Towards Reliable and Generalizable Differentially Private Machine Learning (Extended Version)

Wenxuan Bao
University of Florida
Gainesville, USA
wenxuanbao@ufl.edu

Vincent Bindschaedler
University of Florida
Gainesville, USA
vbindschaedler@ufl.edu

***Abstract*—There is a flurry of recent research papers proposing novel differentially private machine learning (DPML) techniques. These papers claim to achieve new state-of-the-art (SoTA) results and offer empirical results as validation. However, there is no consensus on which techniques are most effective or if they genuinely meet their stated claims. Complicating matters, heterogeneity in codebases, datasets, methodologies, and model architectures make direct comparisons of different approaches challenging.**

In this paper, we conduct a reproducibility and replicability (R+R) experiment on 11 different SoTA DPML techniques from the recent research literature. Results of our investigation are varied: while some methods stand up to scrutiny, others falter when tested outside their initial experimental conditions. We also discuss challenges unique to the reproducibility of DPML, including additional randomness due to DP noise, and how to address them. Finally, we derive insights and best practices to obtain scientifically valid and reliable results.

***Index Terms*—Differential Privacy, Machine Learning, Reproducibility**

1. Introduction*

The reproducibility crisis that plagues machine learning (ML) and ML-based science is well-documented [1], [2], [3]. Due to the breadth and diversity of machine learning applications, studies are conducted within specific domains and application areas such as healthcare [4], [5], life sciences [6], and security [7], [8].

The reproducibility of differentially private machine learning (DPML) has so far received little attention. In a nutshell, DPML seeks to protect the privacy of training data of a model using the mathematical framework of differential privacy (DP) [9], [10], [11]. While machine learning models are typically trained with Stochastic Gradient Descent [12] (SGD), DPML primarily leverages DP-SGD [13], a drop-in replacement for SGD. DP-SGD iteratively updates model parameters using the gradient like SGD, but also clips individual training points' gradients prior to aggregation, and adds Gaussian noise to updates. DP-SGD provably satisfies differential privacy, but the model predictions quality is often drastically degraded.

Recently, numerous research papers proposing novel DPML techniques claim to achieve new state-of-the-art (SoTA) results [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24]. These techniques and the DPML literature more broadly have delivered remarkable improvements in the utility-privacy tradeoff since DP-SGD was introduced by Abadi et al [13] in 2016. Despite this, there is little consensus regarding which techniques are most effective. This is in part the result of the heterogeneity in codebases, datasets and model architecture, making apples-to-apples comparisons difficult to obtain.

In this paper, we systematically survey, taxonomize, and discuss the research literature on differentially private machine learning (DPML). Our goal is to explicate the various competing approaches that have been proposed and identify promising directions for future work.

Then we conduct a reproducibility and replicability (R+R) experiment on 11 different recent DPML techniques. We focus on centralized machine learning — leaving federated learning for future work — and computer vision tasks, since these have received significant recent attention and show great promise.

However, the purpose of our investigation is not to point fingers or cast any specific work in a negative light. We seek to understand what methodological steps make DPML research reproducible and replicable and lead to reliable findings. We discover that a significant challenge with DPML reproducibility — as opposed to (non-private) ML reproducibility — is the additional randomness (e.g., noise added to gradients, etc.). Variability in measured results is often substantial, especially when few runs are performed and (or) when few datasets/models are used. Averaging results of multiple runs could alleviate this issue. But DPML training is much slower than non-private ML training [25], [26], [20] so performing extensive evaluation is a major computational burden.

The net effect of this additional randomness is that reliable results are more difficult to obtain. A single lucky run

∗. This paper is published at ACSAC 2024. This is the extended version that includes an overview of the relevant literature. We open-source our codebase at: https://github.com/wenxuan-Bao/Reliable-and-Generalizable-DPML.

with higher performance than the baseline may be (wrongly) interpreted as a new SoTA result.

To overcome this, we propose a framework based on paired t-tests [27] and Cohen's d [28]. This framework allows us to determine which of our selected techniques indeed outperform their baselines and also quantifies the additional variability of DPML compared to non-private ML.

Stepping back, the concrete goals of our investigation are threefold: (1) quantify the reproducibility of existing work to confirm or disconfirm SoTA claims, thereby separating the wheat from the chaff; (2) identify which techniques provide improvements that are scientifically sound and generalize beyond the (necessarily) narrow experimental setting of their originating papers; and (3) establish guidelines and best practices that future research can adopt to maximize fair comparisons and reduce false discovery risk.

The results of our investigations are mixed. Obtaining implementations of the selected techniques was not a problem. Codebases were readily available in many cases and when they were not, techniques could easily be implemented based on research papers' descriptions. We were able to directly obtain results consistent with what all 11 selected papers reported. However, when using the techniques outside of the narrow experimental settings of their original papers (e.g., on a different dataset or with a different model architecture), only 7 out of 11 completely delivered their claimed improvements.

Among those techniques that disappointed in new experimental settings, we found notable methodological pitfalls such as a lack of ablation studies, narrow sets of evaluation tasks, and results reported for a single run only.

To avoid these pitfalls in future research and maximize the chance of reproducibility and replicability, we derive guidelines and a concrete checklist.

Summary of contributions:

- In this paper, we systematically survey, taxonomize, and discuss the research literature on DPML. We show our taxonomy of DPML in Table 1
- We conduct a thorough reproducibility and replicability evaluation of 11 recent SoTA DPML techniques, showing their variance in generalizability and reliability.
- We introduce a framework utilizing paired t-tests, specifically tailored to assess the inherent variability and reproducibility challenges unique to DPML.
- We propose comprehensive guidelines and a checklist to enhance future DPML research, aiming to standardize practices and reduce the prevalence of invalid claims.

2. Background & Related Work

To ensure the paper is self-contained, we provide background on supervised learning, ML reproducibility, and DPML. Readers may also be interested in recent surveys on DPML [113], [114], [115], [116], [117], [118], [119].

2.1. Supervised Learning

A model maps data points (e.g., images) into (predicted) labels. The model itself is represented by a parameter vector and it is trained by minimizing a loss function that measures the discrepancy between predicted and actual labels of a training dataset. This is done using an optimization algorithm such as Stochastic Gradient Descent (SGD) [12].

Stochastic Gradient Descent (SGD). SGD is an iterative optimization procedure that has been extensively studied and has many variants [120]. Its mini-batch version is most commonly used and the one we refer to (unless otherwise stated) as SGD. It is illustrated in Algorithm 1. Informally, the algorithm computes the gradient of the loss with respect to the parameter vector and iteratively updates the current solution accordingly until convergence.

2.2. Transfer Learning

Transfer learning is the process of adapting learned knowledge on one task or domain to another task or domain. We refer the reader to Weiss et al. [121] for a comprehensive discussion of the topic. For us, it suffices to note that the typical workflow for transfer learning is to use a model (e.g., a neural network) that is pre-trained on some dataset, change its last few layers (e.g., add a classification head), and then run the training process on the new task to fine-tune the existing pre-trained layers alongside with the new layers.

2.3. ML Reproducibilty

There are notable recent concerns regarding reproducibility in machine learning [1], [6], [2], [3]. For instance, Kapoor et al. [1] highlight that the reproducibility crisis in ML, particularly the problem of data leakage. Tatman et al. [122] suggest a taxonomy for reproducibility with three levels. Raff et al. [123] focus on independent reproducibility, where they implement 255 papers and record the features of each paper. They find that papers with a greater empirical focus are more reproducible. Pineau et al. [2] report on the NeurIPS 2019 reproducibility program and discuss the various factors that can lead to unreliable or false results. Heil et al. [6] propose reproducibility standards for machine learning in the life sciences.

2.4. Differential Privacy and Variants

Differential privacy (DP) [11] is defined as follows.

Definition 1. *A randomized algorithm F is said to satisfy* (ε, δ)-differential privacy *if for any neighboring datasets D_0, D_1 and any output set $S \subseteq \mathrm{Range}(F)$, it holds that:*

$$\Pr(F(D_0) \in S) \leq \exp(\varepsilon)\ \Pr(F(D_1) \in S) + \delta\ ,$$

where probabilities are taken over randomness in F, and $\varepsilon > 0$ is called the *privacy budget*. The smaller ε is the more stringent the privacy guarantee. We must also ensure

TABLE 1: Taxonomy of Methods.

Category	References	Section
Tuning model architecture	[25] , [29], [30], [31],[32], [33], [34],[14], [15]	Section 3.1
Feature selection	[13], [19], [25], [35]	Section 3.2
Hyperparameter tuning	[25], [17], [15] ,[36], [16], [37],[18],[38]	Section 3.3
Gradient clipping	[13], [39], [40], [41], [20], [42], [43], [44], [45], [46], [47], [21], [48], [49], [50],[51], [52], [53], [54], [55], [56], [57]	Section 4
Transfer learning	[58], [22], [59], [47],[60], [23], [61], [62], [63], [64], [65], [24], [66], [67],[68]	Section 5
Sparse Training	[69], [70], [71], [25], [72], [73], [74], [75], [76], [77], [78] ,[79], [80]	Section 6.1
Better privacy accounting	[81] , [82],[83], [84], [85], [86],[84], [87], [88],	Section 6.2
DP Auditing	[89],[90],[91], [92], [93]	Section 6.3
Learning Process Modifications	[13], [94], [95], [96], [16], [97], [98], [99], [26], [62], [20], [100], [101], [102], [103] ,[104], [105], [106][107]	Section 6.4
Teacher-Student models	[108], [109], [110], [111], [112]	Section 6.5

TABLE 2: Performance comparison among recent works that claim they have SoTA results. Note that these works usually report results for different ε and datasets. Here we attempt to summarize them for two datasets: CIFAR-10 and ImageNet, and similar levels of privacy setting in terms of ε and δ. We observe that modifying existing model architectures, hyper-parameters tuning with training tricks, and gradient clipping techniques can result in the highest performance in terms of accuracy. That said, other directions are still worth exploring to achieve better performance in terms of running time or memory usage.

Category	Reference	Dataset	Pre-train dataset	ε	σ	DP test acc.
Better privacy accounting	Ye et al. [88]	CIFAR-10	No	3	10^-5	69.30%
Hyper-parameter selection	Dormann et al. [16]			7.42		70.10%
Model architecture search	Cheng et al. [30]			3		68.33%
	Remerscheid et al. [31]			7		73.50%
Activation function modification	Papernot et al. [34]			7.53		66.20%
Existing model modification	Klause et al. [14]			8		82.50%
Data augmentation	De et al. [17]			8		81.40%
Feature selection	Tramèr and Boneh [19]		ImageNet (unlabeled)	2		92.70%
Gradient clipping	Bu et al. [20]		ImageNet1k	1		96.70%
	Bu et al. [21]		ImageNet (unlabeled)	2		92.70%
Teacher-Student model	Zhu et al. [111]		Subset of CIFAR-10 (unlabeled)	2.92		70.80%
Fine-tune technique	Tang et al. [24]		Random process data	1		72.32%
	Cattan et al. [22]		ImageNet	1		95.00%
	Luo et al. [23]		CIFAR-100	1.5		81.57%
	Amid et al. [60]		Subset of CIFAR-10	3.51		67.03%
	Mehta et al. [58]	ImageNet	JFT-300M	4	$8 * 10^-7$	81.70%
Hyper-parameters tuning	Sander et al. [15]		No	8		39.20%

that δ is sufficiently small. In the definition, D_0 and D_1 are neighboring if one can be obtained from the other by adding exactly one example.

An important concept for DP algorithms is sensitivity. Informally, sensitivity measures the maximum change in the output caused by the inclusion/exclusion of exactly one example. The (global) *sensitivity* of a function g can be denoted as Δg and is defined as

$$\Delta g = \max ||g(D_0) - g(D_1)||,$$

where the maximum is taken over pairs of neighboring datasets D_0, D_1 and $||\cdot||$ denotes a norm like the l_1-norm or the l_2-norm.

Rényi DP (RDP). There are several variants (both generalization and relaxations) of differential privacy [124], [125], [126] such as approximate DP [10], Rényi DP (RDP) [127]. Among those DP variants, Rényi DP is the most widely used. It is based on Rényi divergence [128] and is defined as follows (Mironov [127]).

Definition 2. *A randomized algorithm F is said to be (α, ε)-RDP with order $\alpha \geq 1$ and $\varepsilon \geq 0$ if for any neighboring datasets D_0, D_1, it has $\mathrm{Div}_\alpha(F(D_0)||F(D_1)) \leq \varepsilon$ where*

$$\mathrm{Div}_\alpha(F(D_0)||F(D_1)) := \frac{1}{\alpha - 1} \cdot \log \mathop{\mathbb{E}}_{Y \leftarrow F(D_0)} \left[\left(\frac{\Pr(F(D_0) = Y)}{\Pr(F(D_1) = Y)} \right)^{\alpha - 1} \right]$$

RDP has several properties that make it useful in the context of machine learning such as its relationship with the Gaussian mechanism [129], [130] and its composition properties. We refer the reader to Mironov [127] for comprehensive coverage. However, an important property we highlight here is that we can convert an RDP guarantee to a (classical) DP guarantee.

Lemma 1. *If a randomized algorithm F is said to be (α, ε)-RDP, then it also satisfy $(\varepsilon + \log(1/\delta)/(\alpha - 1), \delta)$-DP for all $\delta \in (0, 1)$.*

2.5. Differentially Private Machine Learning

There are numerous randomized learning algorithms that meet differential privacy such as output perturbation [10] or objective perturbation [131]. However, the most widely used technique is the Differentially Private Stochastic Gradient Descent (DP-SGD) of Abadi et al. [13].

Algorithm 1 SGD

Input: Training dataset D, loss function $\mathcal{L}(\theta)$. Parameters: learning rate η_t, mini-batch size L. $N = |D|$ is the number of training data points.
Initialize θ_0 randomly
for $t \in [T]$ **do**
Take a random mini-batch L_t with probability L/N.
`Compute gradient:`
$\mathbf{g}_t \leftarrow \nabla_\theta \mathcal{L}(\theta_t, L_t)$
`Descent step:`
$\theta_{t+1} \leftarrow \theta_t - \eta_t \mathbf{g}_t$
end for
Output: θ_T

Algorithm 2 DP-SGD (Abadi et al. [13])

Input: Training data $x_1, ..., x_N$, loss function $\mathcal{L}(\theta) = \frac{1}{N}\sum_i \mathcal{L}(\theta, x_i)$. Parameters: learning rate η_t, noise scale σ, mini-batch/lot size L, gradient norm bound C.
Initialize θ_0 randomly
for $t \in [T]$ **do**
Take a random sample L_t with probability L/N
`Compute gradient:`
For each $i \in L_t$, compute $\mathbf{g}_t(x_i) \leftarrow \nabla_\theta \mathcal{L}(\theta_t, x_i)$
`Clip gradient:`
$\bar{\mathbf{g}}_t(x_i) \leftarrow \mathbf{g}_t(x_i)/\max(1, \frac{||\mathbf{g_t(x_i)}||_2}{C})$
`Add noise:`
$\tilde{\mathbf{g}}_t \leftarrow \frac{1}{L}\sum_i (\bar{\mathbf{g}}_t(x_i) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I}))$
`Descent step:`
$\theta_{t+1} \leftarrow \theta_t - \eta_t \tilde{\mathbf{g}}_t$
end for
Output: θ_T and and compute the overall privacy cost (ε, δ) using a privacy accounting method

DP-SGD. In essence, DP-SGD computes gradients and applies gradient updates in a similar fashion as SGD, but uses the Gaussian mechanism to add noise to the aggregate mini-batch gradient. However, since the sensitivity of the mini-batch gradient to a single example is unbounded, the per-example gradients are clipped prior to aggregation to ensure that their l_2-norm is smaller than a clipping threshold $C > 0$. This guarantees that the sensitivity of the mini-batch gradient sum is at most C. In other words, the noisy gradient is calculated as:

$$\bar{g} = \frac{1}{L}\sum_i \text{clip}_C(g_i) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I}) \ ,$$

where L is the number of examples in the mini-batch, g_i is the gradient vector of example i, and σ is the noise level. For completeness, we provide a complete description of the DP-SGD algorithm in Algorithm 2. Since it is an iterative algorithm where gradients are computed multiple times over the same data, careful privacy budget account and composition is required [13]. A notable drawback of per-example gradient clipping as used in DP-SGD is processing speed. The running time for one epoch with DP-SGD is between 10 to 30 times slower than with SGD [25], [26].

