refers to the technique that a model provider unlearns data by reorganizing the training dataset. It mainly includes three different processing methods according to the different data reorganization modes: *obfuscation*, *pruning*, and *replacement* [30, 45]. Table 4 compares and summarizes the differences between these schemes.

- Data obfuscation: In data obfuscation, model providers intentionally add some choreographed data to the remaining dataset, that is, $\mathcal{D}_{new} \leftarrow \mathcal{D}_r \cup \mathcal{D}_{obf}$, where $\mathcal{D}_{new}$ and $\mathcal{D}_{obf}$ are the new training dataset and the choreographed data, respectively. The trained model is

then fine-tuned based on $\mathcal{D}_{new}$ to unlearn some specific samples. Such methods are usually based on the idea of erasing information about $\mathcal{D}_u$ by recombining the dataset with choreographed data. For example, Graves et al. [45] relabeled $\mathcal{D}_u$ with randomly selected incorrect labels and then fine-tuned the trained model for several iterations for unlearning data.

- Data pruning: In data pruning, the model provider first segments the training dataset into several sub-datasets and trains several sub-models based on each sub-dataset. Those sub-models are then used to aggregate a consensus prediction collaboratively, that is, $\mathcal{D} \rightarrow \mathcal{D}_1 \cup \mathcal{D}_2 \cup \cdots \cup \mathcal{D}_m$, $\mathbf{w}_i = \mathcal{A}(\mathcal{D}_i)$ and $f(\mathbf{x}) = Agg(M_{\mathbf{w}_i}(\mathbf{x}))$, where $\mathcal{D}_i$, $0 < i < m$ are the sub-datasets, and $\cap\mathcal{D}_i = \emptyset$, $\cup\mathcal{D}_i = \mathcal{D}$, m is the number of sub-dataset, $\mathbf{w}_i$ is the sub-model, and $Agg(\cdot)$ is the aggregation function. After an unlearning request arrives, the model provider deletes the unlearned samples from the sub-datasets that contain them and then retrains the affected sub-models. The flexibility of this methodology is that the influence of unlearning dataset $\mathcal{D}_u$ is limited to each sub-dataset after segmentation rather than the whole dataset. Taking the **SISA** scheme in Reference [30] as an example, the **SISA** framework first randomly divided the training dataset into k shards. A series of models are then trained separately at one per shard. When a sample needs to be unlearned, it is first removed from the shards that contain it, and only the sub-models corresponding to those shards are retrained.
- Data replacement: In data replacement, the model provider deliberately replaces the training dataset $\mathcal{D}$ with some new transformed dataset, that is, $\mathcal{D}_{trans} \leftarrow \mathcal{D}$. The transformed dataset $\mathcal{D}_{trans}$ is then used to train a model that makes it easy to implement unlearning after receiving an unlearning request. For example, Cao et al. [29] replaced the training dataset with several efficiently computable transformations and used those transformations to complete the training of the model. Those transformations can be updated much more quickly after removing any samples from the transformed dataset. Consequently, computational overheads are reduced, and unlearning operations are more efficient.

3.1.2 Model Manipulation. In model manipulation, the model provider aims to realize unlearning operations by adjusting the model's parameters. Model manipulation mainly includes the following three categories. Table 4 compares and summarizes the differences between these schemes.

- Model shifting: In model shifting, the model providers directly update the model parameters to offset the impact of unlearned samples on the model, that is, $\mathbf{w}_u = \mathbf{w} + \delta$, where $\mathbf{w}$ are parameters of the originally trained model, and δ is the updated value. These methods are usually based on the idea of calculating the influence of samples on the model parameters and then updating the model parameters to remove that influence. It is usually extremely difficult to accurately calculate a sample's influence on a model's parameters, especially with complex deep neural models. Therefore, many model shifting-based unlearning schemes are based on specific assumptions. For example, Guo et al.'s [41] unlearning algorithms are designed for linear models with strongly convex regularization.
- Model replacement: In model replacement, the model provider directly replaces some parameters with pre-calculated parameters, that is, $\mathbf{w}_u \leftarrow \mathbf{w}_{noeffect} \cup \mathbf{w}_{pre}$, where $\mathbf{w}_u$ are parameters of the unlearned model, $\mathbf{w}_{noeffect}$ are partially unaffected static parameters, and $\mathbf{w}_{pre}$ are the pre-calculated parameters. These methods usually depend on a specific model structure to predict and calculate the affected parameters in advance. They are only suitable for some special machine learning models, such as decision trees or random forest models. Taking the method in Reference [57] as an example, the affected intermediate decision nodes are replaced based on pre-calculated decision nodes to generate an unlearned model.
- Model pruning: In model pruning, the model provider prunes some parameters from the trained models to unlearn the given samples, that is, $\mathbf{w}_u \leftarrow \mathbf{w}/\delta$, where $\mathbf{w}_u$ are the

Table 4. Summary and Comparison of Differences between Unlearning Schemes

	Schemes	Basic Ideas	Advantages	Limitations
Data Reorganization	Data Obfuscation [27, 45, 46]	Intentionally adds some choreographed dataset to the training dataset and retrains the model	Can be applied to almost all types of models; not too much intermediate redundant data need to be retained	Not easy to completely unlearn information from models
	Data Pruning [29, 30, 47] [48–50]	Deletes the unlearned samples from sub-datasets that contain those unlearned samples. Then only retrains the sub-models that are affected by those samples	Easy to implement and understand; completes the unlearning process at a faster speed	Additional storage space is required; accuracy can be decreased with an increase in the number of sub-datasets
	Data Replacement [29]	Deliberately replaces the training dataset with some new transformed dataset	Supports completely unlearn information from models; easy to implement	Hard to retain all the information about the original dataset through replacement
Model Manipulation	Model Shifting [24, 45] [40, 41, 51] [42, 43, 52, 53]	Directly updates model parameters to offset the impact of unlearned samples on the model	Does not require too much intermediate parameter storage; can provide theoretical verification	Not easy to find an appropriate offset value for complex models; calculating offset value is usually complex
	Model Pruning [54–56]	Replaces partial parameters with pre-calculated parameters	Reduces the cost caused by intermediate storage; the unlearning process can be completed at a faster speed	Only applicable to partial models; not easy to implement and understand
	Model Replacement [57–60]	Prunes some parameters from already-trained models	Easy to completely unlearn information from models	Only applicable to partial machine learning models; original model structure is usually changed

parameters of the unlearned model, $\mathbf{w}$ are the parameters of the trained model, and δ are the parameters that need to be removed. Such unlearning schemes are also usually based on specific model structures and are generally accompanied by a fine-tuning process to recover performance after the model is pruned. For example, Wang et al. [55] introduced the **term frequency-inverse document frequency (TF-IDF)** to quantize the class discrimination of channels in a convolutional neural network model, where channels with high TF-IDF scores are pruned.

3.2 Verification Mechanisms

Verifying whether the unlearning method has the verifiability property is not an easy task. Model providers may claim externally that they remove those influences from their models, but, in reality, this is not the case [48]. For data providers, proving that the model provider has completed the unlearning process may also be tricky, especially for complex deep models with huge training datasets. Removing a small portion of samples only causes a negligible effect on the model. Moreover, even if the unlearned samples have indeed been removed, the model still has a great chance of making a correct prediction, since other users may have provided similar samples. Therefore, providing a reasonable unlearning verification mechanism is a topic worthy of further research.

3.2.1 Empirical Evaluation.

- **Retraining-based verification:** Retraining can naturally provide a verifiability property, since the retraining dataset no longer contains the samples that need to be unlearned. This is the most intuitive and easy-to-understand solution.
- **Attack-based verification:** The essential purpose of an unlearning operation is to reduce leaks of sensitive information caused by model over-fitting. Hence, some attack methods can directly and effectively verify unlearning operations—for example, membership inference attacks [5] and model inversion attacks [4]. In addition, Sommer et al. [44] provided a novel backdoor verification mechanism from an individual user perspective in the context of **machine learning as a service (MLaaS)** [61]. This approach can verify, with high confidence, whether the service provider complies with the user's right to unlearn information.
- **Relearning time-based verification:** Relearning time can be used to measure the amount of information remaining in the model about the unlearned samples. If the model quickly recovers performance as the original trained model with little retraining time, then it is likely to still remember some information about the unlearned samples [27].
- **Accuracy-based verification:** A trained model usually has high prediction accuracy for the samples in the training dataset. This means the unlearning process can be verified by the accuracy of a model's output. For the data that need to be unlearned, the accuracy should ideally be the same as a model trained without seeing $\mathcal{D}_u$ [40]. In addition, if a model's accuracy after being attacked can be restored after unlearning the adversarial data, then we can also claim that the unlearning is verified.

3.2.2 Theoretical Calculation.

- **Theory-based verification:** Some methods provide a certified unlearning definition [41, 53], which ensures that the unlearned model cannot be distinguished from a model trained on the remaining dataset from scratch. This could also provide a verification method that directly guarantees the proposed schemes can unlearn samples.
- **Information bound-based verification:** Golatkar et al. [40, 43] devised a new metric for verifying the effectiveness of unlearning schemes, where they measured the upper bound of the residual information about samples that need to be unlearned. Less residual information represents a more effective unlearning operation.

Table 5 summarizes and compares each verification method's advantages and limitations.

4 DATA REORGANIZATION

In this section, we review how data reorganization methods support the unlearning process. Since proving the verifiability property of unlearning algorithms is also important and should be considered in machine unlearning research, we separately discuss it for each unlearning method.

4.1 Reorganization Based on Data Obfuscation

4.1.1 Unlearning Schemes Based on Data Obfuscation. In general, the majority of model attack scenarios, such as membership inference attacks, arise from model overfitting and rely on observing shifts in the output based on known input shifts [62]. That is, for the vast majority of attackers, it is easy to perform an attack on some trained models by observing the shifts of the output confidence vectors. One optional machine unlearning scheme can be interpreted as confusing the model's understanding of samples so it cannot retain any correct information within models. This method can further confuse the confidence vector of the model's output [46]. As shown in Figure 5,

Table 5. Summary and Comparison of Different Verification Methods

Methods	Basic Ideas	Advantages	Limitations
Retraining-based	Removes unlearned samples and retrains models	Intuitive and easy to understand	Only applicable to special unlearning schemes
Attack-based	Based on membership inference attacks or model inversion attacks	Intuitively measures the defense effect against some attacks	Inadequate verification capability
Relearn time-based	Measures the time when the unlearned model regains performance on unlearned samples	Easy to understand and easy to implement	Inadequate verification capability
Accuracy-based	Same as a model trained without unlearned samples	Easy to understand and easy to implement	Inadequate verification capability
Theory-based	Ensures similarity between the unlearned model and the retrained model.	Comprehensive and has theoretical support	Implementation is complex and only applies to some specified models
Information bound-based	Measures the upper-bound of the residual information about the unlearned samples	Comprehensive and has theoretical support	Hard to implement and only applicable to some specified models

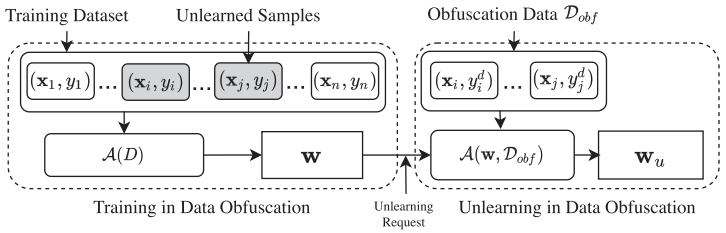

Fig. 5. Unlearning schemes based on data obfuscation.

when receiving an unlearning request, the model continues to train $\mathbf{w}$ based on the constructed obfuscation data $\mathcal{D}_{obf}$ giving rise to an updated $\mathbf{w}_u$.

In this vein, Graves et al. [45] proposed a random *relabel* and *retraining* machine unlearning framework. Sensitive samples are relabeled with randomly selected incorrect labels, and then the machine learning model is fine-tuned based on the modified dataset for several iterations to unlearn those specific samples. Similarly, Felps et al. [46] intentionally poisoned the labels of the unlearning dataset and then fine-tuned the model based on the new poisoned dataset. However, such unlearning schemes only confuse the relationship between the model outputs and the samples; the model parameters may still contain information about each sample.

The trained model is always trained by minimizing the loss for all classes. If one can learn a kind of noise that only maximizes the loss for some classes, then those classes can be unlearned. Based on this idea, Tarrun et al. [27] divided the unlearning process into two steps, *impair* and *repair*. In the first step, an error-maximizing noise matrix is learned that consists of highly influential samples corresponding to the unlearning class. The effect of the noise matrix is somehow the opposite of the unlearning data and can destroy the information of unlearned data to unlearn single/multiple classes. To repair the performance degradation caused by the model unlearning process, the *repair* step further adjusted the model based on the remaining data.

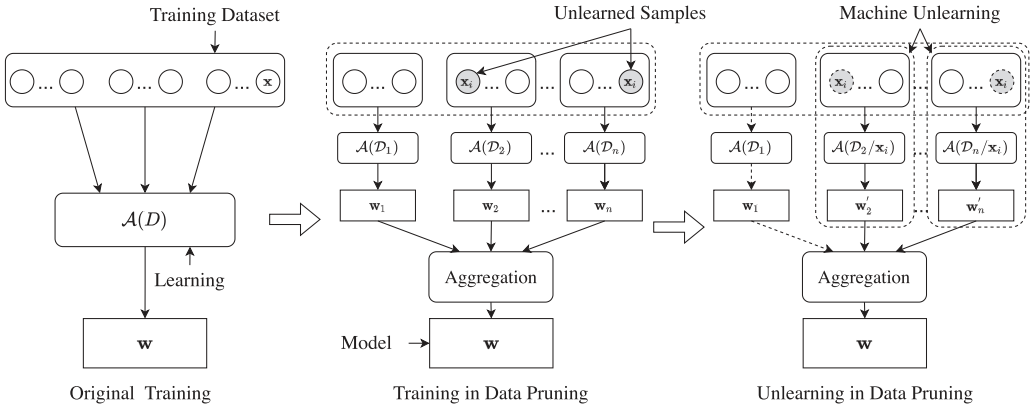

Fig. 6. Unlearning schemes based on data pruning.

Similarly, Zhang et al. [63] considered the unlearning request in the image retrieval field. The approach developed involves creating noisy data using a generative method to adjust the weights of the retrieval model and achieve the unlearning purposes. They also proposed a new learning framework, which includes both static and dynamic learning branches, ensuring that the generated noisy data only affects the unlearning data being forgotten without affecting the contribution of other remaining data. However, the above two schemes consume more time to generate noise for unlearning process, which will affect the efficiency of the unlearning process [27, 63].

4.1.2 Verifiability of Schemes Based on Data Obfuscation. To verify their unlearning process, Graves et al. [45] used two state-of-the-art attack methods—a model inversion attack and a membership inference attack—to evaluate how much information was retained in the model parameters about specific samples after the unlearning process—in other words, how much information might be leaked after the unlearning process. Their model inversion attack is a modified version of the standard model inversion attack proposed by Fredrikson et al. [6]. The three modifications include: adjusting the process function to every n gradient descent steps; adding a small amount of noise to each feature before each inversion; and modifying the number of attack iterations performed. These adjustments allowed them to analyze complex models. For the membership inference attack, they used the method outlined by Yeom et al. in Reference [64]. Felps et al.'s verifiability analysis is also based on the membership inference attack [46].

In comparison, Tarrun et al. [27] evaluated the verifiability through several measurements. They first assessed relearning time by measuring the number of epochs for the unlearned model to reach the same accuracy as the originally trained model. Then, the distance between the original model, the model after the unlearning process, and the retrained model are further evaluated.

4.2 Reorganization Based on Data Pruning

4.2.1 Unlearning Schemes Based on Data Pruning. As shown in Figure 6, unlearning schemes based on data pruning are usually based on ensemble learning techniques. Bourtoule et al. [30] proposed a "**sharded, isolated, sliced, and aggregated**" (**SISA**) framework, similar to the current distributed training strategies [65, 66], as a method of machine unlearning. With this approach, the training dataset $\mathcal{D}$ is first partitioned into k disjoint shards $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_k$. Then, sub-models $\mathcal{M}_w^1, \mathcal{M}_w^2, \ldots, \mathcal{M}_w^k$ are trained in isolation on each of these shards, which limits the influence of the samples to sub-models that were trained on the shards containing those samples. At inference time, k individual predictions from each sub-model are simply aggregated to provide a global prediction

(e.g., with majority voting), similar to the case of machine learning ensembles [67]. When the model owner receives a request to unlearn a data sample, they just need to retrain the sub-models whose shards contain that sample.

As the amount of unlearning data increases, **SISA** will cause degradation in model performance, making them only suitable for small-scale scenarios. The cost of these unlearning schemes is the time required to retrain the affected sub-models, which directly relates to the size of the shard. The smaller the shard, the lower the cost of the unlearning scheme. At the same time, there is less training dataset for each sub-model, which will indirectly degrade the ensemble model's accuracy. Bourtoule et al. [30] provided three key technologies to alleviate this problem, including *unlearning in the absence of isolation*, *data replication*, and *core-set selection*.

In addition to this scheme, Chen et al. [33] introduced the method developed in Reference [30] to recommendation systems and designed three novel data partition algorithms to divide the recommendation training data into balanced groups to ensure that collaborative information was retained. Wei et al. [68] focused on the unlearning problems in patient similarity learning and proposed *PatEraser*. To maintain the comparison information between patients, they developed a new data partition strategy that groups patients with similar characteristics into multiple shards. Additionally, they proposed a novel aggregation strategy to improve the global model utility.

Yan et al. [69] designed an efficient architecture for exact machine unlearning called *ARCANE*, similar to the scheme in Bourtoule et al. [30]. Instead of dividing the dataset uniformly, they split it by class and utilized the one-class classifier to reduce the accuracy loss. Additionally, they preprocessed each sub-dataset to speed up model retraining, which involved representative data selection, model training state saving, and data sorting by erasure probability. Nevertheless, the above unlearning schemes [30, 33, 69] usually need to cache a large number of intermediate results to complete the unlearning process. This will consume a lot of storage space.

SISA is designed to analyze Euclidean space data, such as images and text, rather than non-Euclidean space data, such as graphs. By now, numerous important real-world datasets are represented in the form of graphs, such as social networks [70], financial networks [71], biological networks [72], or transportation networks [73]. To analyze the rich information in these graphs, **graph neural networks (GNNs)** have shown unprecedented advantages [74, 75]. GNNs rely on the graph's structural information and neighboring node features. Yet, naively applying **SISA** scheme to GNNs for unlearning, i.e., randomly partitioning the training dataset into multiple sub-graphs, will destroy the training graph's structure and may severely damage the model's utility.

To allow efficient retraining while keeping the structural information of the graph dataset, Chen et al. [47] proposed *GraphEraser*, a novel machine unlearning scheme tailored to graph data. They first defined two common machine unlearning requests in graph scenario—node unlearning and edge unlearning—and proposed a general pipeline for graph unlearning, which is composed of three main steps: *graph partitioning*, *shard model training*, and *shard model aggravation*. In the *graph partitioning* step, they introduced an improved balanced ***label propagation algorithm* (LPA)** [76] and a balanced *embedding k-means* [77] partitioning strategy to avoid highly unbalanced shard sizes. Given that the different sub-models might provide different contributions to the final prediction, they also proposed a learning-based aggregation method, *OptAggr*, that optimizes the importance score of each sub-model to improve global model utility ultimately.

Deterministic unlearning schemes, such as **SISA** [30] or *GraphEraser* [47], promise nothing about what can be learned about specific samples from the difference between a trained model and an unlearned model. This could exacerbate user privacy issues if an attacker has access to the model before and after the unlearning operation [78]. To avoid this situation, an effective approach is to hide the information about the unlearned model when performing the unlearning operation.

In practical applications, Neel et al. [50] proposed an update-based unlearning method that performs several gradient descent updates to build an unlearned model. The method is designed to handle arbitrarily long sequences of unlearning requests with stable runtime and steady-state errors. In addition, to alleviate the above unlearning problem, they introduced the concept of *secret state*: An unlearning operation is first performed on the trained model. Then, the unlearned models are perturbed by adding Gaussian noise for publication. This effectively ensures that an attacker cannot access the unlearned model actually after the unlearning operation, which effectively hides any sensitive information in the unlearned model. They also provided an (ϵ, δ)-certified unlearning guarantee and leveraged a distributed optimization algorithm and reservoir sampling to grant improved accuracy/runtime tradeoffs for sufficiently high dimensional data.

After the initial model deployment, data providers may make an adaptive unlearning decision. For example, when a security researcher releases a new model attack method that identifies a specific subset of the training dataset, the owners of these subsets may rapidly increase the number of deletion requests. Gupta et al. [49] define the above unlearning requests as adaptive requests and propose an adaptive sequential machine unlearning method using a variant of the **SISA** framework [30] as well as a differentially private aggregation method [79]. They give a general reduction of the unlearning guarantees from the adaptive sequences to the non-adaptive sequences using differential privacy and max-information theory [80]. A strong provable unlearning guarantee for adaptive unlearning sequences is also provided, combined with the previous works of non-adaptive guarantees for sequence unlearning requests.

He et al. [48] developed an unlearning approach for the deep learning model. They first introduce a process called *detrended fluctuation analysis* [81], which quantifies the influence of the unlearned data on the model parameters, termed *temporal residual memory*. They observed that this influence is subject to exponential decay, which fades at an increasing rate over time. Based on these results, intermediate models are retained during the training process and divided into four areas, named *unseen*, *deleted*, *affected,* and *unaffected*. *Unseen* indicates that the unlearned sample has not yet arrived. *Deleted* includes the unlearning dataset. *Unaffected* and *affected* indicate whether temporal residual memory has lapsed or not. An unlearned model can be stitched by reusing the *unseen* and *unaffected* models and retraining the *affected* areas. However, this scheme does not provide any theoretical verification methods to ensure that the information about unlearning data to be unlearned is indeed removed from the model.

4.2.2 Verifiability of Schemes Based on Data Pruning. The unlearning schemes proposed in References [29, 30, 33, 47, 68, 69] are essentially based on a retraining mechanism that naturally has a verifiability property. As discussed in Section 2.3, a straightforward way to give an unlearning scheme the verifiability property is to retrain the model from scratch after removing the samples that need to be unlearned from the training dataset. The above schemes introduce distributed and ensemble learning techniques, which train sub-models separately and independently to optimize the loss function on each sub-dataset. The sub-models are then aggregated to make predictions. In terms of the unlearning process, only the affected sub-models are retrained, which avoids a large computational and time overhead and also provides a verifiability guarantee.

He et al. [48] use a backdoor verification method in Reference [44] to verify their unlearning process. They designed a specially crafted trigger and implanted this "backdoor data" in the samples that need to be unlearned, with little effect on the model's accuracy. They indirectly verify the validity of the unlearning process based on whether the backdoor data can be used to attack the unlearned model with a high success rate. If the attack result has lower accuracy, then it proves that the proposed unlearning method has removed the unlearned data. The other studies [49, 50] did not provide a method for verifying the unlearning process.

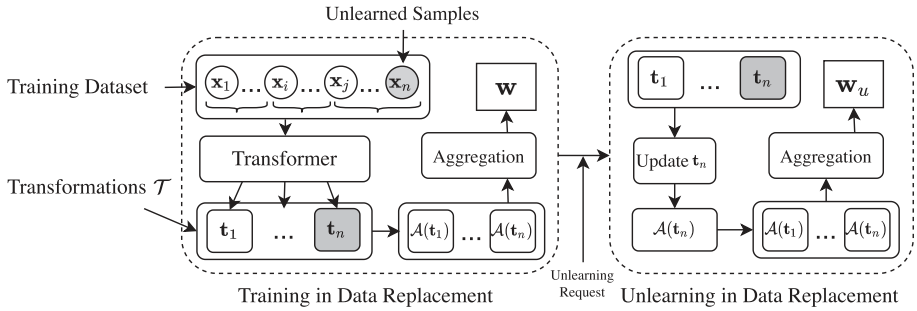

Fig. 7. Unlearning schemes based on data replacement.

4.3 Reorganization Based on Data Replacement

4.3.1 Unlearning Schemes Based on Data Replacement. As shown in Figure 7, when training a model in a data replacement scheme, the first step is usually to transform the training dataset into an easily unlearned type, named transformation $\mathcal{T}$. Those transformations are then used to separately train models. When an unlearning request arrives, only a portion of the transformations $\mathbf{t}_i$—the ones that contain the unlearned samples—need to be updated and used to retrain each sub-model to complete the machine unlearning.

Inspired by the previous work of using MapReduce to accelerate machine learning algorithms [82], Cao et al. [29] proposed a machine unlearning method that transforms the training dataset into summation form. Each summation is the sum of some efficiently computable transformation. The learning algorithms depend only on the summations, not the individual data, which breaks down the dependencies in the training dataset. To unlearn a data sample, the model provider only needs to update the summations affected by this sample and recompute the model. However, since the summation form comes from **statistical query (SQ)** learning, and only a few machine learning algorithms can be implemented as SQ learning, such as naïve bayes classifiers [83], support vector machines [84], and k-means clustering [85], this scheme has low applicability.

Takashi et al. [86] proposed a novel approach to lifelong learning named "Learning with Selective Forgetting," which involves updating a model for a new task by only forgetting specific classes from previous tasks while keeping the rest. To achieve this, the authors designed specific mnemonic codes, which are class-specific synthetic signals that are added to all the training samples of corresponding classes. Then, exploiting the mechanism of catastrophic forgetting, these codes were used to forget particular classes without requiring the original data. It is worth noting, however, that this scheme lacks any theoretical verification methods to confirm that the unlearning data information has been successfully removed from the model.

4.3.2 Verifiability of Schemes Based on Data Replacement. Cao et al. [29] provide an accuracy-based verification method. Specifically, they attack the LensKit model with the system inference attack method proposed by Calandrino et al. [87] and verify that the unlearning operations successfully prevent the attack from yielding any information. For the other three models, they first performed data pollution attacks to influence the accuracy of those models. They then analyzed whether the model's performance after the unlearning process was restored to the same state as before the pollution attacks. If the unlearned model was actually restored to its pre-pollution value, then the unlearning operation was considered to be successful. Takashi et al. [86] provided a new metric, named ***Learning with Selective Forgetting Measure (LSFM)***, that is based on the idea of accuracy.