Privacy-utility trade-off. There is a natural tradeoff between privacy and utility when training the model with DP-SGD. This is because the smaller the privacy budget ε the more noise needs to be added to the clipped gradients and thus the more distortion the training process will experience. In this paper, we think of privacy as the total privacy budget ε and of utility as a combination of test accuracy of the trained model and also training time and memory usage.

3. Architectures, Features, Hyperparameters and Augmentations

Best practices for machine learning in the non-private setting do not typically yield the best results for DPML [13], [132], [34], [19], [25], [35], [29], [30], [31]. This is why significant research has gone into tuning model architectures, hyperparameter optimization and feature selection for DPML.

3.1. Tuning Model Architectures

Several researchers have pointed out that model architectures that perform well in the non-DP setting may have poor performance in the DP setting [132], [133], [134], [25]. Therefore, carefully selecting model architectures is necessary to achieve the best results.

Architecture search.
Priyanshu et al. [29] evaluate Evolutionary, Bayesian, and Reinforcement learning search algorithms, finding them superior to grid search. Bao et al. [25] use a genetic algorithm for model architecture search, mutating model elements as genes and evaluating with the Laplace mechanism for privacy. Cheng et al. [30] focus on a narrower search space, introducing a changeable NAS Cell in a CNN model. The authors optimize this cell using Reinforcement Learning. Remerscheid et al. [31] empirically select model components to design SmoothNet, which achieves better DP performance than other well-known models.

Activation function. Papernot et al. [34] observe that DP-SGD causes model activations to explode, leading to information loss during gradient clipping. The authors propose using a bounded activation function, a tempered sigmoid, to replace ReLU. Experiments show improved privacy-utility trade-offs with this activation function.

Loss function. Wang et al. [32] propose smoothing the loss function using randomized smoothing, improving performance in image and language tasks. Shamsabadi et al.[33] present a DP-SGD tailored loss function combining per-example sum squared error, a focal loss variant, and a regularization penalty. This approach is also thought to smooth out the loss surface, enhancing DP noise tolerance.

Tuning existing architectures. Instead of searching for architectures from scratch, one idea is to take a well-established architecture and make it DP-friendly. For example, Residual networks (ResNet [135]) are very popular in computer vision tasks and won several victories in image challenges such as ImageNet (ILSVR) challenge.

Klause et al. [14] proposed an architecture modification on ResNet, adding a normalization layer after the residual block called ScaleNorm. The authors claim that ScaleNorm can improve the speed of convergence, and achieve SOTA performance on CIFAR-10 when trained from scratch. Sander et al. [15] propose changing the order of layers in a ResNet block. They show experimentally that changing the order of activation and normalization layers has a significant impact on performance. Specifically, that using normalization before ReLU leads to improved performance compared to using ReLU before batch normalization.

3.2. Feature Selection

Reducing data dimensionality can mitigate the curse of dimensionality and enhance performance. DP-PCA was introduced by Dwork et al. [136]. Abadi et al. [13] in the original DP-SGD paper, demonstrates that PCA can boost performance by reducing model parameters.

Aside from PCA, Tramèr and Boneh [19] propose to use handcrafted features and show that a model trained on these features can outperform models trained from scratch without such features. Specifically, they use a Scattering Network as a feature extractor with Group Normalization. They combine it with the linear model and deep models in experiments and show that all of them outperform models that do not use feature selection. We evaluate their methods in our reproducibility experiments (Section 8).

Bao et al. [25] describe three feature selection methods depending on the available privacy budget, pointing out that using all features does not always yield the best model. Singhal et al.[35] propose a dimensionality reduction method that projects data into low-dimension linear subspace.

3.3. Hyperparameter Tuning

Researchers have pointed out the importance of tuning hyperparameters (e.g., learning rate, regularization constant, batch size, etc.) in the DP setting [25], [134], [17], [15].

Kurakin et al. [134] find that hyperparameters (i.e, suitable learning rate, larger batch size, or epochs) have a significant impact on DP performance.

Dormann et al. [16] investigate that the inherent sampling noise in SGD and Gaussian noise in DP-SGD is equivalent to achieving privacy. They propose a novel method for tuning hyperparameters that uses a larger batch size and high noise multiplier to achieve SoTA performance. We include their method in our reproducibility experiments (Section 8).

Costs of hyperparameter tuning. Although hyperparameter tuning is crucial, it can be computationally expensive, and ideally, its privacy effect should be accounted for.

Sander et al. [15] proposed a method for reducing the computational cost of hyperparameter tuning with DP-SGD. The method involves using the Total Amount of Noise (TAN) and a scaling law. TAN is defined as the sum of the variances of the noise added to each gradient update during training and represented as follows:

$$\eta^2 = \frac{1}{\Sigma^2} = \frac{q^2 S}{2\sigma^2} \ ,$$

where η is the signal-to-noise ratio, Σ is the TAN, q is simple rate, S is the step size and σ is noise level.

The authors argue that ε is directly related to TAN and can be minimized, which means that TAN can be used to predict privacy guarantees. Furthermore, the scaling law can be leveraged to estimate the performance of large batch sizes using smaller batch sizes. In experiments, Sander et al. [15] show that their method can find optimal hyperparameters, leading to new SoTA performance on ImageNet.

Ideally, the privacy cost of hyperparameter tuning should also be considered. Papernot and Steinke [36] highlight that hyperparameter settings can leak private information and offer theoretical tools to ensure privacy during tuning. Wang et al. [37] introduce DP-HyPO, an adaptive hyperparameter optimization framework that combines adaptive search with DP and can refine sampling distributions adaptively based on previous runs while maintaining strict DP guarantees.

3.4. Data Augmentation

De et al. [17] advocate for the use of large batches and replace batch normalization with group normalization, and weight standardization. They also propose to create several *self-augmentations* of each example and then average their gradients before clipping. This does *not* affect the privacy analysis because it happens prior to clipping (thus any given example only affects one clipped gradient value per mini-batch). This is particularly effective and results in a new SoTA on CIFAR-10 when trained from scratch.

Building on this, Bao et al. [18] introduced DP-MIX$_{\text{SELF}}$ a technique that employs Mixup [137] for self-augmentation of individual training samples, resulting in state-of-the-art (SoTA) performance in both training models from scratch and fine-tuning pre-existing models. A second method, called DP-MIX$_{\text{DIFF}}$, aims to further enhance performance by using a text-to-image diffusion model to create class-specific synthetic examples. These synthetic samples are then mixed up with actual training data, achieving new SoTA performance levels without incurring additional privacy costs. These two techniques have established new SoTA benchmarks across various image datasets.

Park et al. [38] propose synthesizing a large, diverse dataset from a small set of in-distribution public data using diffusion models to improve warm-up training and boost the final utility of privately trained models.

4. All About Clipping

Gradient clipping is an essential part of DP-SGD and also the focus of numerous techniques claimed to boost DP model performance. These can be categorized into five classes and we summarize them in Table 3. In this section, we provide a brief overview of existing clipping techniques.

TABLE 3: Different types of gradient clipping techniques

Type of clipping techniques	Reference
Improvements on basic clipping	[39], [40], [41],[20]
Adapt clipping	[42], [43], [44], [45], [46], [47], [54]
Auto clipping	[21] , [48]
Layer(Group) clipping	[13], [49], [50], [51]
Global clipping	[53]

4.1. Basic Clipping & Limitations

The original gradient clipping process proposed by Abadi et al. [13] is defined in Eq. (1). We call it *basic clipping*.

$$\bar{\mathbf{g}}_t \leftarrow \mathbf{g}_t \cdot \min(1, \frac{C}{||\mathbf{g_t}||_2}) \ , \tag{1}$$

where $\mathbf{g}_t$ is the original gradient vector at iteration t, $\bar{\mathbf{g}}_t$ is the clipped gradient vector, and C is the clipping threshold.

In a nutshell, clipping preserves the gradient vector whenever its l_2-norm is bounded by the clipping threshold C. When it is not, the gradient vector is first normalized and then re-scaled by a factor of C, preserving the direction but reducing magnitude.

Instead of considering C as an independent parameter, De et al. [17] merge the clipping factor into the learning rate and tune the two parameters together to simplify tuning without altering the privacy guarantee.

Limitations of basic clipping. Basic clipping in DP-SGD can distort gradient vectors, impacting convergence. Chen et al. [39] found that clipping bias affects convergence and propose adding Gaussian noise before clipping to counteract this. Koloskova et al. [55] detail how different clipping thresholds affect convergence and aim for improved convergence guarantees through adaptable thresholds. Meanwhile, Xiao et al. [56] study optimal Gaussian noise with hybrid clipping, introducing a "twice sampling" method for better utility-privacy trade-offs, and exploring optimal noise for RDP in high dimensions.

Beyond basic clipping. A consequence of basic clipping is that per-example gradient clipping slows down training and can take up significant memory. Lee et al. [40] propose a modification on basic clipping that uses the "auto-differentiation" library in deep learning frameworks to compute the norms of per-example gradients. This method can be implemented on different types of neural networks and show significant speed-ups.

Bu et al. [41] propose a fast gradient clipping that approximates per-sample gradient norms instead of computing them by using Johnson–Lindenstrauss projections which could save significant $30\times$ time and use a similar memory footprint as non-private optimization.

4.2. Adaptive Clipping

Instead of using a fixed clipping threshold like basic clipping, Adapt clipping [42], [43], [44], [45], [46], [47] seeks to use different clipping thresholds for different iterations to improve model's performance.

Yu et al. [42] note gradients tend to zero in convergent algorithms, so they linearly decrease the clipping threshold to half its original value. Pichapati et al. [43] introduce AdaClip, a coordinate-wise method that requires less DP noise for a given privacy budget. Andrew et al. [44] adapt the clipping threshold using gradient quantile approximations. Du et al. [45] propose a dynamic threshold that decreases at every iteration, adjusting noise scaling accordingly. Lin et al. [46] suggest a method to find the optimal threshold on public data with a greedy algorithm, and a decay gradient clipping threshold using a decay function. Golatkar et al. [47] employ public data to gauge the 90th percentile of gradient norms, using this insight for adaptive clipping on private data. Xia et al. [54] introduce DP-PSAC, which uses an adaptive weight function instead of a constant clipping norm, narrowing the deviation between true and batch-averaged gradients.

4.3. Efficient Clipping

Auto clipping. No matter the fixed clipping threshold or adaptive clipping threshold, there are always one or more hyper-parameters that need to tune. Bu et al. [21] and Yang et al. [48] proposed a new clipping technique called Auto clipping which can be defined as follows:

$$\bar{\mathbf{g}}_t \leftarrow \mathbf{g}_t \cdot \frac{1}{||\mathbf{g_t}||_2 + \gamma} \ .$$

This method removes the clipping threshold instead of using methods to find it. The idea is to incorporate a parameter γ which can be relatively stable for different sets of hyperparameters, to preserve the gradients' magnitude information. The authors show that the convergence of DP-SGD with Auto clipping is the same as the standard SGD. In experiments, DP-SGD with Auto clipping achieves slightly better performance than DP-SGD with basic clipping.

Mixed ghost clipping. By combining and extending previous works [138], [40], [62], Bu et al. [20] propose a technique called mixed ghost gradient clipping which does not require computing per-sample gradients. Their method yields computation time and memory close to non-private optimization. They claim to achieve a new SOTA performance on CIFAR-10 and CIFAR-100.

4.4. Group Clipping

This subsection surveys group-based gradient-clipping strategies—layer-wise, global, and batch-level—that refine standard per-sample clipping by assigning tailored thresholds to structural or statistical subsets of the gradients, thereby reducing utility loss while preserving differential-privacy guarantees.

Layer clipping. Layer clipping [13], [49], [50] sets unique clipping thresholds for different model layers, recognizing that each layer has distinct gradient norms. In contrast,

Liu et al. [51] introduce a method dividing gradients into k groups, each with its own clipping threshold, showing reduced gradient loss compared to basic clipping. Xu et al. [52] present DPSGD-F, a group clipping technique with an adaptive threshold, adjusting each group's sample contribution to equalize utility loss across groups. Their experiments demonstrate improved DP performance.

Global clipping. Bu et al. [53] present global clipping techniques that assign either 0 or 1 as the clipping factor to all gradients. This method keeps small gradients and removes large ones, typically from noisy samples, aiming to maintain gradient direction and minimize bias.

Batch clipping. Xiao et al. [57] analyze gradient clipping theoretically, suggesting its bias is often underestimated due to the gradient's sampling noise. Based on this, they propose techniques, including inner-outer momentum and Batch Clipping, using public data to inform batch sample clipping. They also offer other strategies to boost DPML performance.

5. Transfer Learning & Fine-tuning

A model pre-trained on a public dataset can be used as a starting point for DP training. A typical methodology is to reuse the weights of the pre-trained model and then fine-tune a new model on the sensitive dataset with DP-SGD.

The benefit comes from the use of public datasets for which the privacy cost is *not* taken into account. The assumption is that the pre-training data is public or otherwise not sensitive. This assumption becomes increasingly dubious the more similar the pre-training and fine-tuning datasets are. At the same time, the more similar the pre-training and fine-tuning datasets are the better the expected performance.

5.1. Transfer Learning, Domains, and Tasks

It is unclear whether pre-training the model on public data is still beneficial when there is a significant domain gap between private data and public data for image tasks. However, many papers [13], [60], [47] that focus on image tasks tend to assume that public data and private data have similar distributions.

By contrast, for language tasks, a large model pre-trained with generic public data appears to still yield good performance when fine-tuned with out-distribution private data [62], [63], [139], [61], [140].

However, using public data in DPML also brings some concerns. Tramèr et al. [64] discuss improving DPML through transfer learning from extensive non-private datasets. They express concerns over differential privacy in web-acquired data and the challenges of outsourced data in large pre-trained models, urging a closer look at privacy implications in public pre-training.

5.2. Parameter-Efficient Fine-Tuning Techniques

Mehta et al. [58] find that increased pre-training data and bigger models enhance performance. They also observe improvements using large batch sizes, LAMB optimization [141], and initializing the final layer's weights to zero for fine-tuning with DP.

Luo et al. [23] propose a new fine-tuning technique. Their techniques fine-tune the normalization layer and percentage of parameters in convolution layers that have high magnitude. This results in SoTA performance.

Cattan et al. [22] propose fine-tuning the first and last layers to improve performance (an increase of 3.2% accuracy compared to fine-tuning the whole model). They claim to achieve SoTA performance.

Bu et al. [59] introduce DP-BiTFiT which fine-tunes only the model's bias using DP-SGD. With only 0.1% parameter changes, this technique achieves leading results. The authors propose a two-phase training for image tasks, first full-model fine-tuning and then DP-BiTFiT. Li et al. [62] describe ghost clipping for fine-tuning, noting enhanced results with larger models and optimal hyperparameters. Yu et al. [63] present a meta-framework, suggesting techniques like low-rank adaptation can boost fine-tuning in large models. Lowy et al. [65] explore optimizing DP learning with public data, highlighting its benefits in reducing sample complexity.

For language tasks, Hoory et al. [61] introduce the DP Word Piece algorithm for language tasks, adding noise to word histograms and leveraging parallel training on TPUs. Yu et al. [66] champion "selective pre-training" on public data, later finetuning on private datasets, benefiting both privacy and model efficiency. Mireshghallah et al. [67] focus on compressing models like BERT to 50% sparsity using DPKD and DPIMP, achieving competitive performance on GLUE.

5.3. DP-Aware Pre-Training and Representation Learning

This subsection focuses on works that first shape the feature extractor (synthetic images, captioning, robustness objectives) so that subsequent DP fine-tuning is both easier and more accurate.

Tang et al. [24] introduce a method to pre-train models on random process data [142], [143], eliminating reliance on existing public datasets. Their three-phase process starts with pre-training the model's feature extractor on this data. Next, they train the classification layer solely on private data, freezing the feature extractor. In the last phase, the entire model is fine-tuned on private data. Their approach achieves top-tier results, underlining its effectiveness.

Similar to Tang et al. [24], Yu et al. [144] propose a method to train foundation vision models with differential privacy using masked autoencoders and DP-SGD. They pretrain MAE on synthetic images which is random process data as [24] and adapt MAEs for DP by utilizing an instance-separable loss function that aligns well with the per-sample gradient computations required for DP-SGD. This method enables the effective training of privacy-preserving models on large-scale, uncurated datasets, maintaining competitive performance on standard vision tasks. Although sharing a

TABLE 4: Different types of gradient size reduction techniques

Type of gradient size reduction	Reference
Random projection	[69], [70], [71], [25]
Low-rank matrix approximation	[72], [73], [72]
Private selection gradient	[74]
Subspace projection	[75], [76], [80]
Public data utilization	[77], [78], [79]

similar idea with Tang et al. [24], Yu et al. [144] focuses on a different training situation: private pre-training.

Sander et al. [145] introduces a representation learning method using image captioning, which enhances model performance by utilizing text captions for better supervision and information extraction under privacy constraints. The authors develop a DP-Cap model that significantly outperforms previous models in learning high-quality image features from a large-scale dataset (LAION-2B).

Wang et al. [68] propose DPAdapter, a pre-training technique that enhances a model's parameter robustness against DP noise by improving sharpness-aware minimization (SAM) [146] with a two-batch strategy for more accurate perturbation estimation, making the model more resilient during downstream DP fine-tuning.

Besides getting a better starting point, public data with a similar distribution as private data can also give some insight during fine-tuning with DP-SGD. Golatkar et al. [47] use an adaptive method that processes gradients in a low-dimensional space, enhancing DP fine-tuning. Amid et al. [60] generate loss based on public data and compare it to the loss on private data. They let the fine-tuned model take smaller gradient steps with less noise when the public loss grows quickly.

6. Other Techniques

This section surveys other techniques to improve privacy-utility from different perspectives.

6.1. Sparse Training

Some research propose to mitigate the impact of clipping and noise on the gradient by reduce the effective size of gradient vectors. We categorize these methods into five classes summarized in Table 4.

Random projection. Random projection of the gradient has been proposed in traditional (non-DP) model training [147], [148]. Zhu and Blaschko [69] introduce a DP training version that employs a random freeze mask, updating only a subset of gradients each iteration. This reduces noise addition, leading to enhanced DP performance, with added memory and communication efficiency benefits. Kasiviswanathan [70] takes a similar approach but uses a subgaussian or sparse Johnson–Lindenstrauss matrix for random gradient projection instead of a freeze mask.

Sidahmed et al. [71] introduce "partial training," freezing parts of the neural network with the most parameters. Based on performance, they adjust which parts are frozen. Their method shows better results with smaller privacy budgets than full-network training. Similarly, Bao et al. [25] suggest "random weights training," initializing and freezing the initial layers' weights.

Low-rank matrix approximation. Using low-rank matrices to represent a high-rank gradient matrix is a popular gradient size-decreasing method [149], [150]. To implement it in DP training, Yu et al. [72] use two low-rank matrices to represent the gradient. By only adding noise to low-rank matrices, they can get better DP performance. Finally, they will reconstruct the gradient using noise low-rank matrices. Based on that work, Ito et al. [73] also generate two low-rank matrices but before adding DP noise, they will set the unimportant gradient of low-rank matrices to 0. So compared to Yu et al. [72], they further decrease gradient size. These works show improved DP performance when the privacy budget is limited.

Private selection gradient. Current neural networks are often over-parameterized and there exists some sparse in the model that can be compressed. Based on this idea, it is natural to consider only updating the important subset of the gradient. To make the selection process differential private, private selection techniques need to be implemented. Zhang et al. [74] firstly clip the gradient and then use a private selection technique to generate a gradient mask that only keeps important gradients. Then, they do a second gradient clipping and add DP noise to it.

Subspace projection. Subspace projection methods focus on projecting gradients to subspace which can still preserve its important information. Asi et al. [75] introduce a private version of AdaGrad with non-isotropic clipping and noise, projecting gradients into an ellipsoid space. Nasr et al. [76] encode gradients into a smaller vector space using different noise distributions for improved utility and employ a denoising mechanism to scale noisy gradients, achieving comparable results to DP-SGD on MNIST. Feng et al. [80] present Spectral-DP, projecting DP noise in the spectral domain with spectral filtering, aiming to reduce noise magnitude.

Public data. Using public data information can assist in gradient reduction. Zhou et al. [77] compute the top-k eigenspace from public data for gradient projection, updating weights with projected noisy gradients. Yu et al. [78] employ unlabeled public data to form an anchor subspace, then project and perturb gradients within this subspace for differential privacy.

The choice of appropriate public data is crucial. Gu et al. [79] present an algorithm to select suitable public datasets for private ML tasks. By leveraging the Gradient Subspace Distance (GSD) to measure dataset differences, the method aims to reduce noise. GSD's effectiveness is evaluated across various models, and found to be enhancing the private training of large-scale models.

6.2. Better Privacy Accounting

There is considerable literature on proposing tighter privacy bounds through composition. This allows the learning algorithm to perform additional iterations while using the same privacy budget and noise level. Asoodeh et al. [81] derive a tighter privacy bound based on a joint range of two f-divergences. f-divergence [82] is a function to measure the difference between two probability distributions. By using this method, the authors optimally convert from RDP to DP which can be applied to the moments accountant, allowing for 100 additional training iterations for the same privacy parameters.

A series of recent studies use an accounting method based on the Fast Fourier Transform (FFT) method [151]. The FFT-based account method was first proposed by Koskela et al. [83]. The proposed accountant method is based on an integral formula numerical approximation which is realized by discretizing the integral and assessing discrete convolutions with the FFT algorithm. They also show in the experiments that their method has better performance than moments accountant and other accountant methods in terms of bound tightness and running time. Koskela et al. [84] propose an accounting method that is based on the privacy loss distribution formalism by using the FFT-based account method. They conduct an error analysis to obtain a tighter bound and they show that their approach can decrease 75% noise variance. Koskela et al. [85] extend the FFT-based accounting method to heterogeneous composition to compute tight privacy bound and conduct an error analysis to produce a better complexity bound. Their experiments show that their method can allow $1.5\times$ more compositions than the original moments accountant. Gopi et al. [86] propose a new algorithm based on previous work [84] that can compute the privacy composition curve more efficiently. Using the coupling approximation, their proposed algorithm shifts the discretized random variables and enhances the approximation procedure. They also improve the truncation approach by offering a tighter tail bound of the privacy loss random variables. Their new algorithm is fast at least k times than previous work [84] (k is the number of compositions).

A different way to achieve tighter privacy bounds is to reconsider traditional assumptions of differential privacy machine learning. In the original DP-SGD setting, all iterations' gradients are implicitly assumed to be visible to the adversary. Thus privacy accounting protects the entire training process's internal state, and not just the final model or iteration.

However, this assumption may not be realistic in practice as model owners may train their models on isolated systems and only publish the model after the end of the training process. To capture this, several works propose the idea of *hidden-state analysis* where only the last iteration of training is observed by the adversary.

Feldman et al. [152] demonstrate privacy guarantees can be amplified by not releasing the intermediate results. They also show that this new analysis can be applied to various applications such as distributed SGD, Multi-query settings, and Public/private data settings.

Chourasia et al. [87] propose a framework to analyze the dynamics privacy loss of noisy gradient descent algorithms based on a pair of continuous-time Langevin diffusion processes. They provide a tighter bound on the Rényi divergence than composition bounds under the assumption that only the last iteration information is released and the loss function is smooth and strongly convex.

Extending previous work from gradient descent to SGD, Ye and Shokri [88] provide a tighter bound relying on the post-processing phenomenon for noisy mini-batch SGD. Their coverage bound can be implemented on convex loss function and is suitable for different batch sampling methods so it can also be implemented with DP-SGD.

6.3. DP Auditing

Jagielski et al. [91] introduce a novel data poisoning attack to evaluate DP-SGD in real-world settings, comparing practical privacy to the theoretical estimates. Nasr et al. [92] propose an approach to assess the actual privacy levels of DP-SGD using a conceptual adversary. They aim to set lower bounds on the likelihood of succeeding in a distinguishing game, a metric for data privacy. Their results indicate that in real-world scenarios, the actual privacy provided may exceed theoretical guarantees.

Lu et al. [93] present ML-Audit, a framework designed to statistically audit the privacy assurances of DPML. Nasr et al. [89] introduce an enhanced auditing scheme for assessing the privacy of machine learning algorithms under differential privacy. Their approach requires just two runs and offers precise privacy estimates for non-adversarially curated datasets. Steinke et al. [90] reduce the computational overhead linked to privacy auditing. They show how to estimate a lower bound of ε in a single run.

Hu et al. [153] introduce the concept of "empirical privacy variance," showing that models with the same theoretical DP guarantee can have significantly different practical privacy leakage based on hyperparameter choices.

There is also a line of research that focuses on auditing the privacy of Large Language Models (LLMs) [154], [155], [156], [157], [158], [159], [160], [161]. For example, Panda et al. [154] insert rare "canary" tokens to amplify memorization, enabling simple black-box membership-inference audits that reveal leakage even in DP-trained LLMs. These tests are lightweight and model-agnostic, and thus provide a way for practitioners to catch privacy bugs and estimate real-world risk.

6.4. Learning Algorithm Modifications

Researchers have proposed variants of DP-SGD to overcome some of its limitations such as slow convergence and long running time and (or) to achieve superior privacy and utility trade-offs. We summarize these variants in Table 5.

TABLE 5: Variants of DP-SGD

Type	Reference
Adaptive optimization	[94]
Adaptive noising	[95],[96],[16], [102]
Improved accounting	[162],[98]
Variants of DP-SGD	[99], [106], [105]
DP-SGD acceleration	[26],[100]
Public data utilization	[101], [103]

Adaptive optimization. Shortening the number of training iterations directly reduces the overall privacy budget consumed. However, the rate of learning has to increase to compensate for having fewer learning iterations. Adaptive optimization techniques such as AdaGrad [163], Adam [164], and RMSProp [165] change the learning rate adaptively based on second-moment gradient information. These techniques have better performance than SGD in the non-DP setting. Therefore, when switching to the DP setting, it is natural to consider replacing DP-SGD with differential privacy adaptive optimization. Zhou et al. [94] propose DP Adam and DP RMSProp and prove their DP guarantee. They also represent their convergence rate of them and evaluate their performance. They use experiments to show that DP adaptive optimization outperforms DP-SGD.

Li et al. [104] introduce a method for private adaptive optimization without extra public data. Using delayed preconditioners, they enhance adaptivity and reduce noise. They provide convergence proofs for both convex and non-convex cases. Tests on real-world datasets show improved convergence rates and model utility within a set privacy constraint.

Adaptive noising. A different idea is to optimize the additive noise added to achieve privacy to reduce its deleterious impact on the training process.

Lee et al. [95] argue that it is necessary to allocate privacy budget carefully for different iterations. They propose an algorithm (DP-AGD) that adaptively allocates privacy budget for different iterations with zero-mean Concentrated Differential Privacy (CDP) [166].

Xu et al. [96] propose a new algorithm called ADADP which implements adaptive learning rate and adaptive noise together to achieve better convergence rate and model performance. To apply adaptive noise, they add noise with smaller variance to the gradients with smaller sensitivity to reduce the effect of noise on the gradient's true direction.

Xiang et al. [102] consider the utility subspace of the model. The authors find that if the DP noise is added to some less important directions, they can achieve improved performance. They proposed a directional noise DP mechanism that selects a noise direction with optimal utility.

Zhang et al. [107] introduce DiSK, a framework that enhances DP optimizers by using a simplified Kalman filter to denoise privatized gradients, treating them as noisy observations of a dynamic system to achieve more accurate gradient estimates.

Improved accounting. The original moments accountant proposed by Abadi et al. [13] for DP-SGD has tighter composition bound than advanced composition [11]. However, when considering variants of the DP-SGD algorithm, privacy accounting can be further enhanced.

Wang et al. [97] propose a new algorithm called Differentially Private Stochastic Recursive Momentum (DP-SRM) based on a sharp analysis of the privacy guarantee using RDP. They build this algorithm based on the stochastic recursive momentum technique that is able to decrease the variance of gradients. Their algorithm is more scalable and efficient than the original DP-SGD.

Ding et al. [98] propose Perturbed Iterative Gradient Descent Optimization (PIGDO) which uses gradient descent optimization (GDO) as part of iteration update among some batches, aggregates these batches into a lot and then clips the mean gradients and add noise. By studying the privacy loss of PIGDO, they also propose a new Modified Moments Accountant (MMA) with a tighter bound to save privacy budgets. Their experiments show that PIGDO with MMA outperforms original DP-SGD in terms of utility and privacy.

Variants of DP-SGD. In each iteration DP-SGD first computes the gradient, clips the gradient, and then adds noise to it. Xiao et al. [99] propose a technique called ModelMix. The iterative perturbation in ModelMix has the same gradient computation and clipping as DP-SGD, however, before adding Gaussian noise to the clipped gradients, ModelMix performs a random aggregation of the model's weights among the last and penultimate iterations' weights. The authors claim that this random aggregation is equal to adding a randomized proximal term into the objective function and decreasing the noise that is added in the last few iterations.

Xiao et al. [106] introduce Data-Specific Indistinguishability (DSI), a variant of differential privacy that provides trust guarantees by adding optimal, anisotropic Gaussian noise to ensure a model's output is statistically close to outputs from a specific set of safe reference datasets.

Wei et al. [105] present DPIS as an alternative optimizer for DP-SGD. By leveraging importance sampling for mini-batch selection in each SGD iteration, DPIS reduces sampling variance and the necessary random noise for maintaining privacy.

Speeding up DP-SGD. Subramani et al. [26] propose a new training method for DP-SGD that combines vectorization, just-in-time compilation, and static graph optimization to reduce running time. They implement their method on JAX and find that it runs $50\times$ faster than the best alternative in some cases.

The most time-consuming and high computation cost part of DP-SGD is per-sample gradient clipping which needs to compute per-example gradients to obtain per-example gradient norm. To overcome this major bottleneck, Ghost clipping [62], [20] is used to compute per-example gradients' norm without actually computing per-example gradients. Essentially, per-example gradient norms can be computed by computing per-example gradient norms for each layer of the model so one only needs to instantiate one per-example gradient tensor. However, this requires two

back-propagation passes. To improve from Ghost clipping, Bu et al. [100] propose a book-keeping technique that could make DP training faster and memory efficient almost as non-DP training. This only requires one back-propagation pass to computes per-example gradient norm like Ghost clipping.

Public data. Some algorithms assume the existence of public data that can be used by the learning algorithm. Since the public data is by assumption non-sensitive there are no privacy concerns when accessing it. We discuss this assumption further in Section 5.

Kairouz et al. [101] introduce a differentially private variant of AdaGrad that adds Gaussian noise to both gradients and the preconditioner while projecting updates onto a gradient subspace estimated from public data. They show that, under low-rank subspace structure and decaying gradient norms, this approach achieves faster convergence and dimension-independent excess risk than standard DP-SGD.

Li et al. [103] propose AdaDPS that uses public data or common knowledge about the private data to estimate gradient statistics to precondition the gradients. Experiments show that their method can improve accuracy on language and image tasks.

6.5. Teacher-Student Approaches

A different setting using public data is the semi-supervised knowledge transfer setting of Papernot et al. [108]. In this setting, an unlabeled public dataset is labeled through a differential private process by an ensemble of teacher models trained on disjoint subsets of a sensitive dataset. A student model is trained on the public data augmented with private labels. This technique is called Private Aggregation of Teacher Ensembles (PATE).

Uniyal et al. [109] propose that PATE has better performance than DP-SGD with the same privacy budget. Papernot et al. [110] further improve the technique by proposing a more selective noisy aggregation mechanism. Zhu et al. [111] propose to use KNN for sampling the public data to avoid data splitting in the original PATE setting to improve teacher models' performance. Sun et al. [112] propose a new voting mechanism with smooth sensitivity called Immutable Noisy ArgMax.

7. Selected papers & Methodology

In this section, we describe how we select papers, our evaluation methodology, research questions, and experimental setup. We then propose a framework to establish whether a method provides statistically significant improvements over a baseline. We justify the need for this framework by empirically measuring the consequences of the additional randomness and variability of DPML (compared to non-private ML).

7.1. Selection Criteria

To identify papers for inclusion, we took a deep dive in the DPML literature from 2020 to the time of writing. We also included papers appearing only on arXiv, as some with claimed state-of-the-art methods such as De et al. [17], are hosted there.

TABLE 6: Selected 11 papers for experiments. Here we use Generalizability (G), and Reliability (R) to measure Reproducibility and Reliability. ● means the paper satisfies all requirements in that part. ◐ means satisfying part of them and ○ means satisfying none of them. A more detailed version can be found in Table 20.