4.4 Summary of Data Reorganization

In these last few subsections, we reviewed the studies that use data obfuscation, data pruning, and data replacement techniques as unlearning methods. A summary of the surveyed studies is shown in Table 6, where we present the key differences between each paper.

From those summaries, we can see that most unlearning algorithms retain intermediate parameters and make use of the original training dataset [30, 47]. This is because those schemes usually segment the original training dataset and retrain the sub-models that were trained on the segments containing those unlearned samples. Consequently, the influence of specific samples is limited to only some of the sub-models and, in turn, the time taken to actually unlearn the samples is reduced. However, segmenting decreases time at the cost of additional storage. Thus, it would be well worth researching more efficient unlearning mechanisms that ensure the validity of the unlearning process and do not add too many storage costs simultaneously.

Moreover, these unlearning schemes usually support various unlearning requests and models, ranging from samples to classes or sequences and from support vector machines to complex deep neural models [29, 47, 50]. Unlearning schemes based on data reorganization rarely operate on the model directly. Instead, they achieve the unlearning purpose by modifying the distribution of the original training datasets and indirectly changing the obtained model. The benefit is that such techniques can be applied to more complex machine learning models. In addition to their high applicability, most of them can provide a strong unlearning guarantee, that is, the distribution of the unlearned model is approximately indistinguishable to that obtained by retraining.

It is worth pointing out that unlearning methods based on data reorganization will affect the consistency and the accuracy of the model as the unlearning process continues [30, 47, 48]. This reduction in accuracy stems from the fact that each sub-model is trained on the part of the dataset rather than the entire training dataset. This phenomenon does not guarantee that the accuracy of the unlearned model is the same as the result before the segmentation. Potential solutions are *to use unlearning in the absence of isolation*, *data replication* [30].

Some of the studies mentioned indirectly verify the unlearning process using a retraining method [30, 47], while others provide verifiability through attack-based or accuracy-based methods [27, 45, 46]. However, most unlearning schemes do not present further investigations at the theoretical level. The vast majority of the above unlearning schemes verify validity through experiments, with no support for the theoretical validity of the schemes. Theoretical validity would show, for example, how much sensitive information attackers can glean from an unlearned model after unlearning process or how similar the parameters of the unlearned model are to the retrained model. Further theoretical research into the validity of unlearning schemes is therefore required.

In summary, when faced with unlearning requests for complex models, unlearning schemes based on data obfuscation seldom unlearn information. This is because it is difficult to offset the influence of the unlearning data completely. Data pruning schemes always affect the model's accuracy, since they usually train sub-models using a partial training dataset. For data replacement schemes, it is impossible to find a new dataset that can replace all the information within an original dataset to train a model. Thus, researchers should turn to design unlearning schemes that strike more of a balance between the effectiveness of the unlearning process and model usability.

5 MODEL MANIPULATION

The model training stage involves creating an effective model replicating the expected relationship between the inputs in the training dataset and the model's outputs. Thus, manipulating the model directly to remove specific relationships may be a good way to unlearn samples. In this section, we comprehensively review the state-of-the-art studies on unlearning through model manipulation. Again, the verification techniques are discussed separately for each category.

Table 6. The Surveyed Studies that Employed Data Reorganization Techniques for Unlearning Process

Papers	Unlearning Methods	Unlearning Target	Training Dataset	Intermediates	Unlearned Samples' Type	Target Models' Type	Consistency	Accuracy	Verifiability
Graves et al. [45]	Data Obfuscation	Strong unlearning	Yes	No	Samples or Class	DNN	No	No	Attack-based
Felps et al. [46]	Data Obfuscation	Strong unlearning	No	No	Sequences	DNN	No	No	Attack-based
Tarrun et al. [27]	Data Obfuscation	Strong unlearning	Yes	No	Classes	DNN	No	No	Accuracy-based and Retrain Time-based
Zhang et al. [63]	Data Obfuscation	Strong unlearning	No	No	Samples	DNN	No	No	Accuracy-based and Retrain Time-based
Bourtoule et al. [30]	Data Pruning	Strong unlearning	Yes	Yes	Batches and Sequences	DNN	No	No	Retrain-based
Chen et al. [33]	Data Pruning	Strong unlearning	Yes	Yes	Samples	DNN	No	No	Retrain-based
Wei et al. [68]	Data Pruning	Strong unlearning	Yes	Yes	Samples	DNN	No	No	Retrain-based
Yan et al. [69]	Data Pruning	Exact unlearning	Yes	Yes	Samples	DNN	Yes	Yes	Retrain-based
Chen et al. [47]	Data Pruning	Exact unlearning	Yes	Yes	Nodes and Edges	GNN	No	No	Retrain-based
Neel et al. [50]	Data Pruning	Strong unlearning	Yes	Yes	Non-adaptive Sequences	Convex Model	No	No	Retrain-based
Gupta et al. [49]	Data Pruning	Strong unlearning	Yes	Yes	Adaptive Sequences	Non-convex Models	No	No	Retrain-based
He et al. [48]	Data Pruning	Strong unlearning	Yes	Yes	Samples	DNN	No	No	Attack-based
Cao et al. [29]	Data Replacement	Exact Unlearning	Yes	Yes	Samples	Statistical Query Learning Models	Yes	Yes	Retrain-based and Accuracy-based
Takashi et al. [86]	Data Replacement	Strong unlearning	No	Yes	Classes	DNN	No	No	Accuracy-based

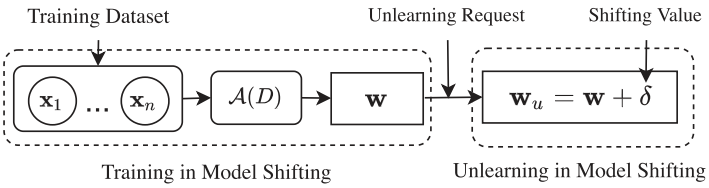

Fig. 8. Unlearning schemes based on model shifting.

5.1 Manipulation Based on Model Shifting

5.1.1 Unlearning Schemes Based on Model Shifting. As shown in Figure 8, model-shifting methods usually eliminate the influence of unlearning data by directly updating the model parameters. These methods mainly fall into one of two types—influence unlearning and Fisher unlearning—but there are a few other methods.

(1) Influence unlearning methods

Influence unlearning methods are usually based on influence theory [38]. Guo et al. [41] proposed a novel unlearning scheme called *certified removal*. Inspired by differential privacy [88], *certified removal* first limits the maximum difference between the unlearned and retrained models. Then, by applying a single step of Newton's method on the model parameters, a *certified removal* mechanism is provided for practical applications of L_2- regularized linear models that are trained using a differentiable convex loss function. Additionally, the training loss is perturbed with a loss perturbation technique that hides the *gradient residual*. This further prevents any adversaries from extracting information from the unlearned model. It is worth noting, however, that this solution is only applicable to simple machine learning models, such as linear models, or only adjusts the linear decision-making layer for deep neural networks, which does not eliminate the information of the removed data sample, since the representations are still learned within the model.

Izzo et al. [51] proposed an unlearning method based on a gradient update called ***projection residual update* (PRU)**. The method focuses on linear regression and shows how to improve the algorithm's runtime given in Reference [41] from quadratic complexity to linear complexity. The unlearning intuition is as follows: If one can calculate the values $\hat{y}_{i_{\mathcal{D}_u}} = w_u(x_{i_{\mathcal{D}_u}})$, predicted by the unlearned model on each of the unlearned samples $x_{i_{\mathcal{D}_u}}$ in $\mathcal{D}_u$ without knowing w_u, and then minimize the loss of already-trained model on the synthetic samples $(x_{i_{\mathcal{D}_u}}, \hat{y}_i)$, then the parameters will move closer to $\mathbf{w}_u$, since it will achieve the minimum loss with samples $(x_{i_{\mathcal{D}_u}}, \hat{y}_{i_{\mathcal{D}_u}})$. To calculate the values $\hat{y}_{i_{\mathcal{D}_u}}$ without knowing w_u, they introduced a statistics technique and computed leave-one-out residuals. Similar to the above, this method only considers the unlearning process in simple models.

Information leaks may not only manifest in a single data sample but also in groups of features and labels [53]. For example, a user's private data, such as their telephone number and place of residence, are collected by data providers multiple times and generated as different samples of the training dataset. Therefore, unlearning operations should also focus on unlearning a group of features and corresponding labels.

To solve such problems, Warnecke et al. [53] proposed a *certified unlearning* scheme for unlearning features and labels. By reformulating the influence estimation of samples on the already-trained models as a form of unlearning, they derived a versatile approach that maps changes of the training dataset in retrospection to closed-form updates of the model parameters. They then proposed different unlearning methods based on *first-order* and *second-order* gradient updates for two different types of machine learning models. For the *first-order* update, the parameters were updated based on the difference between the gradient of the original and the perturbed samples.

For the *second-order* update, they approximated an inverse Hessian matrix based on the scheme proposed in Reference [89] and updated the model parameters based on this approximate matrix. Theoretical guarantees were also provided for feature and label unlearning by extending the concept of differential privacy [88] and certified unlearning [41]. However, this solution is only suitable for feature unlearning from tabular data and does not provide any effective solution for image features.

(2) Fisher unlearning method

The second type of model-shifting technique uses the Fisher information [90] of the remaining dataset to unlearn specific samples, with noise injected to optimize the shifting effect. Golatkar et al. [40] proposed a weight *scrubbing* method to unlearn information about a particular class as a whole or a subset of samples within a class. They first give a computable upper bound to the amount of the information retained about the unlearning dataset after applying the unlearning procedure, which is based on the **Kullback-Leibler (KL)** divergence and Shannon mutual information. Then, an optimal quadratic unlearning algorithm based on a Newton update and a more robust unlearning procedure based on a noisy Newton update were proposed. Both schemes can ensure that a cohort can be unlearned while maintaining good accuracy for the remaining samples. However, this unlearning scheme is based on various assumptions, which limits its applicability.

For deep learning models, bounding the information that can be extracted from the perspective of weight or weight distribution is usually complex and may be too restrictive. Deep networks have a large number of equivalent solutions in the distribution space, which will provide the same activation on all test samples [43]. Therefore, many schemes have redirected unlearning operations from focusing on the weights to focus on the final activation.

Unlike their previous work, Golatkar et al. [43] provide bounds for how much information can be extracted from the final activation. They first transformed the bounding from a weight perspective to final activation based on Shannon mutual information and proposed a computable bound using the KL-divergence between the distribution of final activation of an unlearned model and retrained model. Inspired by the **neural tangent kernel (NTK)** [91, 92], they considered that deep network activations can be approximated as a linear function of the weights. Hence, an optimal unlearning procedure is then provided based on a Fisher information matrix. However, due to the specific structure of deep neural networks, considering unlearning process only in the final activation layer may not satisfy the effectiveness of unlearning. Once an attacker obtains all model parameters in a white-box scenario, they can still infer information from the middle layers.

Golatkar et al. [52] also proposed a mix-privacy unlearning scheme based on a new *mixed-privacy* training process. This new training process assumes the traditional training dataset can be divided into two parts: *core* data and *user* data. Model training on the *core* data is non-convex, and then further training, based on the quadratic loss function, is done with the *user* data to meet the needs of specific user tasks. Based on this assumption, unlearning operations on the *user* data can be well executed based on the existing quadratic unlearning schemes. Finally, they also derived bounds on the amount of information that an attacker can extract from the model weights based on mutual information. Nevertheless, the assumption that the training dataset is divided into two parts and that the model is trained using different methods on each of these parts restricts unlearning requests to only those data that are easy to unlearn, making it difficult to unlearn other parts of the data.

Liu et al. [93] transferred the unlearning method from a centralized environment to federated learning by proposing a distributed Newton-type model updating algorithm to approximate the loss function trained by the local optimizer on the remaining dataset. This method is based on the Quasi-Newton method and uses a first-order Taylor expansion. They also use diagonal empirical **Fisher Information Matrix (FIM)** to efficiently and accurately approximate the inverse Hessian

vector, rather than computing it directly, to further reduce the cost of the retraining process. However, this solution will result in a significant reduction in accuracy when dealing with complex models.

(3) Other Shifting Schemes

Schelter et al. [24] introduced the problem of making trained machine learning models unlearn data via *decremental updates*. They described three decremental update algorithms for different machine learning tasks. These included one based on item-based collaborative filtering, another based on ridge regression, and the last based on k-nearest neighbors. With each machine learning algorithm, the intermediate results are retained, and the model parameters are updated based on the intermediate results and unlearning data D_u, resulting in an unlearned model. However, this strategy can only be utilized with those models that can be straightforwardly computed to obtain the model parameters after the unlearning process, limiting the applicability of this scheme.

In addition, Graves et al. [45] proposed a laser-focused removal of sensitive data, called *amnesiac unlearning*. During training, the model provider retains a variable that stores which samples appear in which batch, as well as the parameter updates for each batch. When a data unlearning request arrives, the model owner undoes the parameter updates from only the batches containing the sensitive data, that is, $\mathcal{M}_{w_u} = \mathcal{M}_w - \sum \Delta_w$, where $\mathcal{M}_w$ is the already-trained model and Δ_w are the parameter updates after each batch. Because undoing some parameters might greatly reduce the performance of the model, the model provider can perform a small amount of fine-tuning after an unlearning operation to regain performance. This approach requires the storage of a substantial amount of intermediate data. As the storage interval decreases, the amount of cached data increases, and smaller intervals lead to more efficient model unlearning. Therefore, a tradeoff exists between efficiency and effectiveness in this method.

The above methods mainly focused on the core problem of empirical risk minimization, where the goal is to find approximate minimizers of the empirical loss on the remaining training dataset after unlearning samples [41, 51]. Sekhari et al. [42] proposed a more general method of reducing the loss of unseen samples after an unlearning process. They produced an unlearned model by removing the contribution of some samples from an already-trained model using a disturbance update calculated based on some cheap-to-store data statistics during training. In addition, they proposed an evaluation parameter to measure the unlearning capacity. They also improved the data unlearning capacity of convex loss functions, which saw a quadratic improvement in terms of the dependence of d over differential privacy, where d is the problem dimension.

5.1.2 Verifiability of Schemes Based on Parameter Shifting. Izzo et al. [51] provided two metrics to measure the effectiveness: L_2 *distance* and *feature injection test*. L_2 *distance* measures the distance between the unlearned model and the retrained model. If the L_2 *distance* is small, then the models are guaranteed to make similar predictions, which could reduce the impact of output-based attacks, like a membership inference attack. The *feature injection test* can be thought of as a verification scheme based on a poisoning attack.

Golatkar et al. [40, 43, 52] verify the effectiveness of their unlearning schemes based on accuracy and relearning time. They also developed two new verification metrics: *model confidence* and *information bound* [40]. *Model confidence* is formulated by measuring the distribution of the entropy of the output predictions on the remaining dataset, the unlearning dataset, and the test dataset. Then they evaluated the similarity of those distributions against the confidence of a trained model that has never seen the unlearning dataset. The higher the degree of similarity, the better the effect of the unlearning process. The *information bound* metric relies on KL-divergence to measure the information remaining about the unlearning dataset within the model after the unlearning process.

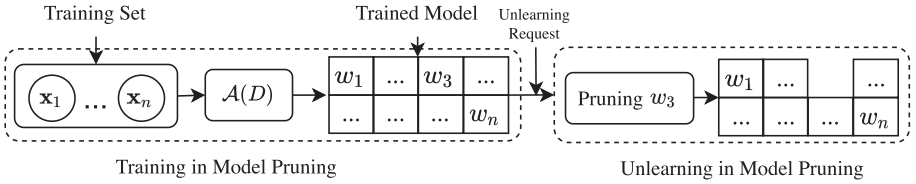

Fig. 9. Unlearning schemes based on model pruning.

Different from their previous work, Golatkar et al. [43] also evaluate the information remaining within the weights and the activation. In their other work [52], they provided a new metric, *activation distance*, to analyze the distance between the final activations of an unlearned model and a retrained model. This is a similar metric to *model confidence* [40]. In addition, they use attack-based methods for verification [43, 52].

Guo et al. [41], Warnecke et al. [53], and Sekhari et al. [42] provide a method of theoretical verification to verify the effectiveness of their proposed unlearning schemes. Based on the guarantee provided by *certified unlearning*, they limit the distribution similarity between the unlearned model and the retrained model. Warnecke et al. [53] also use the *exposure metric* [2] to measure the remaining information after unlearning. Liu et al. [93] analyzed the validity of the unlearning scheme through two aspects. The first metric, **Symmetric Absolute Percentage Error (SAPE)**, is created based on accuracy. The second metric is the difference between the distribution of the model after the unlearning process and the distribution of the retraining model.

5.2 Manipulation Based on Model Pruning

5.2.1 Unlearning Schemes Based on Model Pruning. As shown in Figure 9, methods based on model pruning usually prune a trained model to produce a model that can meet the requests of unlearning. It is usually applied in the scenario of federated learning, where a model provider can modify the model's historical parameters as an update. Federated learning is a distributed machine learning framework that can train a unified deep learning model across multiple decentralized nodes, where each node holds its own local data samples for training, and those samples never need to be exchanged with any other nodes [94]. There are mainly three types of federated learning: *horizontal*, *vertical*, and *transfer learning* [95].

Based on the idea of trading the central server's storage for the unlearned model's construction, Liu et al. [54] proposed an efficient federated unlearning methodology, *FedEraser*. Historical parameter updates from the clients are stored in the central server during the training process, and then the unlearning process unfolds in four steps: (1) *calibration training*, (2) *update calibrating*, (3) *calibrated update aggregating*, and (4) *unlearned model updating*, to achieve the unlearning purpose. In *calibration training* and *update calibration* steps, several rounds of a calibration retraining process are performed to approximate the unlearning updates without the target client. In the *calibrated update aggregating* and the *unlearned model updating* steps, standard federated learning aggregation operations are used to aggregate those unlearning updates and further update the global model. This eliminates the influence of the target data.

However, the effectiveness of this scheme will decrease dramatically as the number of unlearning requests increases; this is because the gradients are cached during the training phase, and the unlearning process will not update these gradients to satisfy subsequent unlearning requests [54]. Second, this solution also requires caching of intermediate data, which will cost more storage.

Inspired by the observation that different channels have a varying contribution to different classes in trained CNN models, Wang et al. [55] analyzed the problem of selectively unlearning classes in a federated learning setting. They introduced the concept of **term frequency-inverse**

document frequency (TF-IDF) [96] to quantify the class discrimination of the channels. Similar to analyzing how relevant a word is to a document in a set of documents, they regarded the output of a channel as a word and the feature map of a category as a document. Channels with high TF-IDF scores have more discriminatory power in the target categories and thus need to be pruned. An unlearning procedure via channel pruning [97] was also provided, followed by a fine-tuning process to recover the performance of the pruned model. In their unlearning scheme, however, while the parameters associated with the class that needs to be unlearned are pruned, the parameters with other classes also become incomplete, which will affect the model performance. Therefore, the unlearned model is only available when the fined-tuned training process is complete.

Baumhauer et al. [56] provided a machine unlearning scheme based on linear filtration. They first transformed the existing logit-based classifier models into an integrated model that can be decomposed into a (potentially nonlinear) feature extraction, followed by a multinomial logistic regression. Then, they focused the unlearning operation on the logistic regression layer, proposing a "black-box" unlearning definition. To unlearn the given samples, four different filtration methods are defined, namely, *naive unlearning*, *normalization*, *randomization*, and *zeroing*. These effectively filter the outputs of the logistic regression layer. On the contrary, they only considered the unlearning process within the last layer, which will lead to a potential risk that if an attacker gets access to the model parameters of the middle layer, then the information of unlearning data may also be leaked.

5.2.2 Verifiability of Schemes Based on Model Pruning. Liu et al. [54] present an experimental verification method based on a membership inference attack. Two evaluation parameters are specified: *attack precision* and *attack recall*, where *attack precision* denotes the proportion of unlearned samples that is expected to participate in the training process. *Attack recall* denotes the fraction of unlearned samples that can be correctly inferred as part of the training dataset. In addition, a *prediction difference* metric is also provided, which measures the difference in prediction probabilities between the original global model and the unlearned model. Wang et al. [55] evaluate verifiability based on model accuracy.

Baumhauer et al. [56] defined a divergence measure based on a Bayes error rate for evaluating the similarity of the resulting distributions $P(\mathbf{L}_{\text{seen}})$ and $P(\mathbf{L}_{\neg\ \text{seen}})$, where $\mathbf{L}_{\text{seen}}$ and $\mathbf{L}_{\neg\ \text{seen}}$ are the pre-softmax outputs of the unlearned model and a retrained model. When the result of the Bayes error rate is close to 0, it indicates that $P(\mathbf{L}_{\text{seen}})$ and $P(\mathbf{L}_{\text{-seen}})$ are similar and the unlearning process has unlearned the sample's information from the model. In addition, they use a model inversion attack to evaluate verifiability [6].

5.3 Manipulation Based on Model Replacement

5.3.1 Unlearning Schemes Based on Model Replacement. As shown in Figure 10, model replacement-based methods usually calculate almost all possible sub-models in advance during the training process and store them together with the deployed model. Then, when an unlearning request arrives, only the sub-models affected by the unlearning operation need to be replaced with the pre-stored sub-models. This type of solution is usually suitable for some machine learning models, such as tree-based models. Decision tree is a tree-based learning model, in which each leaf node represents a prediction value, and each internal node is a decision node associated with an attribute and threshold value. Random forest is an integrated decision tree model that aims to improve prediction performance [98, 99].

To improve the efficiency of the unlearning process for tree-based machine learning models, Schelter et al. [57] proposed *Hedgecut*, a classification model based on **extremely randomized trees (ERTs)** [100]. First, during the training process, the tree model is divided into robust splits

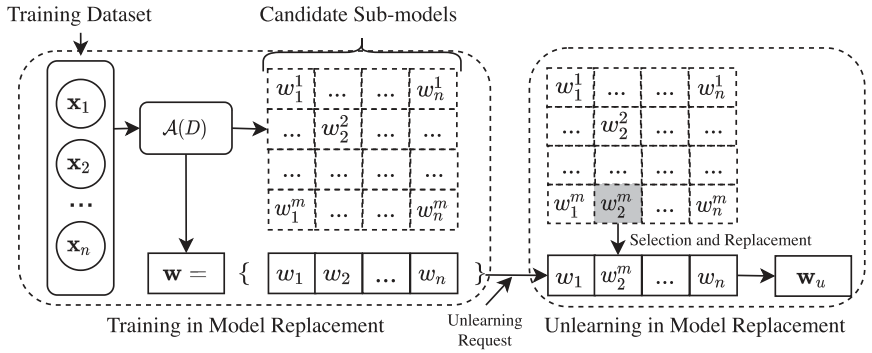

Fig. 10. Unlearning schemes based on model replacement.

and non-robust splits based on the proposed robustness quantification factor. A robust split indicates that the subtree's structure will not change after unlearning a small number of samples, while for non-robust splits, the structure may be changed. In the case of unlearning a training sample, *HedgeCut* will not revise robust splits but will update those leaf statistics. For non-robust splits, *HedgeCut* recomputes the split criterion of the maintained subtree variants, which were previously kept inactive, and selects a subtree variant as a new non-robust split of the current model.

For the tree-based models, Brophy et al. [58] also proposed **DaRE (Data Removal-Enabled)** forests, a random forest variant that enables the efficient removal of training samples. DaRE is mainly based on the idea of retraining subtrees only as needed. Before the unlearning process, most k randomly selected thresholds per attribute are computed, and intermediate statistics data are stored within each node in advance. This information is sufficient to recompute the split criterion of each threshold without iterating through the data, which can greatly reduce the cost of recalculation when unlearning the dataset. They also introduced random nodes at the top of each tree. Intuitively, the nodes near the top of the tree affect more samples than those near the bottom, which makes it more expensive to retrain them when necessary. Random nodes minimally depend on the statistics of the data, rather than the way greedy methods are used, and rarely need to be retrained. Therefore, random nodes can further improve the efficiency of unlearning.

The above two schemes need to compute a large number of possible tree structures in advance, which would cost a large number of storage resources [57, 58]. Besides, this replacement scheme is difficult to be applied to other machine learning models, such as deep learning models, since it is difficult to achieve partial model structure after removing each sample in advance.

Chen et al. [59] proposed a machine unlearning scheme called *WGAN* unlearning, which removes information by reducing the output confidence of unlearned samples. Machine learning models usually have different confidence levels toward the model's outputs [101]. To reduce confidence, *WGAN* unlearning first initializes a generator as the trained model that needs to unlearn data. Then, the generator and discriminator are trained alternatingly until the discriminator cannot distinguish the output difference of the model between unlearning dataset and third-party data. Until this, the generator then becomes the final unlearned model. However, this method achieves unlearning process through an alternating training process, which brings a limited improvement in efficiency compared to the unlearning method of retraining from scratch.

Wu et al. [60] proposed an approximate unlearning method based on intermediate parameters cached during the training phase called *DeltaGrad*, which could quickly unlearn information from machine learning models that are based on gradient descent algorithms. They divided the retraining process into two parts. One part computes the full gradients exactly based on the remaining training dataset. The other part uses the L-BGFS algorithm [102] and a set of updates from some

prior iterations to calculate Quasi-Hessians approximating the true Hessian-vector. These Quasi-Hessians are then used to approximate the update in the remaining process. These two parts train cooperatively to generate the unlearned model. This approach will reduce the performance of the model, however, after unlearning process, since part of the model update is calculated based on the approximative methods. In addition, the number of iterations required for the model to converge will also increase, which will reduce the efficiency of the unlearning process.

5.3.2 Verifiability of Schemes Based on Model Replacement. Chen et al. [59] verified their proposed scheme with a membership inference attack and a technique based on **false negative rates (FNRs)** [103], where FNR: $FNR = \frac{FN}{TP+FN}$, TP means that the membership inference attack test samples were considered to be training dataset and FN means the data was deemed to be non-training data. If the target model successfully unlearns the samples, then the member inference attack will treat the training dataset as non-training data. Thus, FN will be large, while TP will be small, and the corresponding FNR will be large. Indirectly, this reflects the effectiveness of the unlearning process.