Techniques	Paper	Year	Reproducibility G	Reproducibility R
Model architecture	Klause et al. [14]	2022	◐	○
Model architecture	Sander et al. [15]	2023	◐	◐
Hyperparameter selection	Dormann et al. [16]	2021	◐	◐
Augmentation multiplicity	De et al. [17]	2022	●	●
Augmentation multiplicity	Bao et al. [18]	2024	●	●
Feature selection	Tramèr and Boneh [19]	2020	◐	◐
Gradient Clipping	Bu et al. [20]	2022	◐	◐
Gradient Clipping	Bu et al. [21]	2024	●	◐
Fine-tuning technique	Cattan et al. [22]	2022	◐	○
Fine-tuning technique	Luo et al. [23]	2021	●	◐
Fine-tuning technique	Tang et al. [24]	2024	●	●

For a paper to be considered SoTA, it must explicitly claim this status. To ensure a fair and straightforward comparison, we focused on papers employing DP-SGD methods and presenting their main results on computer vision tasks.

From our literature search, we identified 11 papers that are the most representative samples, listed in Table 6. These papers were selected based on the following criteria:

1) They claim to achieve SoTA performance in DPML.
2) Their proposed methods are innovative and straightforward to implement or their code is open source.
3) They represent the latest and most promising research directions, thus providing a comprehensive overview of the current SoTA.

These characteristics make the selected papers particularly noteworthy and suitable for inclusion.

7.2. Methodology & Research questions

We reproduce each of the 11 selected DPML papers using either code open-sourced by the authors or our re-implementation of techniques as described in the paper. We seek to answer the following questions:

(RQ1) Do the proposed methods achieve their claims?
(RQ2) Are improvements obtained outside of the experimental settings used in the papers?
(RQ3) What part(s) of the model should be DP fine-tuned?
(RQ4) Can different techniques be combined?
(RQ5) What techniques are the most promising?
(RQ6) What are important methodological guidelines to ensure scientifically sound and reliable findings?

7.3. Experiments Setup

Datasets. We chose the following 9 datasets: CIFAR-10, CIFAR-100, MNIST, Fashion-MNIST, EuroSAT, ISIC 2018, PathMNIST, Caltech 256, SUN397 and Oxford Pet because they are either used in the 11 papers we evaluated or have a large domain gap to the pretrain dataset. We provide more details for these datasets in Appendix A.

Setup. To ensure a fair comparison, all our experiments are conducted using PyTorch and Opacus [167]. While some works, such as that of De et al. [17], provide open-source code written in JAX, we utilize a reproduced PyTorch version based on the code proposed by Sander et al. [15]. The same codebase was also used by Tang et al. [24] and Bao et al. [18]. Choice of codebase matters as there are notable performance differences, typically in the range of 1-2%, between the JAX and PyTorch versions of [17], as highlighted in some related work [15], [18].

Unless otherwise specified, we adopt the Wide-ResNet 16-4 models as the base model, which is also used in De et al. [17], Klause et al. [14], Tang et al. [24], Bao et al. [18] and Sander et al. [15]. To implement DP-SGD, we use Opacus [167] and make necessary modifications based on different experimental requirements. All experiments (except pre-trained model experiments) are conducted with the same DP setting — $\varepsilon = 8$, $\delta = 10^{-5}$, batch size of 4096, clipping bound $C = 1$, and 200 training epochs. We report the results for 3 independent runs.

7.4. Statistical Framework

We propose a framework to enhance our reproducibility experiments and ascertain if a proposed DPML method provides statistically and practically significant improvements. This framework uses a simplified Cohen's d [168] to measure *effect size* (improvement size) and paired t-tests [169] to measure statistical significance. These methods are well regarded and used across diverse scientific research areas, including psychology [170] and medicine [171].

Statistical tests come with their own sets of drawbacks and their widespread adoption and (mis)use in some disciplines have led to deleterious practices such as p-hacking [172]. Such tests are seldom used in machine learning practice, although there are numerous methods to use them [173], [174]. Machine learning research often reports average performance measures such as accuracy (and sometimes also variation or error bars) and relies on this to establish whether one method is superior to another. We argue that for DPML the use of a framework with statistical testing such as the one we propose is warranted. First, we expect larger uncertainty and variability in each measurement (than for non-private ML). This is in part due to inherently greater randomness in DPML as we demonstrate in Section 7.5. This is especially striking when the privacy budget is small (e.g., $\varepsilon = 0.1$) or the training dataset is small.[1] Second, obtaining each measurement is often an order of magnitude more computationally onerous for DPML, so relying on only a few measurements may be necessary.

Framework Details. Suppose we collect a series of n paired observations from two distinct training runs (e.g., the proposed method vs. the baseline). Let $a_1, a_2, \ldots a_n$ and $b_1, b_2, \ldots b_n$ be the observations for the proposed method and the baseline, respectively. For example, these could be test accuracy measurements on models trained with two different methods but paired such that a_j and b_j are the test accuracies on dataset j (or on run j of n on some fixed dataset). We compute the following:

- Raw means: $\mu_1 = n^{-1} \sum_i a_i$ and $\mu_2 = n^{-1} \sum_i b_i$
- Raw deviations:

$$\sigma_1 = \sqrt{\frac{\sum_i (a_i - \mu_1)^2}{n}} \quad \text{and} \quad \sigma_2 = \sqrt{\frac{\sum_i (b_i - \mu_2)^2}{n}}$$

- Paired differences: $d_i = a_i - b_i$ for $i = 1, 2, \ldots, n$.
- Means and deviation of paired differences: $\mu_d = \sum_i d_i$ and $\sigma_d = \sqrt{\frac{\sum_i (d_i - \mu_d)^2}{n}}$.

To conduct the paired t-test which uses the t-distribution [175], the *t-statistic* is calculated as: $t = \frac{\mu_d}{\sigma_d / \sqrt{n}}$, and from it a *p-value* is obtained [176]. If the p-value is below the set significance level α (e.g., $\alpha = 0.05$) this leads to the rejection of the null hypothesis (in our case the two methods perform the same). If the p-value exceeds this threshold, the null hypothesis stands, suggesting the observed difference between the two methods could be due to chance. In our experiments, we use the default parameters of scipy.stats.ttest_rel to compute the *t-statistic* and p-value.[2]

We also measure practical significance as *improvement size* or *effect size* using Cohen's d. That is, we compute:

$$d = \frac{\mu_1 - \mu_2}{\sigma_p} \quad \text{with} \quad \sigma_p = \sqrt{\frac{\sigma_1^2 + \sigma_2^2}{2}} \,.$$

where μ is the mean of each method's result and σ_p is the pooled standard deviation [28]. Note that the formula for σ_p is simplified from the general case, since in our case the two groups always have the same size.

Test selection and applicability. We chose to base our framework on the paired t-test and Cohen's d because they are well-known, useful even for small n, and straightforward to apply to DPML.

Using a hypothesis testing procedure is more principled than the oft-used rule-of-thumb of looking for an improvement larger than the standard deviation. The paired test is more powerful than its unpaired test because it reduces variability.[3] Further, it allows us to compare methods across multiple experimental settings since test accuracy measurements of models trained by each method in a given experimental setting (same model architecture, same dataset, same number of training epochs, etc.) can be paired.

Test power versus number of runs. The power of the test depends not only on the number of runs, n, but also on the effect size. A statistical rule-of-thumb is that to achieve 80% or more power for paired t-test at significance level $\alpha = 0.05$ requires $n \approx 16 \frac{1}{d^2}$, where d is the effect size (see [177]).

1. When the training dataset is enormous, as is often the case in deep learning research, statistical measurements may be superfluous since any observed difference is automatically statistically significant. But in such cases, obtaining tight privacy guarantees with performance similar to non-private models may not be difficult in the first place.

2. https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.ttest_rel.html

3. The paired t-test's statistical power increases with the correlation.

This means that when the effect size is large, which is the case experimentally for some of the state-of-the-art DPML methods, only a few runs are needed. However, for methods with small effect sizes, the only way to reliably establish an improvement may be to increase the number of runs until the test becomes sufficiently powerful.

7.5. Impact of Randomness

Randomness in the initial weights and mini-batch sampling steps during the training process (e.g., with SGD) plays a significant role in machine learning. However, for DPML, randomness due to the added noise to achieve differential privacy also impedes convergence and decreases the performance of the trained model. Moreover, this additional randomness can obscure the relationship between alternative methods' performance because it tends to increase variability across runs.

We study this phenomenon through a set of principled experiments where we train models with SGD and DP-SGD while varying the random seeds used by the random number generator, thereby allowing us to observe the impact of randomness on model performance. Here it is worth pointing to related work by Picard [178] studying the impact of random seeds on computer vision.

We trained a WRN-16-4 on CIFAR-10 from scratch using 500 randomly chosen seeds. Similarly, we fine-tuned a Vit-base-patch16-224 on CIFAR-10 with 500 randomly chosen seeds. The mean, median, max, min test accuracies, stddev, and the difference between max and min test accuracies from the 500 runs for different ε values and non-private cases. Results are presented in Table 7.

Observe that both the standard deviation and max-min difference is often much greater for DP-SGD than SGD (represented as $\varepsilon = \infty$ in Table 7) e.g. **5.57%** vs 3.61%. In other words, the noise added to the gradient increases the variability in the quality of found solutions. This is in some sense expected given the privacy constraint. However, it suggests that for DPML compared to non-private ML: (1) a larger performance gap between two competing methods is needed to conclude that one outperforms another, and (2) there is greater potential for "*seed hacking*," where unethical researchers specifically select seeds to unfairly claim an advantage for their methods. We discuss this in Section 7.6.

Another relevant observation from Table 7 is that the standard deviation and max-min difference are much lower for fine-tuning compared to the train-from-scratch setting. This suggests that a promising methodological step to ensure reproducibility is to evaluate a method both from scratch and fine-tuning settings. We also included MNIST models trained from scratch in the experiments to establish that variability in the fine-tuning setting is in fact lower and that the observed results are likely not due to this setting typically yielding higher accuracy models.

Finally, we observe higher variability at lower privacy budgets, which is expected but underlines the importance of evaluating methods in a wide range of privacy regimes.

7.6. Seed Hacking?

In this section, we explore the concept of *seed hacking*, inspired by the work of Picard [178]. Seed hacking refers to the process of selectively choosing a small subset of random seeds that anomalously enhance the performance of a proposed method over a baseline, giving a false impression of improvement when, in fact, no genuine enhancement exists from the proposed technique [179], [180]. Note that we are not accusing anyone of engaging in seed hacking; rather, we bring up this possibility as a loose analog to the problem of p-hacking [172] but also to emphasize the significance of the increased randomness in DPML.

We use data from our random seed experiment (Table 7) to simulate a seed-hacking strategy on the fine-tuning task using the CIFAR-10 dataset. For each privacy budget value, we randomly select 10 out of 500 experimental runs, rank these by performance, and choose the top three outcomes for the "proposed method" to simulate what an unethical researcher engaging in seed hacking might do. For the "baseline method", by contrast, we randomly select three outcomes from the same 500 runs.

We conduct two types of statistical tests. The first test employs a common approach where if the mean performance of the proposed method minus its standard deviation exceeds the baseline's mean performance plus its standard deviation, the proposed method is considered superior. The second test applies our proposed framework to conduct paired t-tests with significance denoted by a p-value less than $\alpha = 0.05$. The experiment simulates seed hacking 1000 times and each case executes both tests.

Results are shown in Table 8, where each entry is the number of passing instances of the test. Observe that non-private ML (SGD — $\varepsilon = \infty$) has the lowest numbers. The numbers are greater in the DPML case, especially for low privacy budgets. More importantly, the std test has a much higher false discovery rate than the paired t-test. Note that if seeds were selected randomly instead (no seed hacking) then we would expect an average 50 passing instances in each cell, reflecting a false discovery rate of $\alpha = 0.05$.

The table reports only rates of false discoveries (Type I errors) because the experiment simulates a scenario where a researcher cherry-picks seeds. With all results coming from the same training method, there are no notable improvements (the Null hypothesis is true by definition) and so there are no Type II errors.

Random seeds and differential privacy. There is another subtle but critical difference in the role that random seeds play in DPML versus non-private ML. The reader may wonder why it is unacceptable to optimize the choice of random seed — putting aside for a moment the dishonesty related to seed hacking. After all, if some choices of random seeds are better than others, why not pick the seed that yields the best model? Arguably what matters most is the model that is actually used, not the distribution of possible models that we could have trained. Further, from a pure reproducibility standpoint, fixing the seed to a known value to eliminate its impact is desirable.

TABLE 7: Test accuracy on MNIST and CIFAR-10 for 500 runs with different randomly selected seeds. We use WRN-16-4 for training from scratch on MNIST and CIFAR-10, and Vit_base_patch16_224 for fine-tuning on CIFAR-10. Here ∞ means using SGD (no privacy).

	From scratch								Fine-tuning				
Dataset	CIFAR-10				MNIST				CIFAR-10				
ε	∞	1	4	8	∞	1	4	8	∞	0.5	1	4	8
Mean	77.56%	42.45%	60.20%	66.56%	99.17%	96.80%	98.27%	98.51%	98.32%	96.15%	97.83%	98.00%	98.01%
Std.	0.68%	0.81%	0.93%	0.93%	0.15%	0.20%	0.28%	0.28%	0.02%	0.23%	0.09%	0.07%	0.05%
Median	77.50%	42.53%	60.22%	66.68%	99.21%	96.80%	98.37%	98.63%	98.32%	96.14%	97.83%	98.01%	98.02%
Max	79.64%	45.69%	63.47%	68.61%	99.61%	97.25%	98.67%	98.93%	98.33%	96.61%	98.06%	98.12%	98.14%
Min	76.03%	40.20%	57.90%	64.13%	98.68%	95.60%	96.60%	97.09%	98.17%	95.39%	97.45%	97.79%	97.86%
Max-Min	3.61%	5.49%	5.57%	4.49%	0.93%	1.65%	2.07%	1.84%	0.16%	1.22%	0.61%	0.33%	0.28%

TABLE 8: Number of times that seeds hacking show proposed method better than baseline among 1000 independent runs.

ε	∞	0.5	1	4	8
std test	345	402	397	382	357
t-test	141	191	186	174	153

TABLE 9: Reproduced test accuracy for Dormann et al. [16]

ε	1.93	4.21	7.42
Claimed	58.6% (0.38%)	66.2% (0.38%)	70.1% (0.20%)
Reproduced	58.64% (1.16%)	66.41% (0.78%)	68.88% (1.62%)

This reasoning does *not* run through for DPML because the randomness of the seed is required for the differential privacy guarantee. Since the random seed determines the added noise to the gradient in DP-SGD, fixing it or selecting it on any criteria that an adversary knows about (or can replicate) reduces the uncertainty about the noise distribution, which thus breaks the privacy guarantee. Therefore the seed must truly be selected at random.

8. R+R Experiments

In this section, we conduct our R+R (Reproducibility+Replicability) evaluations on our 11 selected papers. We not only attempt to reproduce their results and check if they match their claims in their papers. We also expand the evaluation settings (e.g., to new datasets, or new model architectures) to ascertain whether the proposed methods still deliver improvements over baselines.

Dormann et al. [16] — Hyperparameter selection. Dormann et al. [16] focuses on hyperparameter selection, advocating for the adoption of large batch sizes (high sampling rate) combined with a higher noise level.

We used their official codebase to reproduce their experiments on CIFAR-10 (Table 9).[4] Our results are consistent with their assertions. Employing larger batch sizes has also been endorsed and adopted by more recent studies such as De et al. [17] and Bu et al. [20] which shows replicability. Consequently, we use this hyperparameter selection strategy as default in subsequent experiments.

4. https://github.com/OsvaldFrisk/dp-not-all-noise-is-equal

TABLE 10: Sander et al. [15] performance for varying ε.

Order	0	1	2	3
$\varepsilon = 8$	71.68% ($\pm$0.50%)	65.74% ($\pm$0.40%)	66.85% ($\pm$1.60%)	**74.07%** ($\pm$0.40%)
$\varepsilon = 1$	52.75% ($\pm$0.22%)	43.10% ($\pm$0.51%)	44.71% ($\pm$0.65%)	**52.96%** ($\pm$0.32%)
$\varepsilon = 0.5$	47.00% ($\pm$0.54%)	40.51% ($\pm$0.23%)	42.54% ($\pm$0.54%)	**47.51%** ($\pm$0.36%)
$\varepsilon = 0.1$	32.19% ($\pm$1.24%)	25.64% ($\pm$1.15%)	29.53% ($\pm$0.85%)	**32.57%** ($\pm$0.92%)

Takeaway 1. *Large batch sizes and high noise levels consistently provide superior performance in experiments across a wide variety of scenarios.*

Sander et al. [15] — Changing order. Recall that Sander et al. [15] proposed changing the order of activation function and normalization layers to obtain a performance boost of 5% to 10%.[5] As a baseline, we train the WRN-16-4 model (on CIFAR-10) using DP-SGD to achieve an average test accuracy of 71.68% (Table 11).

We empirically explored four different ordering schemes as represented in their code.[6] Results are shown in Table 11. The results are consistent with the claims made by Sander et al. [15] as we observed a 5% to 10% boost in performance. Specifically, using order 3 resulted in the best performance, with an average test accuracy of 74.07%.

Although Sander et al. [15] only report results for $\varepsilon = 8$, we perform experiments varying ε from $\varepsilon = 0.1$ to $\varepsilon = 8$ and show results in Table 10. We find that the same pattern holds across ε values, showing that changing layer order as specified in their paper does indeed replicate.

De et al. [17] — Self-augmentation. De et al. [17] achieved a new SoTA performance on CIFAR-10 through hyperparameter tuning and a combination of techniques such as self-augmentation (aka augmentation multiplicity), weight standardization, and parameter averaging (ema). They use a codebase based on JAX, whereas we use a PyTorch version of it from Sander et al. [15].[7]

5. Although Sander et al. [15] proposed other improvements, we are primarily interested in evaluating their changing order method.

6. Order 0, where the order of layers follows Conv-ReLU-GN, with the same order in the shortcut (here Conv means convolution layers, ReLU is the activation function and GN means group normalization layer); Order 1, where the order of layers follows Conv-GN-ReLU, with the same order in the shortcut; Order 2, where the order of layers follows Conv-GN-ReLU, but the shortcut follows Conv-ReLU-GN; and Order 3, where the order of layers follows Conv-ReLU-GN, but the shortcut follows Conv-GN-ReLU. Results applied to the WRN-16-4 model

7. https://github.com/facebookresearch/tan

TABLE 11: Average test accuracy (± standard deviation) for reproducing SoTA methods. The privacy budget for training is 8 and $\delta = 10^{-5}$. We use the same DP setting for all experiments, i.e., batch size is 4096, $C = 1$, 200 training epochs.

Order	0	1	2	3
Baseline (WRN-16-4)	71.68% (±0.50%)	65.74% (±0.40%)	66.85% (±1.60%)	**74.07%** (±0.40%)
Baseline + ScaleNorm	72.71% (±1.20%)	63.87% (±1.40%)	64.18% (±2.90%)	**72.95%** (±0.60%)
Baseline + Mixed Ghost Clipping	**72.96%** (±0.30%)	66.91% (±1.50%)	67.26% (±2.30%)	72.79% (±0.60%)
Baseline + Self-augmentation [17]	77.79% (±0.50%)	68.87% (±0.80%)	69.68% (±0.90%)	**78.10%** (±0.50%)
Baseline + DP-MIX$_{\text{SELF}}$ [18]	78.49% (±0.21%)	69.17% (±0.45%)	69.89% (±0.42%)	**79.83%** (±0.32%)
Baseline + Self-augmentation [17] + ScaleNorm	77.43% (±0.30%)	66.40% (±0.9%)	67.37% (±0.40%)	**78.19%** (±0.20%)

TABLE 12: We fine-tune Clip-Vit-B-16 models on Caltech256, SUN397 and Oxford Pet datasets using different ε with $\delta = 10^{-5}$, and report the test accuracy (%). We can observe that DP-MIX$_{\text{SELF}}$ and DP-MIX$_{\text{DIFF}}$, outperform the baselines in all cases.

Dataset	Method	$\varepsilon = 1$	$\varepsilon = 2$	$\varepsilon = 4$	$\varepsilon = 8$
Caltech256	Self-Aug [17]	80.36(±.11)	89.67(±.16)	92.01(±.08)	93.17(±.15)
	DP-MIX$_{\text{SELF}}$	81.21(±.15)	90.12(±.17)	92.17(±.21)	93.39(±.08)
	DP-MIX$_{\text{DIFF}}$	**89.69**(±.23)	**91.82**(±.15)	**92.86**(±.14)	**93.87**(±.10)
SUN397	Self-Aug [17]	72.65(±.09)	76.02(±.14)	78.05(±.11)	79.54(±.15)
	DP-MIX$_{\text{SELF}}$	73.19(±.13)	76.45(±.17)	78.67(±.16)	79.57(±.14)
	DP-MIX$_{\text{DIFF}}$	**75.12**(±.17)	**77.78**(±.12)	**79.47**(±.18)	**80.57**(±.09)
Oxford Pet	Self-Aug [17]	72.21(±.21)	82.11(±.19)	85.84(±.25)	88.23(±.11)
	DP-MIX$_{\text{SELF}}$	72.45(±.24)	82.51(±.21)	86.75(±.17)	88.70(±.15)
	DP-MIX$_{\text{DIFF}}$	**83.24**(±.26)	**86.28**(±.19)	**88.25**(±.24)	**89.41**(±.21)

For reproducibility, the results we obtain are shown in Table 11. The method significantly improves the model's performance (from 71.68% to 77.79%), which is consistent with De et al.'s claims. Although they reported slightly higher performance, we believe that our results are comparable, considering the randomness in training and the fact that we use a different codebase.

To evaluate replicability, we extend their proposed method by testing it on a wider range of ε from 0.1 to 8 on CIFAR-10 and present the results in Table 13. We observe that their proposed improvements decrease as the privacy budget is decreased. When $\varepsilon = 0.1$, the improvement is marginal. This is in stark contrast to the result obtained varying ε for order switching (Sander et al. [15]).

Bao et al. [18] — DP-Mix. Bao et al. [18] proposed two new techniques DP-MIX$_{\text{SELF}}$ and DP-MIX$_{\text{DIFF}}$ which achieve SoTA results on multiple datasets. We reproduce their results using their official codebase.[8] The reproducible results of DP-MIX$_{\text{SELF}}$ we obtain are shown in Table 11. The method significantly improves the model's performance (from 71.68% to 79.83% for baseline and from 78.10% to 79.83% for De et al. [17]), which is consistent with Bao et al.'s claims. We also reproduce DP-MIX$_{\text{DIFF}}$ using the same settings as the authors presented and show results in Table 12. We observe that DP-MIX$_{\text{SELF}}$ and DP-MIX$_{\text{DIFF}}$ improve the test accuracy in all cases, especially for Caltech256 and Oxford Pet datasets.

We also extended their method to a wider range of privacy budgets (i.e., ε ranging from 0.1 to 8) on CIFAR-10 and show the results in Table 13. We observe that the performance boost from the method decreases significantly as the privacy budget decreases, similar to the method proposed by De et al.[17].

8. https://github.com/wenxuan-Bao/DP-Mix

TABLE 13: De et al. [17] and Bao et al. [18] proposed method performance on CIFAR-10 under different ε. Results show that the impact of the improvements decreases as ε decreases.

ε	0.1	0.5	1	8
Baseline	32.19%(±1.24%)	47.00%(±0.54%)	52.75%(±0.52%)	71.68%(±0.55%)
De et al. [17]	32.42%(±1.03%)	48.98%(±0.42%)	56.06%(±0.45%)	77.79%(±0.50%)
Bao et al. [18]	**32.57%**(±1.23%)	**49.14%**(±0.49%)	**57.24%**(±0.42%)	**78.49%** (±0.21%)

Takeaway 2. *Augmentation multiplicity delivers remarkable performance improvements in practice. It also appears to be a promising direction for future research, albeit its applicability beyond computer vision remains unclear.*

Klause et al. [14] — ScaleNorm. Recall that Klause et al. [14] proposed to add a normalization layer (ScaleNorm) after the residual block to achieve better performance.

We added Scale Normalization layers to the model and evaluated the performance of the modified model on CIFAR-10 (Table 11). We find that this improves the model's performance by about 1%, which is consistent with the results reported in the paper. However, when we applied Sander et al.'s method of changing the order of the activation function and normalization layer, the performance of the model decreased. This result suggests that the benefits of ScaleNorm may not be widely applicable and (or) may not be combined with other techniques.

Bu et al. [20] — Mixed ghost clipping. Recall that Bu et al. [20] propose a gradient clipping method using pre-trained Transformer models. However, without experiments on models trained from scratch or ablation studies, it is not clear whether the observed improvements are due to the clipping technique, the pre-trained models, or both.

We apply Mixed Ghost Clipping to the WRN-16-4 model and train it from scratch using an implementation based on the code provided by the authors.[9] We find that the method only slightly improved performance (Table 11).

We also tested the performance using pre-trained Transformers, following the same setting as the paper. We used the CrossVit-base-224 and Vit-base-patch16-224 models, pre-trained on ImageNet and provided by timm[10], and fine-tuned the models on CIFAR-10. We compared Mixed Ghost

9. https://github.com/woodyx218/private_vision
10. https://github.com/huggingface/pytorch-image-models

TABLE 14: Results of pre-trained transformers for independent 3 runs. The privacy budget $\varepsilon = 1$ and $\delta = 10^{-5}$. We set the training epochs to 2 and batch size of 5000 as Bu et al.[20] suggested.

Model	Clip method	Test accuracy	t	p-value	Effect size
CrossViT_base_224	Basic clipping	94.71%(±0.14%)	0.78	0.52	0.44
	Mixed Ghost clipping	**94.77%**(±0.13%)			
Vit_base_patch16_224	Basic clipping	**97.34%**(±0.20%)	-16.61	**0.004**	-13.57
	Mixed Ghost clipping	95.05%(±0.13%)			

TABLE 15: Reproduced results for Bu et al. [21]. According to the framework of Section 7.4, both clipping methods achieve similar performance, with neither being statistically superior.

Dataset	Abadi's clipping	Auto clipping	t	p-value	Effect size
MNIST	98.15% (±0.14%)	98.14% (±0.12%)	-0.43	0.71	-0.22
Fashion MNIST	86.65% (±0.35%)	86.72% (±0.32%)	0.37	0.75	0.21

clipping and basic gradient clipping, with $\varepsilon = 1$ and two training epochs, as suggested in the paper (Table 14). Mixed Ghost Clipping *did not* outperform basic gradient clipping in a statistically significant way (Section 7.4). When the model architecture is Vit-base-patch16-224, basic gradient clipping achieves better performance (in this case the effect size is massive and the result is statistically significant). Moreover, we did not observe the out-of-memory problem reported in the original paper when using basic gradient clipping.

While running these experiments we inadvertently achieved a new SoTA performance of **97.34%** for fine-tuning pre-trained models on CIFAR-10 with a privacy budget of $\varepsilon = 1$. In this case, the SoTA result was achieved (accidentally) by using a powerful model architecture and tuning some hyperparameters and we argue it does not constitute a meaningful improvement.

Takeaway 3. *Differences between claimed SoTA results and baselines are sometimes so small that one may accidentally achieve new SoTA results. Such small differences may also not be statistically significant. This highlights the risk with chasing SoTA performance as a strategy for DPML research. Arguably, researchers should focus on designing novel techniques that have a meaningful rationale or are otherwise expected to be reliable and generalizable.*

Bu et al. [21] — Auto clipping. Recall that Bu et al. [21] introduced an alternative gradient clipping technique termed "Auto Clipping." As recommended in their paper, we modified the Opacus library to incorporate this new clipping method and conducted experiments on MNIST and Fashion MNIST. Table 15 shows our results, which are consistent with the performance reported in Bu et al. [21]. In their paper, a stated goal is to improve the hyperparameters search time of DP-SGD by reducing the number of hyperparameters that need to be tuned, which they achieve. However, their method does not outperform the original clipping proposed by Abadi et al. [13]. According to our framework, neither method can be claimed to provide superior performance with statistical significance.

TABLE 16: Our reproduced results for Tramèr and Boneh [19] and compare it to De et al. [17] for ε from 1 to 8. We find that Tramèr and Boneh [19] excel in performance with a limited privacy budget ($\varepsilon = 1$ and $\varepsilon = 2$). However, when the privacy budget exceeds 4, their performance plateaus. (For statistical tests, we take Tramèr and Boneh [19] as proposed method and De et al. [17] as baseline.)

Paper	ε=1	ε=2	ε=4	ε=8
[17]	56.80%(±0.49%)	62.90%(±0.32%)	69.45%(±0.41%)	**78.74%(±0.45%)**
[19]	**60.88%(±0.33%)**	**66.96%(±0.56%)**	69.74%(±0.24%)	72.40%(±0.11%)
t-statistic	21.25	13.35	1.67	-60.93
p-value	**0.002**	**0.006**	0.238	**0.003**
Effect size	9.77	8.90	0.86	-19.36

Takeaway 4. *Claimed improvements of methods may not always achieve statistical significance. Our proposed framework or other statistically valid methods should be used to conclusively determine whether a method truly outperforms its baseline.*

Cattan et al. [22] — First & last fine-tuning. Cattan et al. [22] argue that fine-tuning the first and last layers yields superior results to only fine-tuning the last layer or the entire model. However, in our experiments, we observed the opposite in Table 19: fine-tuning the first and last layers yields similar or worse performance compared to just fine-tuning the last layer or even the whole model in some cases.

A plausible explanation for this discrepancy is that the claims by Cattan et al. are limited to their settings, such as using ResNet on CIFAR-10 and CIFAR-100. For our experiments, we use Vit_base_patch16_224 pre-trained on ImageNet and test their method using multiple different datasets including CIFAR-10, EuroSAT, ISIC 2018, Caltech256, SUN397, and Oxford Pet.

Luo et al. [23] — Sparse fine-tuning. Luo et al. [23] propose fine-tuning the classification layer, normalization layer, and a minor subset (i.e., 1%) of the convolution layer parameters. Our replication corroborates that this fine-tuning approach outperforms fine-tuning the entire model (baseline), across all three datasets they used in their paper. However, we find that the proposed 1% parameter selection is not universally optimal. For instance, selecting 10% of parameters provides comparatively superior performance. We also find that their proposed method does not achieve better performance compared to only fine-tuning the last layer for datasets such as Caltech-256, SUN 397 and Oxford Pet which are not tested by the authors.

Tramèr and Boneh [19] — Hand-crafted features. We reproduced Tramèr and Boneh [19] using their official code and their best method (ScatterNet+CNN) to train a model from scratch on CIFAR-10.[11] We then compare the results to De et al. [17] under varying privacy budgets (Table 16). Tramèr and Boneh [19] outperform De et al. [17] when the privacy budget is limited. However, for $\varepsilon = 8$ (or larger), De et al. [17] provide substantially better performance. Further experiments with an increasing privacy budget show

11. https://github.com/ftramer/Handcrafted-DP

TABLE 17: Our reproduced results on CIFAR-10 and PathMNIST for Tang et al. [24] using different datasets for Phase 1 with $\varepsilon = 1$.

Method	CIFAR-10	PathMNIST
Phase1 w/ Random processes	72.48% (±0.21%)	90.65% (±0.18%)
Phase1 w/ EuroSAT	70.81% (±0.32%)	90.58% (±0.25%)

TABLE 18: FID values between pre-training data (random process data and fine-tuning set — CIFAR-10 and PathMNIST)

Dataset	Random processes	EuroSAT
CIFAR-10 Train	107.78	123.33
CIFAR-10 Test	158.69	160.48
PathMNIST Train	196.61	201.71
PathMNIST Test	196.75	202.64

performance plateauing for Tramèr and Boneh whereas the test accuracy for De et al. keeps increasing.

Takeaway 5. *It is not uncommon for one method to outperform another in one privacy regime but have the reverse occur in a different privacy regime. This underlines the importance of reporting results with a wide range of ε values to ensure comprehensive comparisons between methods.*

Tang et al. [24] — Random-process pretraining. Tang et al. [24] advocate for initially pre-training a model on data produced from random processes, and then fine-tuning it using the private dataset. We replicated their method using their official codebase and validated their performance as shown in Table 17.[12]

Their method essentially initializes the model using random process data instead of public pre-training data. Model performance typically improves with the similarity between pre-training and fine-tuning datasets. In practice, dissimilar datasets to the fine-tuning data are more likely to be publicly available. So this approach raises the question of whether random process data is always beneficial. We evaluate this using the Fréchet Inception Distance (FID) [181] as a measure of the domain gap.