Schelter et al. [57], Brophy et al. [58], and Wu et al. [60] only provide evaluations in terms of runtime and accuracy, and they do not provide reasonable experimental or theoretical verifiability guarantees of their unlearning processes.

5.4 Summary of Model Manipulation

In these last subsections, we reviewed studies that apply model shifting, model pruning, and model replacement techniques as unlearning processes. A summary of the surveyed studies is shown in Table 7, where we list the key differences between each paper.

Compared to the unlearning schemes based on data reorganization, we can see that few of the above papers make use of intermediate data for unlearning. This is because the basic idea of those unlearning schemes is to directly manipulate the model itself, rather than the training dataset. The model manipulation methods calculate the influence of each sample and offset that influence using a range of techniques [38], while data reorganization schemes usually reorganize the training dataset to simplify the unlearning process. For this reason, model manipulation methods somewhat reduce the resource consumption used by intermediate storage.

Second, most of the above schemes focus on relatively simple machine learning problems, such as linear logistic regression, or complex models with special assumptions [40, 41, 43, 51]. Removing information from the weights of standard convolutional networks is still an open problem, and some preliminary results are only applicable to small-scale problems. One of the main challenges with unlearning processes for deep networks is how to estimate the impact of a given training sample on the model parameters. Also, the highly non-convex losses of CNNs make it very difficult to analyze those impacts on the optimization trajectory. Current research has focused on simpler convex learning problems, such as linear or logistic regression, for which theoretical analysis is feasible. Therefore, evaluating the impact of specific samples on deep learning models and further proposing unlearning schemes for those models are two urgent research problems.

In addition, most model manipulation-based methods will affect the consistency or prediction accuracy of the original models. There are two main reasons for this problem. First, due to the complexity of calculating the impact of the specified sample on the model, manipulating a model's parameters based on unreliable impact results or assumptions will lead to a decline in model accuracy. Second, Wang et al.'s [55] scheme pruned specific parameters in the original models, which will also reduce the accuracy of the model due to the lack of some model prediction information. Thus, more efficient unlearning mechanisms, which simultaneously ensure the validity of the unlearning process and guarantee performance, are worthy of research.

Table 7. The Surveyed Studies that Employed Model Manipulation Techniques for Unlearning Process

Papers	Unlearning Methods	Unlearning Target	Training Dataset	Intermediates	Unlearned Samples' Type	Target Models' Type	Consistency	Accuracy	Verifiability
Guo et al. [41]	Model Shifting	Strong Unlearning	Yes	No	Samples	Linear Models with Strongly Convex Regularization	No	No	Theory-based
Izzo et al. [51]	Model Shifting	Strong Unlearning	No	Yes	Batches	Linear and Logistic Regression Models	No	No	L^2 distance and Attack-based
Warnecke et al. [53]	Model Shifting	Strong Unlearning	Yes	No	Features and Labels	Convex or Non-convex models	No	No	Theory-based and Method in [2]
Golatkar et al. [40]	Model Shifting	Strong Unlearning	Yes	No	Samples in One Class	DNN	No	No	Accuracy-based, Relearn time-based, Model confidence and Information Bound-based
Golatkar et al. [43]	Model Shifting	Strong Unlearning	Yes	No	Samples	DNN	No	No	Accuracy-based, Relearn time-based, Attack-based and Information Bound-based
Liu et al. [93]	Model Shifting	Strong Unlearning	Yes	Yes	Samples	DNN	No	No	Accuracy-based
Golatkar et al. [52]	Model Shifting	Strong Unlearning	Yes	No	Samples	DNN	No	No	Accuracy-based, Relearn time-based, Activation distance and Attack-based
Schelter et al. [24]	Model Shifting	Exact Unlearning	No	Yes	Samples	Specified model	Yes	Yes	-
Graves et al. [45]	Model Shifting	Strong Unlearning	No	Yes	Samples	DNN	No	No	Attack-based
Sekhari et al. [42]	Model Shifting	Strong Unlearning	No	Yes	Samples	Convex Models	No	No	Theory-based
Wang et al. [55]	Model Pruning	Strong Unlearning	Yes	No	Client Data	Federated Learning Model	No	No	-
Baumhauer et al. [56]	Model Pruning	Weak Unlearning	No	No	Classes	Logit-based Classifiers	No	No	Attack-based
Schelter et al [57]	Model Replacement	Exact Unlearning	No	Yes	Samples	Extremely Randomized Trees	Yes	Yes	-
Brophy et al [58]	Model Replacement	Exact Unlearning	Yes	Yes	Batches	Random Forests	Yes	Yes	-
Chen et al. [59]	Model Replacement	Strong Unlearning	No	No	Samples	Deep Classifier Models	No	No	Attack-based
Wu et al. [60]	Model Replacement	Strong Unlearning	Yes	Yes	Samples	SGD-based Models	No	No	-

It is worth pointing out that most schemes provide a reasonable method with which to evaluate the effectiveness of the unlearning process. Significantly, model manipulation methods usually give a verifiability guarantee using theory-based and information bound-based methods [40, 41, 43]. Compared to the simple verification methods based on accuracy, relearning, or attacks, the methods based on theory or information bounds are more effective. This is because simple verification methods usually verify effectiveness based on output confidence. While the effects of the samples to be unlearned can be hidden from the output of the network, insights may still be gleaned by probing deep into its weights. Therefore, calculating and limiting the maximum amount of information that may be leaked at the theoretical level will be a more convincing method. Overall, however, more theory-based techniques for evaluating verifiability are needed.

In summary, the unlearning methods based on model shifting usually aim to offer higher efficiency by making certain assumptions about the training process, such as which training dataset or optimization techniques have been used. In addition, those mechanisms that are effective for simple models, such as linear regression models, become more complex when faced with advanced deep neural networks. Model pruning schemes require far-reaching modifications of the existing architecture of the model in the unlearning process [55, 56], which could affect the performance of the unlearned models. It is worth noting that model replacement unlearning methods usually need to calculate all possible parameters and store them in advance, since they unlearn by quickly replacing the model parameters using these pre-calculated parameters. Thus, more effective unlearning schemes, that simultaneously consider model usability, storage costs, and the applicability of the unlearning process, are urgent research problems.

6 OPEN QUESTIONS AND FUTURE DIRECTIONS

In this section, we will analyze current and potential trends in machine unlearning and summarize our findings. In addition, we identify several unanswered research directions that could be addressed to progress the foundation of machine unlearning and shape the future of AI.

6.1 Open Questions

As research continues to evolve, machine unlearning may expand further in the following areas, and this potential trend has already begun to take shape:

6.1.1 The Universality of Unlearning Solutions. Unlearning schemes with higher compatibility need to be explored. As development progresses, machine unlearning schemes supporting different models and unlearning data types have been proposed in various fields. For example, Zhang et al. [63] provided an unlearning scheme in image retrieval, while Chen et al. [47] considered graph unlearning problem. However, most of the current unlearning schemes are limited to a specific scenario. They are mostly designed to leverage the special characteristics of a particular learning process or training scheme [24, 47, 54]. Although it is feasible to design an appropriate unlearning scheme for every model, this is an inefficient approach that would require many manual interventions [104, 105].

Therefore, universality unlearning schemes should be not only applicable to different model structures and training methods, but also to different types of training datasets, such as graphs, images, text, or audio data. The data pruning-based scheme is an existing and effective approach that could achieve universality unlearning purposes based on ensemble learning techniques [30]. However, this method breaks the correlation relationships in some scenarios, which is not suitable for models that require correlation information to complete training.

6.1.2 The Security of Machine Unlearning. Unlearning schemes should ensure the security of any data, especially the unlearned dataset. Recently, existing research has shown that the

unlearning operation not only does not reduce the risk of user privacy leakage but actually increases this risk [106, 107]. These attack schemes mainly compare the models before and after the unlearning process. Thus, a membership inference attack or a poisoning attack would reveal a significant amount of detailed information about the unlearned samples [78, 108]. To counteract such attacks, Neel et al. [50] have proposed a protection method based on Gaussian perturbation in their unlearning scheme.

In addition, many previous unlearning schemes rely on the remaining dataset, intermediate cached model's parameters. However, they do not consider the security of this intermediate information and whether an attack would recover any information about the unlearned samples [30, 57]. Therefore, the design of further unlearning schemes needs to consider that any before and after models should not expose any information about the samples that need to be unlearned. Further, the security of the data cached during the unlearning process also needs to be explored.

6.1.3 The Verification of Machine Unlearning. Verification methods should be easy to implement and applicable to users. Most current simple verification schemes, such as those based on attacks, relearning time, and accuracy [45, 52], are derived from existing learning or attack metrics. Those one-sided methods seldom provide strong verification of the unlearning process's effectiveness [44, 109, 110]. Meanwhile, unlearning methods with a theoretical guarantee are usually based on rich assumptions and can rarely be applied to complex models, since complex deep models usually make those assumptions invalid [41, 53]. In addition, these verification schemes are not user-friendly and easy to implement.

Therefore, the verification schemes should consider the feasibility and acceptability, that is, users should be able to understand and verify whether their unlearning request has been completed based on some simple operations. There are already some relevant schemes, such as the backdoor-based verification mechanism in Reference [44] and the encryption-based verification scheme in Reference [111]. However, these schemes are still quite difficult for ordinary users. Therefore, an easy-to-implement and understanding verification scheme is a topic worthy of research.

6.1.4 The Applications of Machine Unlearning. While promoting individual data privacy, machine unlearning has also gradually emerged as a solution for other applications. Regulations and privacy issues have resulted in the need to allow a trained model to unlearn some of its training data. Apart from these, there are several other scenarios where efficient machine unlearning would be beneficial. For instance, it could be used to accelerate the process of leave-one-out-cross-validation, removing adversarial or poisoning samples, and identifying significant and valuable data samples within a model [13]. As of now, some relevant applications have emerged [53, 112]. For example, Alexander et al. [53] proposed a feature unlearning scheme that could be used to address fairness issues.

At the same time, the machine unlearning scheme can also serve as an effective attack strategy to strengthen the robustness of the model. One potential attack scenario to consider is as follows: The attacker first introduces pre-designed malicious samples into the dataset, which are subsequently used by the model provider to train the model. After that, the attacker initiates unlearning requests to remove the information about those pre-designed samples from the model, which will affect the performance and fairness of the model, or unlearning efficiency [108]. Therefore, in addition to strengthening data protection, machine unlearning has enormous potential in other areas.

6.2 Future Directions

Information synchronization: Similar to process synchronization in operating systems, machine unlearning may create information synchronization problems [113, 114]. Since machine unlearning is usually computationally costly, the model provider may not be able to complete the

unlearning process immediately. In the interim, how to handle incoming prediction requests deserves careful consideration. Consider that, if predictions continue to be returned prior to the model's update, then unlearned data may be revealed. However, if all requests for prediction are rejected until the unlearning process is completed, then model utility and service standards will surely suffer. Therefore, how to handle prediction requests within this interval needs comprehensive consideration.

Federated unlearning: Federated learning is a special kind of distributed learning that is characterized by various unstable users distributed in different places, each of whom has control over their devices and data [115, 116]. Imteaj et al. [95] show that model providers are more likely to receive requests to remove specific samples from a model trained in a federated learning setting. For example, when a user quits the collaborative training process, they may ask for their contribution to be removed from the collaborative model. Therefore, how to efficiently realize machine unlearning in a federated learning setting, considering the limitations of such a setting, such as unacceptable training data, unstable connections, and so on, is worthy of research [111].

Disturbance techniques: Problems with privacy leaks before and after machine unlearning, are mainly caused by the differences between the two models. A feasible solution is to interfere with the training process or adjust the model parameters so the model is different from what it should have been. Data disturbance techniques have the ability to interfere with specific data while ensuring overall data availability [117]. For example, Guo et al. [41] hide information about the unlearned samples using a loss perturbation technique [118] at the time of training. The technique involves perturbing the empirical risk through a random linear term. As such, a useful direction for future research may be to incorporate data disturbance into machine unlearning problems and to develop new mechanisms to support more sophisticated analyses.

Feature-based unlearning methods: Unlearning based on model shifting usually removes the impact of the unlearning dataset by calculating the influence on the model [40, 43]. However, calculating the influence of the samples directly may be too complex [38]. Can we shift the calculation of influence from the original training samples to a group of specific features? When an unlearning request arrives, influence can be calculated based on the features instead of the original training samples. Technologies that may be relevant to this question include feature extraction [119], feature generation [120], and feature selection [121], which could be integrated into unlearning operations.

Game-theory-based balance: Game theory has been a booming field with several representative privacy-preserving techniques coming out in the past decade [122]. There are many schemes involving privacy-preserving solutions based on game theory that trade off data privacy and utility issues [123, 124]. For a model provider, machine unlearning is also a tradeoff between model performance and user privacy, where an over-unlearning strategy may lead to performance degradation, while insufficient protection may lead to privacy leaks. Can we formalize the unlearning problem as a game between two players: a model provider and a data provider? If so, then we could provide a game model between these two entities and determine a set of strategies and utilities to figure out how to perform unlearning operations that maintain the model's performance to the maximum extent possible. Such an approach could also protect the user's sensitive data from being leaked. These are open issues that need to be explored further.