As shown in Table 17, the performance obtained when pre-training on EuroSAT is analogous to that achieved with random process data. The FID values between private datasets (e.g., CIFAR-10 and PathMNIST) and public datasets (e.g., random process data and EuroSAT), are shown in appendix Table 18. Note that we selected these datasets specifically to have large FID values. Interestingly, despite EuroSAT exhibiting a larger domain gap compared to random process data, models pre-trained on EuroSAT still deliver performance metrics that closely align with those trained using random process data.

9. Evaluations Beyond R+R

In this section, we answer research questions RQ3 to RQ6 beyond R+R experiments and investigate the computational cost of different methods. Additional experiments such as empirical privacy measurements are in Appendix B.1.

9.1. RQ3: Which Part of the Model to Fine-tune?

We explore several fine-tuning strategies, drawing from Luo et al. [23] and Cattan et al. [22]. We apply these strategies to six distinct datasets: CIFAR-10, EuroSAT, ISIC 2018, Caltech 256, SUN397, and Oxford Pet. These datasets present a range of domain gaps with respect to the pre-training dataset. For this study, we use the Vit_base_patch16_224 model, pre-trained on ImageNet, and fine-tuned it with 5 epochs. We present the results in Table 19. To ensure a fair comparison, we use fine-tuning the whole model as a baseline. In addition to applying methods from the DPML literature, we experimented with alternatives. Instead of magnitude-based parameter selection, we test random parameter selection for 1%, 2%, and 10% of the parameters ("Random Subset"). Also, rather than selecting parameters dispersed across different blocks of the ViT model, we tested randomly selecting 2, 3, or 6 blocks for fine-tuning, a strategy we call "partial training."

Results (Table 19) show that both alternative approaches sometimes match or surpass the performance of Luo et al. [23] for the first three datasets. Considering all 6 datasets, only fine-tuning the last layer achieves stable good performance. This suggests that fine-tuning only a subset of the model is the important variable, not the specific subset, or that the fine-tuning strategy is dataset-specific such as domain gap and data size per class.

Takeaway 6. *Although several papers propose sophisticated strategies for DP fine-tuning models, we find that none of them consistently outperform alternatives across all datasets. It appears that only fine-tuning the last layer performs well across datasets.*

9.2. RQ4: Can Different Methods be Combined?

Can two or more methods from the selected papers be combined to provide further improvements? Given that the techniques are different in nature they are compatible and can in many cases be applied in combination (e.g., self-augmentation + changing order + ScaleNorm).

Since large batch sizes and high noise levels can be combined with all other improvements, we use this as the default setting for all experiments. We find that order switching (Sander et al. [15]) and the techniques proposed by De et al. [17] can be combined and achieve 78.10% test accuracy on CIFAR-10. By contrast, combining Mixed Ghost Clipping with order switching slightly decreased performance for order 3.

Combining ScaleNorm (Klause et al. [14]) with De et al.'s techniques or order switching did not significantly increase performance. Specifically, combining ScaleNorm with De et al.'s techniques achieved an average test accuracy of 77.43%, which is lower than only applying De et al.'s

12. https://github.com/inspire-group/DP-RandP

TABLE 19: Test accuracy using Vit_base_patch16_224 on CIFAR-10, EuroSAT, and ISIC 2018 using different fine-tuning methods.

					Luo et al. [23]			Random Subset			Partial Training		
Dataset	Whole model	First and last [22]	Last only	Non-private	1%	2%	10%	1%	2%	10%	2	3	6
CIFAR-10	97.82%(0.08%)	95.41%(0.14%)	95.68%(0.12%)	98.34%(0.04%)	97.85%(0.06%)	97.86%(0.09%)	**97.93%(0.11%)**	97.89%(0.06%)	97.92%(0.08%)	97.91%(0.08%)	97.91%(0.12%)	97.90%(0.10%)	97.85%(0.08%)
EuroSAT	95.75%(0.44%)	93.74%(0.17%)	94.13%(0.15%)	98.69%(0.07%)	95.78%(0.21%)	95.23%(0.75%)	95.87%(0.22%)	95.67%(0.72%)	95.94%(0.26%)	**96.15%(0.30%)**	95.74%(0. 13%)	95.78%(0.13%)	95.29%(0.56%)
ISIC 2018	72.34%(0.16%)	67.78%(0.27%)	67.76%(0.31%)	90.58%(0.49%)	71.78%(0.58%)	71.79%(0.83%)	**72.51%(0.46%)**	72.15%(0.41%)	71.73%(0.59%)	72.41%(0.35%)	72.31%(0.57%)	70.82%(0.84%)	71.44%(0.38%)
Caltech 256	30.55%(0.19%)	80.58%(0.14%)	**80.74%(0.15%)**	95.61%(0.14%)	42.56%(0.25%)	31.05%(0.29%)	30.37%(0.28%)	31.63%(0.23%)	29.56%(0.31%)	29.53%(0.26%)	31.61%(0.36%)	29.46%(0.34%)	30.65%(0.39%)
SUN 397	43.53%(0.24%)	67.78%(0.29%)	**67.90%(0.24%)**	84.49%(0.34%)	42.93%(0.35%)	43.51%(0.37%)	43.96%(0.32%)	43.42%(0.36%)	44.12%(0.43%)	43.56%(0.37%)	43.47%(0.39%)	42.38%(0.44%)	42.74%(0.31%)
Oxford Pet	34.81%(0.34%)	73.80%(0.21%)	**73.92%(0.35%)**	92.89%(0.19%)	42.30%(0.39%)	39.62%(0.41%)	36.48%(0.37%)	35.79%(0.34%)	41.26%(0.45%)	38.81%(0.33%)	37.01%(0.27%)	38.76%(0.39%)	34.83%(0.44%)

techniques (77.79%) and less than the 1% performance boost claimed in the paper [14]. Similarly, when combining ScaleNorm with De et al.'s techniques and order switching the average test accuracy increased by only 0.09% (smaller than the standard deviation across different runs).

Takeaway 7. *Combining DPML methods, even orthogonal ones, often does not provide cumulative improvements. Some combinations of methods actually decrease performance. Notable exceptions include large batch sizes and augmentation multiplicity.*

9.3. RQ5: What are the Most Promising Methods?

Our experiments show that while we could reproduce all selected papers, **only seven out of eleven** papers including Dormann et al. [16], Sander et al. [15], De et al. [17], Bao et al. [18], Luo et al. [23], Tramèr and Boneh [19] and Tang et al. [24] reliably and consistently achieved their claimed performance improvements. The other four papers did not for various reasons, including not delivering consistent improvements outside of the narrow experimental setting in their paper.

As a further demonstration of our proposed framework, we summarize its application to some of the evaluated techniques in Table 21. We observe that (in this case) feature selection and augmentation multiplicity techniques achieve substantial improvements (i.e., large effect sizes) that are also statistical significance. By contrast, the clipping and fine-tuning techniques in this case do not achieve statistical significance (or large effect size).

Further discussion of statistical power and number of runs using Table 21 for illustration is warranted. Reporting results for a few runs (sometimes a single run) is common practice in DPML research [20], [14], [23], [18], [15]. Accordingly, we used $n = 3$ for the experiments in the table. Ideally, research should involve more extensive testing (e.g., $n = 20$), but the high computational cost of DPML training makes this challenging for many researchers. We discuss the computational overhead of DPML in the next subsection.

Reporting too few runs may result in an under-powered test, and being unable to establish whether the method provides notable improvements. However, as discussed in Section 7.4, a large enough effect size can overcome a small number of runs. Power analysis on the results of Table 21 show that for [19], [17] the test has plenty of power due to the large effect sizes. By contrast, for the other two methods in the table, the effect size is too small.

We reiterate that the goal of our R+R experiment is not to assign blame or cast any specific work in a negative light. Rather we seek to identify what methods and techniques work best and how to perform the evaluation of DPML to achieve high degrees of reproducibility (Section 10).

In the rest of this section, we discuss insights from investigations and highlight promising future research directions.

Model architecture, feature, and hyperparameter selection. Whenever possible comprehensive searches over model architecture [25], [29], features [19], and hyperparameters [17], [15] should be performed as all of these factors play a pivotal role in DPML performance. Ideally, the privacy cost of hyperparameter searches should be accounted for, which will likely reduce the obtained performance. There is promising recent work in this direction such as[36], [182], [183], but more research is necessary.

Large batch sizes. Larger batch sizes and higher noise levels provide consistently higher performance according to numerous studies [16], [17], [15] and our own empirical findings. However, large batch size brings a problem of high computational cost for hyperparameter tuning. To address this, techniques like Sander et al. [15] can be applied.

Clipping strategies. Although there is a plethora of papers exploring the use of clipping techniques to improve DPML, we find that such methods provide little improvement. Notably, several recent works [57], [55], [39] investigate the effect of clipping and how it may bias the learning process.

Augmentation multiplicity. The augmentation multiplicity (self-augmentation) approaches of De et al. [17] and Bao et al. [18] appear to deliver consistent and significant improvements in model performance. However, these works only explore a small subset of possible augmentations, so a promising avenue for future research is to comprehensively study the potential benefits of various data augmentation techniques.

Architecture-specific methods. Some methods such as changing the order of layers [15] seem to provide improvements while others such as ScaleNorm [14] did not in our reproduction. Practitioners should be caution before adopting methods specifically tailored to an architecture as improvements may not be consistently obtained.

13. Because the authors do not report standard deviation, we cannot determine whether the improvement is larger than the standard deviation.

14. We use the latest version of Opacus so the time comparison may be different from reported in their paper.

15. We use this method as the DP-SGD baseline as it is well-known.

16. Because feature pre-processing of this method is not counted, the running time for this method is lower.

17. We implement this technique ourself so it is not be optimized for minimizing running time.

TABLE 20: Detail of 11 selected papers about their Generalizability and Reliability. Note that effect size is computed using our reproduced results of the baseline they used in their papers and their proposed methods. We measured the running time using one A100 GPU, running each method for 3 epochs on CIFAR-10. For the baseline of time comparison, DP-SGD from scratch took 90.61 seconds, while SGD from scratch took 10.62 seconds. For fine-tuning, DP-SGD took 571.16 seconds, compared to 388.76 seconds for SGD. The Time Overhead in the table is the slowdown factor relative to the corresponding SGD baseline in the same setting (i.e., method time divided baseline time).

Paper	Method(s)	Generalizability			Reliability					Effect size	Time Overhead	
		Multi-Settings	Datasets	Multi-ε	Open source	Param. Search	Account	Multi-runs	Statistically significant	Ablation		vs SGD	vs DP-SGD
Bu et al. [20]	Clipping techniques	×	✓	✓	✓	×	×	N/A[13]		×	0.44	1.76	1.20 [14]
Bu et al. [21]	Clipping techniques	✓	✓	✓	✓	×	✓	×		✓	-0.22	2.17	1.48
Klause et al. [14]	Model architecture	×	✓	×	×	×	×	N/A		✓	1.12	9.57	1.12
Sander et al. [15]	Model architecture	×	✓	×	✓	×	✓	✓		✓	5.28	8.89	1.04
Dormann et al. [16]	Hyperparameter tuning	×	✓	×	✓	×	✓	✓		✓	1.91	8.53	1.00 [15]
De et al. [17]	Augmentation multiplicity	✓	✓	✓	✓	✓	✓	✓		✓	12.22	93.90	11.01
Bao et al. [18]	Augmentation multiplicity	✓	✓	✓	✓	✓	✓	✓		✓	13.62	99.93	11.71
Tramèr and Boneh [19]	Feature selection	×	✓	✓	✓	×	✓	✓		✓	9.77	1.05	0.12 [16]
Cattan et al. [22]	Fine-tuning technique	N/A	×	✓	×	×	×	N/A		×	-21.14	0.43	0.29
Luo et al. [23]	Fine-tuning technique	N/A	✓	✓	×	×	×	N/A		✓	0.42	11.84	8.06 [17]
Tang et al. [24]	Fine-tuning technique	N/A	✓	✓	✓	✓	✓	✓		✓	46.76	93.16	10.92

TABLE 21: Summary of statistical framework results. We use CIFAR-10 for all, except for Bu et al. [21] where we used MNIST. We set run times $n = 3$. We use Vit-base-patch16-224 for Bu et al. [21] and Luo et al. [23], WideResNet-16-4 for De et al. [17] and ScatterNet+CNN for Tramèr and Boneh [19].

Method	Paper	t	p-value	Effect size
Clipping technique	Bu et al. [21]	-0.43	0.71	-0.22
Feature selection	Tramèr and Boneh [19]	21.25	**0.002**	9.77
Augmentation multiplicity	De et al. [17]	22.01	**0.002**	12.22
Fine-tuning technique	Luo et al. [23]	0.49	0.67	0.42

TABLE 22: Proposed checklist for DPML.

Checklist item		
Generalizability	Evaluation in different settings	☐
	Evaluation with different datasets	☐
	Evaluation with different model architectures	☐
	Evaluation for different privacy requirements	☐
	Evaluation of combination with other techniques	☐
Reliability	Code open source	☐
	Results of multiple runs are reported	☐
	Improvement is statistically significant	☐
	Accounts for hyperparameter search	☐
	Includes ablation study	☐

Fine-tuning methods. Fine-tuning a subset of a model's parameters with DP appears to be a viable strategy. However, no single method except only fine-tuning the last layer performs best across datasets and architecture in our experiments. Practitioners should attempt only fine-tuning the last layer and apply whatever method gives the best performance for their particular use cases rather than adopting whole cloth any of the methods that claim to provide SoTA results.

Pre-training and public data. Fine-tuning a pre-trained model with DP is a consistent way to achieve performance closer to the non-private setting than training from scratch. That said, the more similar the pre-training data and fine-tuning datasets are, the better the performance. When no suitable public dataset is available for pre-training, techniques such as Tramèr and Boneh [19] or Tang et al. [24] provide a viable alternative. However, our results suggest that using an unrelated public dataset for pre-training provides comparable results.

9.4. Computational Cost

We evaluate computational cost by measuring the average GPU time per epoch for different methods over 3 epochs, all run on a single A100 GPU. Our benchmarks included training WRN-16-4 from scratch and fine-tuning the ViT model on CIFAR-10 using both SGD and DP-SGD. Results showed: Training from scratch with DP-SGD required 90.61 sec/epoch, whereas training with SGD took only 10.62 sec/epoch. For fine-tuning, DP-SGD took 571.16 sec/epoch, while SGD took 388.76 sec/epoch. We calculate the time overhead as a ratio of time per epoch to both SGD and DP-SGD. Results are shown in Table 20.

We observe that more complex training strategies, such as those proposed by De et al. [17], Bao et al. [18], Luo et al. [23], and Tang et al. [24], require significantly longer training times per epoch. On the other hand, techniques like feature selection [19] or fine-tuning fewer layers [22] can substantially reduce computational time.

10. Towards Reproducibility & Replicability

We distill our insights from our R+R experiment into a set of proposed guidelines and a checklist. We hope researchers who seek to evaluate new methods can follow these guidelines to maximize the chance of reproducibility.

Criteria. We propose to think of reproducibility and replicability along two separate axes: generalizability, and reliability.

- ***Generalizability*** assess whether the proposed method's benefits are likely to generalize outside of the narrow experimental setting demonstrated. For example, if a method was shown to provide improvements in multiple settings, varied datasets, and multiple privacy regimes, it is more likely to provide similar improvements in other contexts than a method only evaluated on a single task, dataset, and privacy budget.
- ***Reliability*** assess the extent to which evaluation methodology suggests results reported are reliable, stable, and likely to reproduce. For instance, results from a single run

are less reliable than those averaged over five independent runs. Additionally, reliability involves determining whether performance improvements are truly due to the proposed method, especially when combined techniques or unique settings might skew results. In such cases, the apparent enhancements in performance may stem from these ancillary factors rather than from the intrinsic merits of the proposed method.

The checklist is shown in Table 22.