7 CONCLUSION

Machine learning methods have become a strong driving force in revolutionizing a wide range of applications. However, they are also bringing requests to delete training samples from models due to privacy, usability, or other entitlement requirements. Machine unlearning is a new technology that can cater to these requests for deletion, and many research studies have been carried out in

this regard. In this survey, we provided a comprehensive overview of machine unlearning techniques with a particular focus on the two main types of unlearning processes: data reorganization and model manipulation. First, we provided the basic concept and different targets of machine unlearning. By analyzing typical approaches, we proposed a novel taxonomy and summarized their basic principles. We also reviewed many existing studies and discussed the strengths and limitations of those studies within each category. In addition, we emphasized the importance of verifying machine unlearning processes and reviewed the different ways in which machine unlearning can be verified. Finally, we discussed several issues that would merit future research and provided some feasible directions that need to be explored in the future. Our future work will focus on exploring the potential of machine unlearning in intriguing areas such as federated learning with a verifiability property.

REFERENCES

[1] M. I. Jordan and T. M. Mitchell. 2015. Machine learning: Trends, perspectives, and prospects. *Science* 349, 6245 (2015), 255–260. DOI: http://dx.doi.org/10.1126/science.aaa8415

[2] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. 2019. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX Security Symposium*. USENIX Association, 267–284. Retrieved from https://www.usenix.org/conference/usenixsecurity19/presentation/carlini.

[3] Minxing Zhang, Zhaochun Ren, Zihan Wang, Pengjie Ren, Zhumin Chen, Pengfei Hu, and Yang Zhang. 2021. Membership inference attacks against recommender systems. In *ACM SIGSAC Conference on Computer and Communications Security*. ACM, 864–879. DOI: http://dx.doi.org/10.1145/3460120.3484770

[4] Mahdi Khosravy, Kazuaki Nakamura, Yuki Hirose, Naoko Nitta, and Noboru Babaguchi. 2022. Model inversion attack by integration of deep generative models: Privacy-sensitive face generation from a face recognition system. *IEEE Trans. Inf. Forens. Secur.* 17 (2022), 357–372. DOI: http://dx.doi.org/10.1109/TIFS.2022.3140687

[5] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership inference attacks against machine learning models. In *IEEE Symposium on Security and Privacy*. IEEE Computer Society, 3–18. DOI: http://dx.doi.org/10.1109/SP.2017.41

[6] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. 2015. Model inversion attacks that exploit confidence information and basic countermeasures. In *22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1322–1333. DOI: http://dx.doi.org/10.1145/2810103.2813677

[7] Eduard Fosch-Villaronga, Peter Kieseberg, and Tiffany Li. 2018. Humans forget, machines remember: Artificial intelligence and the right to be forgotten. *Comput. Law Secur. Rev.* 34, 2 (2018), 304–313. DOI: http://dx.doi.org/10.1016/j.clsr.2017.08.007

[8] Michael Veale, Reuben Binns, and Lilian Edwards. 2018. Algorithms that remember: model inversion attacks and data protection law. *CoRR* abs/1807.04644 (2018).

[9] 2018. General Data Protection Regulation(GDPR). Retrieved from https://gdpr-info.eu/.

[10] 2018. California Consumer Privacy Act (CCPA). Retrieved from https://oag.ca.gov/privacy/ccpa.

[11] 2019. Japan - Data Protection Overview(JDPO). Retrieved from https://www.dataguidance.com/notes/japan-data-protection-overview.

[12] 2022. Consumer Privacy Protection Act(CPPA). Retrieved from https://blog.didomi.io/en-us/canada-data-privacy-law.

[13] Antonio Ginart, Melody Y. Guan, Gregory Valiant, and James Zou. 2019. Making AI forget you: Data deletion in machine learning. In *Advances in Neural Information Processing Systems: Annual Conference on Neural Information Processing Systems*. 3513–3526. Retrieved from https://proceedings.neurips.cc/paper/2019/hash/cb79f8fa58b91d3af6c9c991f63962d3-Abstract.html.

[14] Quoc Phong Nguyen, Bryan Kian Hsiang Low, and Patrick Jaillet. 2020. Variational Bayesian unlearning. In *Advances in Neural Information Processing Systems: Annual Conference on Neural Information Processing Systems*. Retrieved from https://proceedings.neurips.cc/paper/2020/hash/b8a6550662b363eb34145965d64d0cfb-Abstract.html.

[15] Sanjam Garg, Shafi Goldwasser, and Prashant Nalini Vasudevan. 2020. Formalizing data deletion in the context of the right to be forgotten. In *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part II*, Vol. 12106. Springer, 373–402. DOI: http://dx.doi.org/10.1007/978-3-030-45724-2_13

[16] Sanjay Krishnan and Eugene Wu. 2017. PALM: Machine learning explanations for iterative debugging. In *2nd Workshop on Human-In-the-Loop Data Analytics*. ACM, 4:1–4:6. DOI: http://dx.doi.org/10.1145/3077257.3077271

[17] Tianqing Zhu, Dayong Ye, Wei Wang, Wanlei Zhou, and Philip S. Yu. 2020. More than privacy: Applying differential privacy in key areas of artificial intelligence. *IEEE Trans. Knowl. Data Eng.* 34, 6 (2020). DOI: http://dx.doi.org/10.1109/TKDE.2020.3014246

[18] Kang Wei, Jun Li, Chuan Ma, Ming Ding, Cailian Chen, Shi Jin, Zhu Han, and H. Vincent Poor. 2022. Low-latency federated learning over wireless channels with differential privacy. *IEEE J. Sel. Areas Commun.* 40, 1 (2022), 290–307. DOI: http://dx.doi.org/10.1109/JSAC.2021.3126052

[19] Tianqing Zhu, Gang Li, Wanlei Zhou, and Philip S. Yu. 2017. Differentially private data publishing and analysis: A survey. *IEEE Trans. Knowl. Data Eng.* 29, 8 (2017), 1619–1638. DOI: http://dx.doi.org/10.1109/TKDE.2017.2697856

[20] Lefeng Zhang, Tianqing Zhu, Ping Xiong, Wanlei Zhou, and Philip S. Yu. 2022. More than privacy: Adopting differential privacy in game-theoretic mechanism design. *ACM Comput. Surv.* 54, 7 (2022), 136:1–136:37. DOI: http://dx.doi.org/10.1145/3460771

[21] Nan Xiang, Xiongtao Zhang, Yajie Dou, Xiangqian Xu, Ke-Wei Yang, and Yuejin Tan. 2021. High-end equipment data desensitization method based on improved Stackelberg GAN. *Expert Syst. Appl.* 180 (2021), 114989. DOI: http://dx.doi.org/10.1016/j.eswa.2021.114989

[22] Zhuo Wang, Kai Wei, Chunyu Jiang, Jiafeng Tian, Minjing Zhong, Yuan Liu, and Yanmei Liu. 2021. Research on productization and development trend of data desensitization technology. In *20th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*. IEEE, 1564–1569. DOI: http://dx.doi.org/10.1109/TrustCom53373.2021.00227

[23] Muhannad Al-Omari, Fangjun Li, David C. Hogg, and Anthony G. Cohn. 2022. Online perceptual learning and natural language acquisition for autonomous robots. *Artif. Intell.* 303 (2022), 103637. DOI: http://dx.doi.org/10.1016/j.artint.2021.103637

[24] Sebastian Schelter. 2020. "Amnesia'—Towards machine learning models that can forget user data very fast. In *10th Conference on Innovative Data Systems Research*. Retrieved from http://cidrdb.org/cidr2020/papers/p32-schelter-cidr20.pdf.

[25] Pei-Hung Chen, Wei Wei, Cho-Jui Hsieh, and Bo Dai. 2021. Overcoming catastrophic forgetting by Bayesian generative regularization. In *38th International Conference on Machine Learning (Proceedings of Machine Learning Research*, Vol. 139). PMLR, 1760–1770. Retrieved from http://proceedings.mlr.press/v139/chen21v.html.

[26] Huihui Liu, Yiding Yang, and Xinchao Wang. 2021. Overcoming catastrophic forgetting in graph neural networks. In *35th AAAI Conference on Artificial Intelligence, 33rd Conference on Innovative Applications of Artificial Intelligence, 11th Symposium on Educational Advances in Artificial Intelligence*. AAAI Press, 8653–8661. Retrieved from https://ojs.aaai.org/index.php/AAAI/article/view/17049.

[27] Ayush K. Tarun, Vikram S. Chundawat, Murari Mandal, and Mohan S. Kankanhalli. 2021. Fast yet effective machine unlearning. *CoRR* abs/2111.08947 (2021).

[28] Enayat Ullah, Tung Mai, Anup Rao, Ryan A. Rossi, and Raman Arora. 2021. Machine unlearning via algorithmic stability. In *Conference on Learning Theory (Proceedings of Machine Learning Research*, Vol. 134). PMLR, 4126–4142. Retrieved from http://proceedings.mlr.press/v134/ullah21a.html.

[29] Yinzhi Cao and Junfeng Yang. 2015. Towards making systems forget with machine unlearning. In *IEEE Symposium on Security and Privacy*. IEEE Computer Society, 463–480. DOI: http://dx.doi.org/10.1109/SP.2015.35

[30] Lucas Bourtoule, Varun Chandrasekaran, Christopher A. Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. 2021. Machine unlearning. In *42nd IEEE Symposium on Security and Privacy*. IEEE, 141–159. DOI: http://dx.doi.org/10.1109/SP40001.2021.00019

[31] Chen Wu, Sencun Zhu, and Prasenjit Mitra. 2022. Federated unlearning with knowledge distillation. Retrieved from https://arxiv.org/abs/2201.09441.

[32] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. 2019. Federated machine learning: Concept and applications. *ACM Trans. Intell. Syst. Technol.* 10, 2 (2019), 12:1–12:19. DOI: http://dx.doi.org/10.1145/3298981

[33] Chong Chen, Fei Sun, Min Zhang, and Bolin Ding. 2022. Recommendation unlearning. In *ACM Web Conference*. ACM, 2768–2777. DOI: http://dx.doi.org/10.1145/3485447.3511997

[34] Yuyuan Li, Xiaolin Zheng, Chaochao Chen, and Junlin Liu. 2022. Making recommender systems forget: Learning and unlearning for erasable recommendation. DOI: http://dx.doi.org/10.48550/arXiv.2203.11491

[35] Thanh Tam Nguyen, Thanh Trung Huynh, Phi Le Nguyen, Alan Wee-Chung Liew, Hongzhi Yin, and Quoc Viet Hung Nguyen. 2022. A survey of machine unlearning. DOI: http://dx.doi.org/10.48550/arXiv.2209.02299

[36] Saurabh Shintre, Kevin A. Roundy, and Jasjeet Dhaliwal. 2019. Making machine learning forget. In *Privacy Technologies and Policy - 7th Annual Privacy Forum (Lecture Notes in Computer Science*, Vol. 11498). Springer, 72–83. DOI: http://dx.doi.org/10.1007/978-3-030-21752-5_6

[37] Anvith Thudi, Gabriel Deza, Varun Chandrasekaran, and Nicolas Papernot. 2022. Unrolling SGD: Understanding factors influencing machine unlearning. In *7th IEEE European Symposium on Security and Privacy*. IEEE, 303–319. DOI: http://dx.doi.org/10.1109/EuroSP53844.2022.00027

[38] Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *34th International Conference on Machine Learning (Proceedings of Machine Learning Research*, Vol. 70). PMLR, 1885–1894. Retrieved from http://proceedings.mlr.press/v70/koh17a.html.

[39] Jinu Gong, Osvaldo Simeone, Rahif Kassab, and Joonhyuk Kang. 2021. Forget-SVGD: Particle-based Bayesian federated unlearning. *CoRR* abs/2111.12056 (2021).

[40] Aditya Golatkar, Alessandro Achille, and Stefano Soatto. 2020. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Computer Vision Foundation/IEEE, 9301–9309. DOI: http://dx.doi.org/10.1109/CVPR42600.2020.00932

[41] Chuan Guo, Tom Goldstein, Awni Y. Hannun, and Laurens van der Maaten. 2020. Certified data removal from machine learning models. In *37th International Conference on Machine Learning (Proceedings of Machine Learning Research*, Vol. 119). PMLR, 3832–3842. Retrieved from http://proceedings.mlr.press/v119/guo20c.html.

[42] Ayush Sekhari, Jayadev Acharya, Gautam Kamath, and Ananda Theertha Suresh. 2021. Remember what you want to forget: Algorithms for machine unlearning. *CoRR* abs/2103.03279 (2021).

[43] Aditya Golatkar, Alessandro Achille, and Stefano Soatto. 2020. Forgetting outside the box: Scrubbing deep networks of information accessible from input-output observations. In *16th European Conference on Computer Vision (Lecture Notes in Computer Science*, Vol. 12374). Springer, 383–398. DOI: http://dx.doi.org/10.1007/978-3-030-58526-6_23

[44] David Marco Sommer, Liwei Song, Sameer Wagh, and Prateek Mittal. 2020. Towards probabilistic verification of machine unlearning. *CoRR* abs/2003.04247 (2020).

[45] Laura Graves, Vineel Nagisetty, and Vijay Ganesh. 2021. Amnesiac machine learning. In *35th AAAI Conference on Artificial Intelligence, 33rd Conference on Innovative Applications of Artificial Intelligence, 11th Symposium on Educational Advances in Artificial Intelligence*. AAAI Press, 11516–11524. Retrieved from https://ojs.aaai.org/index.php/AAAI/article/view/17371.

[46] Daniel L. Felps, Amelia D. Schwickerath, Joyce D. Williams, Trung N. Vuong, Alan Briggs, Matthew Hunt, Evan Sakmar, David D. Saranchak, and Tyler Shumaker. 2021. Class clown: Data redaction in machine unlearning at enterprise scale. In *10th International Conference on Operations Research and Enterprise Systems*. SCITEPRESS, 7–14. DOI: http://dx.doi.org/10.5220/0010419600070014

[47] Min Chen, Zhikun Zhang, Tianhao Wang, Michael Backes, Mathias Humbert, and Yang Zhang. 2022. Graph unlearning. In *ACM SIGSAC Conference on Computer and Communications Security*. ACM, 499–513. DOI: http://dx.doi.org/10.1145/3548606.3559352

[48] Yingzhe He, Guozhu Meng, Kai Chen, Jinwen He, and Xingbo Hu. 2021. DeepObliviate: A powerful charm for erasing data residual memory in deep neural networks. *CoRR* abs/2105.06209 (2021).

[49] Varun Gupta, Christopher Jung, Seth Neel, Aaron Roth, Saeed Sharifi-Malvajerdi, and Chris Waites. 2021. Adaptive machine unlearning. In *Advances in Neural Information Processing Systems: Annual Conference on Neural Information Processing Systems*. 16319–16330. Retrieved from https://proceedings.neurips.cc/paper/2021/hash/87f7ee4fdb57bdfd52179947211b7ebb-Abstract.html.

[50] Seth Neel, Aaron Roth, and Saeed Sharifi-Malvajerdi. 2021. Descent-to-delete: Gradient-based methods for machine unlearning. In *Algorithmic Learning Theory (Proceedings of Machine Learning Research*, Vol. 132). PMLR, 931–962. Retrieved from http://proceedings.mlr.press/v132/neel21a.html.

[51] Zachary Izzo, Mary Anne Smart, Kamalika Chaudhuri, and James Zou. 2021. Approximate data deletion from machine learning models. In *24th International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research*, Vol. 130). PMLR, 2008–2016. Retrieved from http://proceedings.mlr.press/v130/izzo21a.html.

[52] Aditya Golatkar, Alessandro Achille, Avinash Ravichandran, Marzia Polito, and Stefano Soatto. 2021. Mixed-privacy forgetting in deep networks. In *IEEE Conference on Computer Vision and Pattern Recognition*. Computer Vision Foundation/IEEE, 792–801. Retrieved from https://openaccess.thecvf.com/content/CVPR2021/html/Golatkar_Mixed-Privacy_Forgetting_in_Deep_Networks_CVPR_2021_paper.html.

[53] Alexander Warnecke, Lukas Pirch, Christian Wressnegger, and Konrad Rieck. 2021. Machine unlearning of features and labels. *CoRR* abs/2108.11577 (2021).

[54] Gaoyang Liu, Xiaoqiang Ma, Yang Yang, Chen Wang, and Jiangchuan Liu. 2021. FedEraser: Enabling efficient client-level data removal from federated learning models. In *29th IEEE/ACM International Symposium on Quality of Service*. IEEE, 1–10. DOI: http://dx.doi.org/10.1109/IWQOS52092.2021.9521274

[55] Junxiao Wang, Song Guo, Xin Xie, and Heng Qi. 2022. Federated unlearning via class-discriminative pruning. In *ACM Web Conference*. ACM, 622–632. DOI: http://dx.doi.org/10.1145/3485447.3512222

[56] Thomas Baumhauer, Pascal Schöttle, and Matthias Zeppelzauer. 2020. Machine unlearning: Linear filtration for logit-based classifiers. *CoRR* abs/2002.02730 (2020).

[57] Sebastian Schelter, Stefan Grafberger, and Ted Dunning. 2021. HedgeCut: Maintaining randomised trees for low-latency machine unlearning. In *International Conference on Management of Data*. ACM, 1545–1557. DOI: http://dx.doi.org/10.1145/3448016.3457239

[58] Jonathan Brophy and Daniel Lowd. 2021. Machine unlearning for random forests. In *38th International Conference on Machine Learning* (*Proceedings of Machine Learning Research*, Vol. 139). PMLR, 1092–1104. Retrieved from http://proceedings.mlr.press/v139/brophy21a.html.

[59] Kongyang Chen, Yao Huang, and Yiwen Wang. 2021. Machine unlearning via GAN. *CoRR* abs/2111.11869 (2021).

[60] Yinjun Wu, Edgar Dobriban, and Susan B. Davidson. 2020. DeltaGrad: Rapid retraining of machine learning models. In *37th International Conference on Machine Learning (Proceedings of Machine Learning Research*, Vol. 119). PMLR, 10355–10366. Retrieved from http://proceedings.mlr.press/v119/wu20b.html.

[61] Huaizheng Zhang, Yuanming Li, Yizheng Huang, Yonggang Wen, Jianxiong Yin, and Kyle Guan. 2020. MLModelCI: An automatic cloud platform for efficient MLaaS. In *28th ACM International Conference on Multimedia*. ACM, 4453–4456. DOI: http://dx.doi.org/10.1145/3394171.3414535

[62] Hailong Hu and Jun Pang. 2021. Membership inference attacks against GANs by leveraging over-representation regions. In *SIGSAC Conference on Computer and Communications Security*. ACM, 2387–2389. DOI: http://dx.doi.org/10.1145/3460120.3485338

[63] PengFei Zhang, Guangdong Bai, Zi Huang, and Xin-Shun Xu. 2022. Machine unlearning for image retrieval: A generative scrubbing approach. In *30th ACM International Conference on Multimedia*. ACM, 237–245. DOI: http://dx.doi.org/10.1145/3503161.3548378

[64] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. 2018. Privacy risk in machine learning: Analyzing the connection to overfitting. In *31st IEEE Computer Security Foundations Symposium*. IEEE Computer Society, 268–282. DOI: http://dx.doi.org/10.1109/CSF.2018.00027

[65] Nguyen Van Huynh, Dinh Thai Hoang, Diep N. Nguyen, and Eryk Dutkiewicz. 2022. Joint coding and scheduling optimization for distributed learning over wireless edge networks. *IEEE J. Sel. Areas Commun.* 40, 2 (2022), 484–498. DOI: http://dx.doi.org/10.1109/JSAC.2021.3118432

[66] Cristiano Gratton, Naveen K. D. Venkategowda, Reza Arablouei, and Stefan Werner. 2022. Privacy-preserved distributed learning with zeroth-order optimization. *IEEE Trans. Inf. Forens. Secur.* 17 (2022), 265–279. DOI: http://dx.doi.org/10.1109/TIFS.2021.3139267

[67] João Carlos Xavier Junior, Alex Alves Freitas, Teresa Bernarda Ludermir, Antonino Feitosa Neto, and Cephas A. S. Barreto. 2020. An evolutionary algorithm for automated machine learning focusing on classifier ensembles: An improved algorithm and extended results. *Theor. Comput. Sci.* 805 (2020), 1–18. DOI: http://dx.doi.org/10.1016/j.tcs.2019.12.002

[68] Wei Qian, Chenxu Zhao, Huajie Shao, Minghan Chen, Fei Wang, and Mengdi Huai. 2022. Patient similarity learning with selective forgetting. In *IEEE International Conference on Bioinformatics and Biomedicine*. IEEE, 529–534. DOI: http://dx.doi.org/10.1109/BIBM55620.2022.9995016

[69] Haonan Yan, Xiaoguang Li, Ziyao Guo, Hui Li, Fenghua Li, and Xiaodong Lin. 2022. ARCANE: An efficient architecture for exact machine unlearning. In *31st International Joint Conference on Artificial Intelligence*. ijcai.org, 4006–4013. DOI: http://dx.doi.org/10.24963/ijcai.2022/556

[70] Chao Huang, Huance Xu, Yong Xu, Peng Dai, Lianghao Xia, Mengyin Lu, Liefeng Bo, Hao Xing, Xiaoping Lai, and Yanfang Ye. 2021. Knowledge-aware coupled graph neural network for social recommendation. In *35th AAAI Conference on Artificial Intelligence, 33rd Conference on Innovative Applications of Artificial Intelligence, 11th Symposium on Educational Advances in Artificial Intelligence*. AAAI Press, 4115–4122. Retrieved from https://ojs.aaai.org/index.php/AAAI/article/view/16533.

[71] Dawei Cheng, Fangzhou Yang, Sheng Xiang, and Jin Liu. 2022. Financial time series forecasting with multi-modality graph neural network. *Pattern Recognit.* 121 (2022), 108218. DOI: http://dx.doi.org/10.1016/j.patcog.2021.108218

[72] Ziynet Nesibe Kesimoglu and Serdar Bozdag. 2021. SUPREME: A cancer subtype prediction methodology integrating multiple biological datatypes using graph convolutional neural networks. In *12th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics*. ACM, 83:1. DOI: http://dx.doi.org/10.1145/3459930.3470853

[73] Zulong Diao, Xin Wang, Dafang Zhang, Yingru Liu, Kun Xie, and Shaoyao He. 2019. Dynamic spatial-temporal graph convolutional neural networks for traffic forecasting. In *33rd AAAI Conference on Artificial Intelligence, 31st Innovative Applications of Artificial Intelligence Conference, 9th AAAI Symposium on Educational Advances in Artificial Intelligence*. AAAI Press, 890–897. DOI: http://dx.doi.org/10.1609/aaai.v33i01.3301890

[74] Luís C. Lamb, Artur S. d'Avila Garcez, Marco Gori, Marcelo O. R. Prates, Pedro H. C. Avelar, and Moshe Y. Vardi. 2020. Graph neural networks meet neural-symbolic computing: A survey and perspective. In *29th International Joint Conference on Artificial Intelligence*. ijcai.org, 4877–4884. DOI: http://dx.doi.org/10.24963/ijcai.2020/679

[75] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. 2021. A comprehensive survey on graph neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* 32, 1 (2021), 4–24. DOI: http://dx.doi.org/10.1109/TNNLS.2020.2978386

[76] Dongxiao He, Youyou Wang, Jinxin Cao, Weiping Ding, Shizhan Chen, Zhiyong Feng, Bo Wang, and Yuxiao Huang. 2021. A network embedding-enhanced Bayesian model for generalized community detection in complex networks. *Inf. Sci.* 575 (2021), 306–322. DOI: http://dx.doi.org/10.1016/j.ins.2021.06.020

[77] Zhenwen Ren, Quansen Sun, and Dong Wei. 2021. Multiple kernel clustering with kernel k-means coupled graph tensor learning. In *35th AAAI Conference on Artificial Intelligence, 33rd Conference on Innovative Applications of Artificial Intelligence, 11th Symposium on Educational Advances in Artificial Intelligence*. AAAI Press, 9411–9418. Retrieved from https://ojs.aaai.org/index.php/AAAI/article/view/17134.

[78] Min Chen, Zhikun Zhang, Tianhao Wang, Michael Backes, Mathias Humbert, and Yang Zhang. 2021. When machine unlearning jeopardizes privacy. In *ACM SIGSAC Conference on Computer and Communications Security*. ACM, 896–911. DOI: http://dx.doi.org/10.1145/3460120.3484756

[79] Cynthia Dwork and Vitaly Feldman. 2018. Privacy-preserving prediction. *CoRR* abs/1803.10266 (2018).

[80] Gavin Brown, Mark Bun, Vitaly Feldman, Adam D. Smith, and Kunal Talwar. 2021. When is memorization of irrelevant training data necessary for high-accuracy learning? In *53rd Annual ACM SIGACT Symposium on Theory of Computing*. ACM, 123–132. DOI: http://dx.doi.org/10.1145/3406325.3451131

[81] C.-K. Peng, S. V. Buldyrev, S. Havlin, M. Simons, H. E. Stanley, and A. L. Goldberger. 1994. Mosaic organization of DNA nucleotides. *Phys. Rev. E* 49, 2 (Feb. 1994), 1685–1689. DOI: http://dx.doi.org/10.1103/PhysRevE.49.1685

[82] Rituparna Sinha, Rajat Kumar Pal, and Rajat K. De. 2022. GenSeg and MR-GenSeg: A novel segmentation algorithm and its parallel MapReduce based approach for identifying genomic regions with copy number variations. *IEEE ACM Trans. Comput. Biol. Bioinform.* 19, 1 (2022), 443–454. DOI: http://dx.doi.org/10.1109/TCBB.2020.3000661

[83] YooJung Choi, Golnoosh Farnadi, Behrouz Babaki, and Guy Van den Broeck. 2020. Learning fair naive Bayes classifiers by discovering and eliminating discrimination patterns. In *34th AAAI Conference on Artificial Intelligence, 32nd Innovative Applications of Artificial Intelligence Conference, 10th AAAI Symposium on Educational Advances in Artificial Intelligence*. AAAI Press, 10077–10084. Retrieved from https://aaai.org/ojs/index.php/AAAI/article/view/6565.