Selected papers. We grade our 11 selected papers according to our checklist. Results are shown in Table 20. We found that Dormann et al. [16], De et al.[17], Bao et al. [18], Tramèr and Boneh [19], Luo et al. [23], Sander et al.[15] and Tang et al. [24] performed well overall according to our two criteria. This is not the case for Klause et al. [14], Bu et al. [20], and other works. For example, Klause et al. [14] lack reliability (no open source and report results for only one run), Bu et al. [20] lack generalizability (only pre-trained tasks) and reliability (no ablation experiments), Bu et al. [21] lacks reliability (report improvement without statistical difference) while Cattan et al. [22] also lack reliability (no open source code, report results with one run).

Checklist details. We describe the items of the checklist, and their rationale, and illustrate their utility through examples in Appendix C.

11. Conclusion and Future work

We conducted a R+R investigation on 11 recent SoTA DPML techniques, which revealed significant variations in their reproducibility. We identified the inherent randomness of DPML as a challenge and proposed a statistical framework to deal with it. We distilled our insights into a set of comprehensive guidelines and a checklist to standardize future DPML research. Our investigation also uncovered open questions for future research, such as determining the optimal fine-tuning strategy with DP. The training convergence behavior of different DP methods is another possible direction for future work, and so is reproducibility of DPML methods for types of data other than images.

Acknowledgments

This work was supported in part by the National Science Foundation under CNS-2055123 and CNS-1933208. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] S. Kapoor and A. Narayanan, "Leakage and the reproducibility crisis in machine-learning-based science," *Patterns*, vol. 4, no. 9, 2023.

[2] J. Pineau, P. Vincent-Lamarre, K. Sinha, V. Larivière, A. Beygelzimer, F. d'Alché Buc, E. Fox, and H. Larochelle, "Improving reproducibility in machine learning research (a report from the neurips 2019 reproducibility program)," *The Journal of Machine Learning Research*, vol. 22, no. 1, pp. 7459–7478, 2021.

[3] H. Semmelrock, S. Kopeinik, D. Theiler, T. Ross-Hellauer, and D. Kowald, "Reproducibility in machine learning-driven research," *arXiv preprint arXiv:2307.10320*, 2023.

[4] A. L. Beam, A. K. Manrai, and M. Ghassemi, "Challenges to the reproducibility of machine learning models in health care," *Jama*, vol. 323, no. 4, pp. 305–306, 2020.

[5] M. McDermott, S. Wang, N. Marinsek, R. Ranganath, L. Foschini, and M. Ghassemi, "Reproducibility in machine learning for health research: Still a ways to go." *Science Translational Medicine*, vol. 13, no. 586, pp. eabb1655–eabb1655, 2021.

[6] B. J. Heil, M. M. Hoffman, F. Markowetz, S.-I. Lee, C. S. Greene, and S. C. Hicks, "Reproducibility standards for machine learning in the life sciences," *Nature Methods*, vol. 18, no. 10, pp. 1132–1135, 2021.

[7] D. Olszewski, A. Lu, C. Stillman, K. Warren, C. Kitroser, A. Pascual, D. Ukirde, K. Butler, and P. Traynor, "" get in researchers; we're measuring reproducibility": A reproducibility study of machine learning papers in tier 1 security conferences," in *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, 2023, pp. 3433–3459.

[8] N. Daoudi, K. Allix, T. F. Bissyandé, and J. Klein, "Lessons learnt on reproducibility in machine learning based android malware detection," *Empirical Software Engineering*, vol. 26, no. 4, p. 74, 2021.

[9] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," vol. 7, no. 3, 2016, pp. 17–51.

[10] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, "Our data, ourselves: Privacy via distributed noise generation," in *Annual international conference on the theory and applications of cryptographic techniques*. Springer, 2006, pp. 486–503.

[11] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014.

[12] R. M. Gower, N. Loizou, X. Qian, A. Sailanbayev, E. Shulgin, and P. Richtárik, "Sgd: General analysis and improved rates," in *International Conference on Machine Learning*. PMLR, 2019, pp. 5200–5209.

[13] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 308–318.

[14] H. Klause, A. Ziller, D. Rueckert, K. Hammernik, and G. Kaissis, "Differentially private training of residual networks with scale normalisation," *arXiv preprint arXiv:2203.00324*, 2022.

[15] T. Sander, P. Stock, and A. Sablayrolles, "TAN without a burn: Scaling laws of DP-SGD," in *Proceedings of the 40th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 202. PMLR, 23–29 Jul 2023, pp. 29 937–29 949.

[16] F. Dörmann, O. Frisk, L. N. Andersen, and C. F. Pedersen, "Not all noise is accounted equally: How differentially private learning benefits from large sampling rates," in *2021 IEEE 31st International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2021, pp. 1–6.

[17] S. De, L. Berrada, J. Hayes, S. L. Smith, and B. Balle, "Unlocking high-accuracy differentially private image classification through scale," *arXiv preprint arXiv:2204.13650*, 2022.

[18] W. Bao, F. Pittaluga, V. K. BG, and V. Bindschaedler, "Dp-mix: Mixup-based data augmentation for differentially private learning," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[19] F. Tramèr and D. Boneh, "Differentially private learning needs better features (or much more data)," in *International Conference on Learning Representations*, 2020.

[20] Z. Bu, J. Mao, and S. Xu, "Scalable and efficient training of large convolutional neural networks with differential privacy," *Advances in Neural Information Processing Systems*, vol. 35, pp. 38 305–38 318, 2022.

[21] Z. Bu, Y.-X. Wang, S. Zha, and G. Karypis, "Automatic clipping: Differentially private deep learning made easier and stronger," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[22] Y. Cattan, C. A. Choquette-Choo, N. Papernot, and A. Thakurta, "Fine-tuning with differential privacy necessitates an additional hyperparameter search," *arXiv preprint arXiv:2210.02156*, 2022.

[23] Z. Luo, D. J. Wu, E. Adeli, and L. Fei-Fei, "Scalable differential privacy with sparse network finetuning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 5059–5068.

[24] X. Tang, A. Panda, V. Sehwag, and P. Mittal, "Differentially private image classification by learning priors from random processes," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[25] W. Bao, L. A. Bauer, and V. Bindschaedler, "On the importance of architecture and feature selection in differentially private machine learning," *arXiv preprint arXiv:2205.06720*, 2022.

[26] P. Subramani, N. Vadivelu, and G. Kamath, "Enabling fast differentially private sgd via just-in-time compilation and vectorization," *Advances in Neural Information Processing Systems*, vol. 34, pp. 26 409–26 421, 2021.

[27] A. Ross, V. L. Willson, A. Ross, and V. L. Willson, "Paired samples t-test," *Basic and Advanced Statistical Tests: Writing Results Sections and Creating Tables and Figures*, pp. 17–19, 2017.

[28] J. Cohen, *Statistical power analysis for the behavioral sciences*. Routledge, 2013.

[29] A. Priyanshu, R. Naidu, F. Mireshghallah, and M. Malekzadeh, "Efficient hyperparameter optimization for differentially private deep learning," *arXiv preprint arXiv:2108.03888*, 2021.

[30] A. Cheng, J. Wang, X. S. Zhang, Q. Chen, P. Wang, and J. Cheng, "Dpnas: Neural architecture search for deep learning with differential privacy," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 6, 2022, pp. 6358–6366.

[31] N. W. Remerscheid, A. Ziller, D. Rueckert, and G. Kaissis, "Smoothnets: Optimizing cnn architecture design for differentially private deep learning," *arXiv preprint arXiv:2205.04095*, 2022.

[32] W. Wang, T. Wang, L. Wang, N. Luo, P. Zhou, D. Song, and R. Jia, "Dplis: Boosting utility of differentially private deep learning via randomized smoothing," *Proceedings on Privacy Enhancing Technologies*, vol. 2021, no. 4, pp. 163–183, 2021.

[33] A. S. Shamsabadi and N. Papernot, "Losing less: A loss for differentially private deep learning," 2021.

[34] N. Papernot, A. Thakurta, S. Song, S. Chien, and Ú. Erlingsson, "Tempered sigmoid activations for deep learning with differential privacy," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 10, 2021, pp. 9312–9321.

[35] V. Singhal and T. Steinke, "Privately learning subspaces," *Advances in Neural Information Processing Systems*, vol. 34, pp. 1312–1324, 2021.

[36] N. Papernot and T. Steinke, "Hyperparameter tuning with renyi differential privacy," in *International Conference on Learning Representations*, 2021.

[37] H. Wang, S. Gao, H. Zhang, W. Su, and M. Shen, "Dp-hypo: An adaptive private framework for hyperparameter optimization," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[38] J. Park, Y. Choi, and J. Lee, "In-distribution public data synthesis with diffusion models for differentially private image classification," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 12 236–12 246.

[39] X. Chen, S. Z. Wu, and M. Hong, "Understanding gradient clipping in private sgd: A geometric perspective," *Advances in Neural Information Processing Systems*, vol. 33, pp. 13 773–13 782, 2020.

[40] J. Lee and D. Kifer, "Scaling up differentially private deep learning with fast per-example gradient clipping," *Proceedings on Privacy Enhancing Technologies*, vol. 2021, no. 1, 2021.

[41] Z. Bu, S. Gopi, J. Kulkarni, Y. T. Lee, H. Shen, and U. Tantipongpipat, "Fast and memory efficient differentially private-sgd via jl projections," *Advances in Neural Information Processing Systems*, vol. 34, pp. 19 680–19 691, 2021.

[42] D. Yu, H. Zhang, and W. Chen, "Improve the gradient perturbation approach for differentially private optimization," 2018.

[43] V. Pichapati, A. T. Suresh, F. X. Yu, S. J. Reddi, and S. Kumar, "Adaclip: Adaptive clipping for private sgd," *arXiv preprint arXiv:1908.07643*, 2019.

[44] G. Andrew, O. Thakkar, B. McMahan, and S. Ramaswamy, "Differentially private learning with adaptive clipping," *Advances in Neural Information Processing Systems*, vol. 34, pp. 17 455–17 466, 2021.

[45] J. Du, S. Li, M. Feng, and S. Chen, "Dynamic differential-privacy preserving sgd," *arXiv preprint arXiv:2111.00173*, 2021.

[46] G. Lin, H. Yan, G. Kou, T. Huang, S. Peng, Y. Zhang, and C. Dong, "Understanding adaptive gradient clipping in dp-sgd, empirically," *International Journal of Intelligent Systems*, 2022.

[47] A. Golatkar, A. Achille, Y.-X. Wang, A. Roth, M. Kearns, and S. Soatto, "Mixed differential privacy in computer vision," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 8376–8386.

[48] X. Yang, H. Zhang, W. Chen, and T.-Y. Liu, "Normalized/clipped sgd with perturbation for differentially private non-convex optimization," *arXiv preprint arXiv:2206.13033*, 2022.

[49] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, "Learning differentially private recurrent language models," in *International Conference on Learning Representations*, 2018.

[50] K. L. van der Veen, R. Seggers, P. Bloem, and G. Patrini, "Three tools for practical differential privacy," *arXiv preprint arXiv:1812.02890*, 2018.

[51] H. Liu, C. Li, B. Liu, P. Wang, S. Ge, and W. Wang, "Differentially private learning with grouped gradient clipping," in *ACM Multimedia Asia*, 2021, pp. 1–7.

[52] D. Xu, W. Du, and X. Wu, "Removing disparate impact of differentially private stochastic gradient descent on model accuracy," *arXiv preprint arXiv:2003.03699*, 2020.

[53] Z. Bu, H. Wang, Q. Long, and W. J. Su, "On the convergence and calibration of deep learning with differential privacy," *arXiv preprint arXiv:2106.07830*, 2021.

[54] T. Xia, S. Shen, S. Yao, X. Fu, K. Xu, X. Xu, and X. Fu, "Differentially private learning with per-sample adaptive clipping," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 9, 2023, pp. 10 444–10 452.

[55] A. Koloskova, H. Hendrikx, and S. U. Stich, "Revisiting gradient clipping: Stochastic bias and tight convergence guarantees," in *ICML 2023-40th International Conference on Machine Learning*, 2023.

[56] H. Xiao, J. Wan, and S. Devadas, "Geometry of sensitivity: Twice sampling and hybrid clipping in differential privacy with optimal gaussian noise and application to deep learning," *arXiv preprint arXiv:2309.02672*, 2023.

[57] H. Xiao, Z. Xiang, D. Wang, and S. Devadas, "A theory to instruct differentially-private learning via clipping bias reduction," in *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, 2023, pp. 2170–2189.

[58] H. Mehta, A. Thakurta, A. Kurakin, and A. Cutkosky, "Large scale transfer learning for differentially private image classification," *arXiv preprint arXiv:2205.02973*, 2022.

[59] Z. Bu, Y.-X. Wang, S. Zha, and G. Karypis, "Differentially private bias-term only fine-tuning of foundation models," *arXiv preprint arXiv:2210.00036*, 2022.

[60] E. Amid, A. Ganesh, R. Mathews, S. Ramaswamy, S. Song, T. Steinke, V. M. Suriyakumar, O. Thakkar, and A. Thakurta, "Public data-assisted mirror descent for private model training," in *International Conference on Machine Learning*. PMLR, 2022, pp. 517–535.

[61] S. Hoory, A. Feder, A. Tendler, S. Erell, A. Peled-Cohen, I. Laish, H. Nakhost, U. Stemmer, A. Benjamini, A. Hassidim, and Y. Matias, "Learning and evaluating a differentially private pre-trained language model," in *Findings of the Association for Computational Linguistics: EMNLP 2021*, M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, Eds. Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 1178–1189. [Online]. Available: https://aclanthology.org/2021.findings-emnlp.102/

[62] X. Li, F. Tramer, P. Liang, and T. Hashimoto, "Large language models can be strong differentially private learners," in *International Conference on Learning Representations*, 2021.

[63] D. Yu, S. Naik, A. Backurs, S. Gopi, H. A. Inan, G. Kamath, J. Kulkarni, Y. T. Lee, A. Manoel, L. Wutschitz, S. Yekhanin, and H. Zhang, "Differentially private fine-tuning of language models," in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=Q42f0dfjECO

[64] F. Tramèr, G. Kamath, and N. Carlini, "Considerations for differentially private learning with large-scale public pretraining," *arXiv preprint arXiv:2212.06470*, 2022.

[65] A. Lowy, Z. Li, T. Huang, and M. Razaviyayn, "Optimal differentially private learning with public data," *arXiv preprint arXiv:2306.15056*, 2023.

[66] D. Yu, S. Gopi, J. Kulkarni, Z. Lin, S. Naik, T. L. Religa, J. Yin, and H. Zhang, "Selective pre-training for private fine-tuning," *arXiv preprint arXiv:2305.13865*, 2023.

[67] F. Mireshghallah, A. Backurs, H. A. Inan, L. Wutschitz, and J. Kulkarni, "Differentially private model compression," *Advances in Neural Information Processing Systems*, vol. 35, pp. 29 468–29 483, 2022.

[68] Z. Wang, R. Zhu, D. Zhou, Z. Zhang, J. Mitchell, H. Tang, and X. Wang, "{DPAdapter}: Improving differentially private deep learning through noise tolerance pre-training," in *33rd USENIX Security Symposium (USENIX Security 24)*, 2024, pp. 991–1008.

[69] J. Zhu and M. Blaschko, "Differentially private sgd with sparse gradients," *arXiv preprint arXiv:2112.00845*, 2021.

[70] S. P. Kasiviswanathan, "Sgd with low-dimensional gradients with applications to private and distributed learning," in *Uncertainty in Artificial Intelligence*. PMLR, 2021, pp. 1905–1915.

[71] H. Sidahmed, Z. Xu, A. Garg, Y. Cao, and M. Chen, "Efficient and private federated learning with partially trainable networks," *arXiv preprint arXiv:2110.03450*, 2021.

[72] D. Yu, H. Zhang, W. Chen, J. Yin, and T.-Y. Liu, "Large scale private learning via low-rank reparametrization," in *International Conference on Machine Learning*. PMLR, 2021, pp. 12 208–12 218.

[73] R. Ito, S. P. Liew, T. Takahashi, Y. Sasaki, and M. Onizuka, "Scaling private deep learning with low-rank and sparse gradients," *arXiv preprint arXiv:2207.02699*, 2022.

[74] H. Zhang, I. Mironov, and M. Hejazinia, "Wide network learning with differential privacy," *arXiv preprint arXiv:2103.01294*, 2021.

[75] H. Asi, V. Feldman, T. Koren, and K. Talwar, "Private stochastic convex optimization: Optimal rates in l1 geometry," 2021.