[84] Allan Grønlund, Lior Kamma, and Kasper Green Larsen. 2020. Near-tight margin-based generalization bounds for support vector machines. In *37th International Conference on Machine Learning (Proceedings of Machine Learning Research*, Vol. 119). PMLR, 3779–3788. Retrieved from http://proceedings.mlr.press/v119/gronlund20a.html.

[85] Shuyin Xia, Daowan Peng, Deyu Meng, Elisabeth Giem, Wei Wei, and Zizhong Chen. 2022. Ball kk-Means: Fast adaptive clustering with no bounds. *IEEE Trans. Pattern Anal. Mach. Intell.* 44, 1 (2022), 87–99. DOI: http://dx.doi.org/10.1109/TPAMI.2020.3008694

[86] Takashi Shibata, Go Irie, Daiki Ikami, and Yu Mitsuzumi. 2021. Learning with selective forgetting. In *30th International Joint Conference on Artificial Intelligence*. ijcai.org, 989–996. DOI: http://dx.doi.org/10.24963/ijcai.2021/137

[87] Joseph A. Calandrino, Ann Kilzer, Arvind Narayanan, Edward W. Felten, and Vitaly Shmatikov. 2011. "You might also like:" Privacy risks of collaborative filtering. In *32nd IEEE Symposium on Security and Privacy*. IEEE Computer Society, 231–246. DOI: http://dx.doi.org/10.1109/SP.2011.40

[88] Di Wang, Marco Gaboardi, Adam D. Smith, and Jinhui Xu. 2020. Empirical risk minimization in the non-interactive local model of differential privacy. *J. Mach. Learn. Res.* 21 (2020), 200:1–200:39. Retrieved from http://jmlr.org/papers/v21/19-253.html.

[89] Naman Agarwal, Brian Bullins, and Elad Hazan. 2017. Second-order stochastic optimization for machine learning in linear time. *J. Mach. Learn. Res.* 18 (2017), 116:1–116:40. Retrieved from http://jmlr.org/papers/v18/16-491.html.

[90] James Martens. 2020. New insights and perspectives on the natural gradient method. *J. Mach. Learn. Res.* 21 (2020), 146:1–146:76. Retrieved from http://jmlr.org/papers/v21/17-678.html.

[91] Arthur Jacot, Franck Gabriel, and Clément Hongler. 2021. Neural tangent kernel: Convergence and generalization in neural networks (invited paper). In *53rd Annual ACM SIGACT Symposium on Theory of Computing*. ACM. DOI: http://dx.doi.org/10.1145/3406325.3465355

[92] Baihe Huang, Xiaoxiao Li, Zhao Song, and Xin Yang. 2021. FL-NTK: A neural tangent kernel-based framework for federated learning analysis. In *38th International Conference on Machine Learning (Proceedings of Machine Learning Research*, Vol. 139). PMLR, 4423–4434. Retrieved from http://proceedings.mlr.press/v139/huang21c.html.

[93] Yi Liu, Lei Xu, Xingliang Yuan, Cong Wang, and Bo Li. 2022. The right to be forgotten in federated learning: An efficient realization with rapid retraining. In *IEEE Conference on Computer Communications*. IEEE, 1749–1758. DOI: http://dx.doi.org/10.1109/INFOCOM48880.2022.9796721

[94] Wei Yang Bryan Lim, Nguyen Cong Luong, Dinh Thai Hoang, Yutao Jiao, Ying-Chang Liang, Qiang Yang, Dusit Niyato, and Chunyan Miao. 2020. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Commun. Surv. Tutor.* 22, 3 (2020), 2031–2063. DOI: http://dx.doi.org/10.1109/COMST.2020.2986024

[95] Ahmed Imteaj, Urmish Thakker, Shiqiang Wang, Jian Li, and M. Hadi Amini. 2022. A survey on federated learning for resource-constrained IoT devices. *IEEE Internet Things J.* 9, 1 (2022), 1–24. DOI: http://dx.doi.org/10.1109/JIOT.2021.3095077

[96] Ankit Thakkar and Kinjal Chaudhari. 2020. Predicting stock trend using an integrated term frequency-inverse document frequency-based feature weight matrix with neural networks. *Appl. Soft Comput.* 96 (2020), 106684. DOI: http://dx.doi.org/10.1016/j.asoc.2020.106684

[97] Shaopeng Guo, Yujie Wang, Quanquan Li, and Junjie Yan. 2020. DMCP: Differentiable Markov channel pruning for neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Computer Vision Foundation/IEEE, 1536–1544. DOI: http://dx.doi.org/10.1109/CVPR42600.2020.00161

[98] Wei Shen, Yilu Guo, Yan Wang, Kai Zhao, Bo Wang, and Alan L. Yuille. 2021. Deep differentiable random forests for age estimation. *IEEE Trans. Pattern Anal. Mach. Intell.* 43, 2 (2021), 404–419. DOI: http://dx.doi.org/10.1109/TPAMI.2019.2937294

[99] Shlomi Maliah and Guy Shani. 2021. Using POMDPs for learning cost sensitive decision trees. *Artif. Intell.* 292 (2021), 103400. DOI: http://dx.doi.org/10.1016/j.artint.2020.103400

[100] Hongju Cheng, Yushi Shi, Leihuo Wu, Yingya Guo, and Naixue Xiong. 2021. An intelligent scheme for big data recovery in internet of things based on multi-attribute assistance and extremely randomized trees. *Inf. Sci.* 557 (2021), 66–83. DOI: http://dx.doi.org/10.1016/j.ins.2020.12.041

[101] Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom B. Brown, Dawn Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. 2021. Extracting training data from large language models. In *30th USENIX Security Symposium*. USENIX Association, 2633–2650. Retrieved from https://www.usenix.org/conference/usenixsecurity21/presentation/carlini-extracting.

[102] Aryan Mokhtari and Alejandro Ribeiro. 2015. Global convergence of online limited memory BFGS. *J. Mach. Learn. Res.* 16 (2015), 3151–3181. Retrieved from http://dl.acm.org/citation.cfm?id=2912100.

[103] Yuma Koizumi, Shin Murata, Noboru Harada, Shoichiro Saito, and Hisashi Uematsu. 2019. SNIPER: Few-shot learning for anomaly detection to minimize false-negative rate with ensured true-positive rate. In *IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 915–919. DOI: http://dx.doi.org/10.1109/ICASSP.2019.8683667

[104] Ga Wu, Masoud Hashemi, and Christopher Srinivasa. 2022. PUMA: Performance Unchanged Model Augmentation for training data removal. In *36th AAAI Conference on Artificial Intelligence, 34th Conference on Innovative Applications of Artificial Intelligence, 12th Symposium on Educational Advances in Artificial Intelligence*. AAAI Press, 8675–8682. Retrieved from https://ojs.aaai.org/index.php/AAAI/article/view/20846.

[105] Vikram S. Chundawat, Ayush K. Tarun, Murari Mandal, and Mohan S. Kankanhalli. 2022. Zero-shot machine unlearning. *CoRR* abs/2201.05629 (2022).

[106] Santiago Zanella Béguelin, Lukas Wutschitz, Shruti Tople, Victor Rühle, Andrew Paverd, Olga Ohrimenko, Boris Köpf, and Marc Brockschmidt. 2020. Analyzing information leakage of updates to natural language models. In *ACM SIGSAC Conference on Computer and Communications Security*. ACM, 363–375. DOI: http://dx.doi.org/10.1145/3372297.3417880

[107] Shruti Tople, Marc Brockschmidt, Boris Köpf, Olga Ohrimenko, and Santiago Zanella Béguelin. 2019. Analyzing privacy loss in updates of natural language models. *CoRR* abs/1912.07942 (2019).

[108] Neil G. Marchant, Benjamin I. P. Rubinstein, and Scott Alfeld. 2022. Hard to forget: Poisoning attacks on certified machine unlearning. In *36th AAAI Conference on Artificial Intelligence, 34th Conference on Innovative Applications of Artificial Intelligence, 12th Symposium on Educational Advances in Artificial Intelligence*. AAAI Press, 7691–7700. Retrieved from https://ojs.aaai.org/index.php/AAAI/article/view/20736.

[109] Xiao Liu and Sotirios A. Tsaftaris. 2020. Have you forgotten? A method to assess if machine learning models have forgotten data. In *23rd International Conference on Medical Image Computing and Computer Assisted Intervention*. Springer, 95–105. DOI: http://dx.doi.org/10.1007/978-3-030-59710-8_10

[110] Anvith Thudi, Hengrui Jia, Ilia Shumailov, and Nicolas Papernot. 2021. On the necessity of auditable algorithmic definitions for machine unlearning. *CoRR* abs/2110.11891 (2021).

[111] Yang Liu, Zhuo Ma, Yilong Yang, Ximeng Liu, Jianfeng Ma, and Kui Ren. 2022. RevFRF: Enabling cross-domain random forest training with revocable federated learning. *IEEE Trans. Depend. Secur. Comput.* 19, 6 (2022), 3671–3685. DOI: http://dx.doi.org/10.1109/TDSC.2021.3104842

[112] Yang Liu, Mingyuan Fan, Cen Chen, Ximeng Liu, Zhuo Ma, Li Wang, and Jianfeng Ma. 2022. Backdoor defense with machine unlearning. In *IEEE Conference on Computer Communications*. IEEE, 280–289. DOI: http://dx.doi.org/10.1109/INFOCOM48880.2022.9796974

[113] Gunter Schlageter. 1978. Process synchronization in database systems. *ACM Trans. Datab. Syst.* 3, 3 (1978), 248–271. DOI: http://dx.doi.org/10.1145/320263.320279

[114] Rajive L. Bagrodia. 1989. Process synchronization: Design and performance evaluation of distributed algorithms. *IEEE Trans. Softw. Eng.* 15, 9 (1989), 1053–1065. DOI: http://dx.doi.org/10.1109/32.31364

[115] Shiqiang Wang, Tiffany Tuor, Theodoros Salonidis, Kin K. Leung, Christian Makaya, Ting He, and Kevin Chan. 2019. Adaptive federated learning in resource constrained edge computing systems. *IEEE J. Sel. Areas Commun.* 37, 6 (2019), 1205–1221. DOI: http://dx.doi.org/10.1109/JSAC.2019.2904348

[116] Latif U. Khan, Walid Saad, Zhu Han, Ekram Hossain, and Choong Seon Hong. 2021. Federated learning for internet of things: Recent advances, taxonomy, and open challenges. *IEEE Commun. Surv. Tutor.* 23, 3 (2021), 1759–1799. DOI: http://dx.doi.org/10.1109/COMST.2021.3090430

[117] Peiying Zhang, Yaqi Wang, Neeraj Kumar, Chunxiao Jiang, and Guowei Shi. 2022. A security- and privacy-preserving approach based on data disturbance for collaborative edge computing in social IoT systems. *IEEE Trans. Comput. Soc. Syst.* 9, 1 (2022), 97–108. DOI: http://dx.doi.org/10.1109/TCSS.2021.3092746

[118] Kamalika Chaudhuri, Claire Monteleoni, and Anand D. Sarwate. 2011. Differentially private empirical risk minimization. *J. Mach. Learn. Res.* 12 (2011), 1069–1109. Retrieved from http://dl.acm.org/citation.cfm?id=2021036.

[119] Xiaofeng Ding, Hongbiao Fang, Zhilin Zhang, Kim-Kwang Raymond Choo, and Hai Jin. 2022. Privacy-preserving feature extraction via adversarial training. *IEEE Trans. Knowl. Data Eng.* 34, 4 (2022), 1967–1979. DOI: http://dx.doi.org/10.1109/TKDE.2020.2997604

[120] Bo Zhang, Hancheng Ye, Gang Yu, Bin Wang, Yike Wu, Jiayuan Fan, and Tao Chen. 2022. Sample-centric feature generation for semi-supervised few-shot learning. *IEEE Trans. Image Process.* 31 (2022), 2309–2320. DOI: http://dx.doi.org/10.1109/TIP.2022.3154938

[121] Jiajing Zhu, Yongguo Liu, Chuanbiao Wen, and Xindong Wu. 2022. DGDFS: Dependence Guided Discriminative Feature Selection for predicting adverse drug-drug interaction. *IEEE Trans. Knowl. Data Eng.* 34, 1 (2022), 271–285. DOI: http://dx.doi.org/10.1109/TKDE.2020.2978055

[122] Stefanos Leonardos and Georgios Piliouras. 2022. Exploration-exploitation in multi-agent learning: Catastrophe theory meets game theory. *Artif. Intell.* 304 (2022), 103653. DOI: http://dx.doi.org/10.1016/j.artint.2021.103653

[123] Lei Cui, Youyang Qu, Mohammad Reza Nosouhi, Shui Yu, Jianwei Niu, and Gang Xie. 2019. Improving data utility through game theory in personalized differential privacy. *J. Comput. Sci. Technol.* 34, 2 (2019), 272–286. DOI: http://dx.doi.org/10.1007/s11390-019-1910-3

[124] Xueqin Liang, Zheng Yan, Robert H. Deng, and Qinghua Zheng. 2021. Investigating the adoption of hybrid encrypted cloud data deduplication with game theory. *IEEE Trans. Parallel Distrib. Syst.* 32, 3 (2021), 587–600. DOI: http://dx.doi.org/10.1109/TPDS.2020.3028685

Received 3 May 2022; revised 5 March 2023; accepted 31 May 2023