[76] M. Nasr, R. Shokri, and A. Houmansadr, "Improving deep learning with differential privacy using gradient encoding and denoising," *arXiv preprint arXiv:2007.11524*, 2020.

[77] Y. Zhou, S. Wu, and A. Banerjee, "Bypassing the ambient dimension: Private sgd with gradient subspace identification," in *International Conference on Learning Representations*, 2020.

[78] D. Yu, H. Zhang, W. Chen, and T.-Y. Liu, "Do not let privacy overbill utility: Gradient embedding perturbation for private learning," in *International Conference on Learning Representations*, 2020.

[79] X. Gu, G. Kamath, and Z. S. Wu, "Choosing public datasets for private machine learning via gradient subspace distance," *arXiv preprint arXiv:2303.01256*, 2023.

[80] C. Feng, N. Xu, W. Wen, P. Venkitasubramaniam, and C. Ding, "Spectral-dp: Differentially private deep learning through spectral perturbation and filtering," in *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, 2023, pp. 1944–1960.

[81] S. Asoodeh, J. Liao, F. P. Calmon, O. Kosut, and L. Sankar, "A better bound gives a hundred rounds: Enhanced privacy guarantees via f-divergences," in *2020 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2020, pp. 920–925.

[82] P. Harremoës and I. Vajda, "On pairs of f-divergences and their joint range," *IEEE Transactions on Information Theory*, vol. 57, no. 6, pp. 3230–3235, 2011.

[83] A. Koskela, J. Jälkö, and A. Honkela, "Computing tight differential privacy guarantees using fft," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 2560–2569.

[84] A. Koskela, J. Jälkö, L. Prediger, and A. Honkela, "Tight differential privacy for discrete-valued mechanisms and for the subsampled gaussian mechanism using fft," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2021, pp. 3358–3366.

[85] A. Koskela and A. Honkela, "Computing differential privacy guarantees for heterogeneous compositions using fft," in *International Conference on Learning Representations*, 2021.

[86] S. Gopi, Y. T. Lee, and L. Wutschitz, "Numerical composition of differential privacy," *Advances in Neural Information Processing Systems*, vol. 34, pp. 11 631–11 642, 2021.

[87] R. Chourasia, J. Ye, and R. Shokri, "Differential privacy dynamics of langevin diffusion and noisy gradient descent," *Advances in Neural Information Processing Systems*, vol. 34, pp. 14 771–14 781, 2021.

[88] J. Ye and R. Shokri, "Differentially private learning needs hidden state (or much faster convergence)," *Advances in neural information processing systems*, 2022.

[89] M. Nasr, J. Hayes, T. Steinke, B. Balle, F. Tramèr, M. Jagielski, N. Carlini, and A. Terzis, "Tight auditing of differentially private machine learning," *arXiv preprint arXiv:2302.07956*, 2023.

[90] T. Steinke, M. Nasr, and M. Jagielski, "Privacy auditing with one (1) training run," *arXiv preprint arXiv:2305.08846*, 2023.

[91] M. Jagielski, J. Ullman, and A. Oprea, "Auditing differentially private machine learning: How private is private sgd?" *Advances in Neural Information Processing Systems*, vol. 33, pp. 22 205–22 216, 2020.

[92] M. Nasr, S. Songi, A. Thakurta, N. Papernot, and N. Carlin, "Adversary instantiation: Lower bounds for differentially private machine learning," in *2021 IEEE Symposium on security and privacy (SP)*. IEEE, 2021, pp. 866–882.

[93] F. Lu, J. Munoz, M. Fuchs, T. LeBlond, E. Zaresky-Williams, E. Raff, F. Ferraro, and B. Testa, "A general framework for auditing differentially private machine learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 4165–4176, 2022.

[94] Y. Zhou, X. Chen, M. Hong, Z. S. Wu, and A. Banerjee, "Private stochastic non-convex optimization: Adaptive algorithms and tighter generalization bounds," *arXiv preprint arXiv:2006.13501*, 2020.

[95] J. Lee and D. Kifer, "Concentrated differentially private gradient descent with adaptive per-iteration privacy budget," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 1656–1665.

[96] Z. Xu, S. Shi, A. X. Liu, J. Zhao, and L. Chen, "An adaptive and fast convergent approach to differentially private deep learning," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE, 2020, pp. 1867–1876.

[97] L. Wang, B. Jayaraman, D. Evans, and Q. Gu, "Efficient privacy-preserving stochastic nonconvex optimization," *arXiv preprint arXiv:1910.13659*, 2019.

[98] X. Ding, L. Chen, P. Zhou, W. Jiang, and H. Jin, "Differentially private deep learning with iterative gradient descent optimization," *ACM/IMS Transactions on Data Science (TDS)*, vol. 2, no. 4, pp. 1–27, 2022.

[99] H. Xiao, J. Wan, and S. Devadas, "Differentially private deep learning with modelmix," *arXiv preprint arXiv:2210.03843*, 2022.

[100] Z. Bu, Y.-X. Wang, S. Zha, and G. Karypis, "Differentially private optimization on large model at small cost," *arXiv preprint arXiv:2210.00038*, 2022.

[101] P. Kairouz, M. Ribero, K. Rush, and A. Thakurta, "Fast dimension independent private adagrad on publicly estimated subspaces," *arXiv preprint arXiv:2008.06570*, 2020.

[102] L. Xiang, W. Li, J. Yang, X. Wang, and B. Li, "Differentially-private deep learning with directional noise," *IEEE Transactions on Mobile Computing*, 2021.

[103] T. Li, M. Zaheer, S. Reddi, and V. Smith, "Private adaptive optimization with side information," in *International Conference on Machine Learning*. PMLR, 2022, pp. 13 086–13 105.

[104] T. Li, M. Zaheer, K. Liu, S. J. Reddi, H. B. McMahan, and V. Smith, "Differentially private adaptive optimization with delayed preconditioners," in *The Eleventh International Conference on Learning Representations*, 2022.

[105] J. Wei, E. Bao, X. Xiao, and Y. Yang, "Dpis: An enhanced mechanism for differentially private sgd with importance sampling," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, 2022, pp. 2885–2899.

[106] H. Xiao, Z. Yang, and G. E. Suh, "Trustworthy machine learning through data-specific indistinguishability," in *Forty-second International Conference on Machine Learning*, 2025. [Online]. Available: https://openreview.net/forum?id=JRMoeNZgNl

[107] X. Zhang, Z. Bu, B. Balle, M. Hong, M. Razaviyayn, and V. Mirrokni, "DiSK: Differentially private optimizer with simplified kalman filter for noise reduction," in *The Thirteenth International Conference on Learning Representations*, 2025. [Online]. Available: https://openreview.net/forum?id=Lfy9q7Icp9

[108] N. Papernot, M. Abadi, U. Erlingsson, I. Goodfellow, and K. Talwar, "Semi-supervised knowledge transfer for deep learning from private training data," *International Conference on Learning Representations*, 2017.

[109] A. Uniyal, R. Naidu, S. Kotti, S. Singh, P. J. Kenfack, F. Mireshghallah, and A. Trask, "Dp-sgd vs pate: Which has less disparate impact on model accuracy?" *arXiv preprint arXiv:2106.12576*, 2021.

[110] N. Papernot, S. Song, I. Mironov, A. Raghunathan, K. Talwar, and U. Erlingsson, "Scalable private learning with pate," in *International Conference on Learning Representations*, 2018.

[111] Y. Zhu, X. Yu, M. Chandraker, and Y.-X. Wang, "Private-knn: Practical differential privacy for computer vision," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 854–11 862.

[112] L. Sun, Y. Zhou, P. S. Yu, and C. Xiong, "Differentially private deep learning with smooth sensitivity," *arXiv preprint arXiv:2003.00505*, 2020.

[113] T. Ha, T. K. Dang, T. T. Dang, T. A. Truong, and M. T. Nguyen, "Differential privacy in deep learning: an overview," in *2019 International Conference on Advanced Computing and Applications (ACOMP)*. IEEE, 2019, pp. 97–102.

[114] A. Blanco-Justicia, D. Sánchez, J. Domingo-Ferrer, and K. Muralidhar, "A critical review on the use (and misuse) of differential privacy in machine learning," *ACM Computing Surveys*, vol. 55, no. 8, pp. 1–16, 2022.

[115] A. El Ouadrhiri and A. Abdelhadi, "Differential privacy for deep and federated learning: A survey," *IEEE Access*, vol. 10, pp. 22 359–22 380, 2022.

[116] N. Ponomareva, H. Hazimeh, A. Kurakin, Z. Xu, C. Denison, H. B. McMahan, S. Vassilvitskii, S. Chien, and A. G. Thakurta, "How to dp-fy ml: A practical guide to machine learning with differential privacy," *Journal of Artificial Intelligence Research*, vol. 77, pp. 1113–1201, 2023.

[117] K. Pan, Y.-S. Ong, M. Gong, H. Li, A. Qin, and Y. Gao, "Differential privacy in deep learning: A literature survey," *Neurocomputing*, p. 127663, 2024.

[118] Z. Zheng, L. Yao, H. Hu, and G. Wu, "Benchmarking relaxed differential privacy in private learning: A comparative survey," *ACM Computing Surveys*, 2025.

[119] L. Demelius, R. Kern, and A. Trügler, "Recent advances of differential privacy in centralized deep learning: A systematic survey," *ACM Computing Surveys*, vol. 57, no. 6, pp. 1–28, 2025.

[120] S. H. Haji and A. M. Abdulazeez, "Comparison of optimization techniques based on gradient descent algorithm: A review," *PalArch's Journal of Archaeology of Egypt/Egyptology*, vol. 18, no. 4, pp. 2715–2743, 2021.

[121] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *Journal of Big data*, vol. 3, no. 1, pp. 1–40, 2016.

[122] R. Tatman, J. VanderPlas, and S. Dane, "A practical taxonomy of reproducibility for machine learning research," 2018.

[123] E. Raff, "A step toward quantifying independently reproducible machine learning research," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[124] S. Asoodeh, J. Liao, F. P. Calmon, O. Kosut, and L. Sankar, "Three variants of differential privacy: Lossless conversion and applications," *IEEE Journal on Selected Areas in Information Theory*, vol. 2, no. 1, pp. 208–222, 2021.

[125] S. P. Kasiviswanathan and A. Smith, "On the'semantics' of differential privacy: A bayesian formulation," *Journal of Privacy and Confidentiality*, vol. 6, no. 1, 2014.

[126] D. Desfontaines and B. Pejó, "Sok: differential privacies," *Proceedings on privacy enhancing technologies*, vol. 2020, no. 2, pp. 288–313, 2020.

[127] I. Mironov, "Rényi differential privacy," in *2017 IEEE 30th computer security foundations symposium (CSF)*. IEEE, 2017, pp. 263–275.

[128] A. Rényi, "On measures of entropy and information," in *Proceedings of the fourth Berkeley symposium on mathematical statistics and probability*, vol. 1, no. 547-561. Berkeley, California, USA, 1961.

[129] Y.-X. Wang, B. Balle, and S. P. Kasiviswanathan, "Subsampled rényi differential privacy and analytical moments accountant," in *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, 2019, pp. 1226–1235.

[130] I. Mironov, K. Talwar, and L. Zhang, "R\'enyi differential privacy of the sampled gaussian mechanism," *arXiv preprint arXiv:1908.10530*, 2019.

[131] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate, "Differentially private empirical risk minimization." *Journal of Machine Learning Research*, vol. 12, no. 3, 2011.

[132] N. Papernot, S. Chien, S. Song, A. Thakurta, and U. Erlingsson, "Making the shoe fit: Architectures, initializations, and tuning for learning with privacy," 2020.

[133] F. Morsbach, T. Dehling, and A. Sunyaev, "Architecture matters: Investigating the influence of differential privacy on neural network design," in *NeurIPS 2021 Workshop Privacy in Machine Learning*, 2021.

[134] A. Kurakin, S. Chien, S. Song, R. Geambasu, A. Terzis, and A. Thakurta, "Toward training at imagenet scale with differential privacy," *arXiv preprint arXiv:2201.12328*, 2022.

[135] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[136] C. Dwork, K. Talwar, A. Thakurta, and L. Zhang, "Analyze gauss: optimal bounds for privacy-preserving principal component analysis," in *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, 2014, pp. 11–20.

[137] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *International Conference on Learning Representations*, 2018.

[138] I. Goodfellow, "Efficient per-example gradient computations," *arXiv preprint arXiv:1510.01799*, 2015.

[139] W. Shi, S. Chen, C. Zhang, R. Jia, and Z. Yu, "Just fine-tune twice: Selective differential privacy for large language models," *arXiv preprint arXiv:2204.07667*, 2022.

[140] G. Kerrigan, D. Slack, and J. Tuyls, "Differentially private language models benefit from public pre-training," in *Proceedings of the Second Workshop on Privacy in NLP*, 2020, pp. 39–45.

[141] Y. You, J. Li, S. Reddi, J. Hseu, S. Kumar, S. Bhojanapalli, X. Song, J. Demmel, K. Keutzer, and C.-J. Hsieh, "Large batch optimization for deep learning: Training bert in 76 minutes," in *International Conference on Learning Representations*, 2019.

[142] M. Baradad Jurjo, J. Wulff, T. Wang, P. Isola, and A. Torralba, "Learning to see by looking at noise," *Advances in Neural Information Processing Systems*, vol. 34, pp. 2556–2569, 2021.

[143] M. Baradad, R. Chen, J. Wulff, T. Wang, R. Feris, A. Torralba, and P. Isola, "Procedural image programs for representation learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 6450–6462, 2022.

[144] Y. Yu, M. Sanjabi, Y. Ma, K. Chaudhuri, and C. Guo, "Vip: A differentially private foundation model for computer vision," *arXiv preprint arXiv:2306.08842*, 2023.

[145] T. Sander, Y. Yu, M. Sanjabi, A. O. Durmus, Y. Ma, K. Chaudhuri, and C. Guo, "Differentially private representation learning via image captioning," in *Forty-first International Conference on Machine Learning*, 2024. [Online]. Available: https://openreview.net/forum?id=Nw7yOe8nBi

[146] P. Foret, A. Kleiner, H. Mobahi, and B. Neyshabur, "Sharpness-aware minimization for efficiently improving generalization," *arXiv preprint arXiv:2010.01412*, 2020.

[147] A. Nedić, "Random projection algorithms for convex set intersection problems," in *49th IEEE Conference on Decision and Control (CDC)*. IEEE, 2010, pp. 7655–7660.

[148] S. Lee and A. Nedic, "Distributed random projection algorithm for convex optimization," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 2, pp. 221–229, 2013.

[149] R.-A. Pitaval, W. Dai, and O. Tirkkonen, "Convergence of gradient descent for low-rank matrix approximation," *IEEE Transactions on Information Theory*, vol. 61, no. 8, pp. 4451–4457, 2015.

[150] T. Ye and S. S. Du, "Global convergence of gradient descent for asymmetric low-rank matrix factorization," *Advances in Neural Information Processing Systems*, vol. 34, pp. 1429–1439, 2021.

[151] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex fourier series," *Mathematics of computation*, vol. 19, no. 90, pp. 297–301, 1965.

[152] V. Feldman, I. Mironov, K. Talwar, and A. Thakurta, "Privacy amplification by iteration," in *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2018, pp. 521–532.

[153] Y. Hu, F. Wu, R. Xian, Y. Liu, L. Zakynthinou, P. Kamath, C. Zhang, and D. Forsyth, "Empirical privacy variance," in *Forty-second International Conference on Machine Learning*, 2025. [Online]. Available: https://openreview.net/forum?id=oEvbe7vtOm

[154] A. Panda, X. Tang, C. A. Choquette-Choo, M. Nasr, and P. Mittal, "Privacy auditing of large language models," in *The Thirteenth International Conference on Learning Representations*, 2025. [Online]. Available: https://openreview.net/forum?id=60Vd7QOXlM

[155] B. Marek, V. Hanke, X. Wang, M. Backes, A. Dziedzic, and F. Boenisch, "Auditing empirical privacy protection of private LLM adaptations," in *Neurips Safe Generative AI Workshop 2024*, 2024. [Online]. Available: https://openreview.net/forum?id=vFIC8Le5UH

[156] S. Chard, B. Johnson, and D. Lewis, "Auditing large language models for privacy compliance with specially crafted prompts," *OSF Preprint*, 2024.

[157] V. Rathod, S. Nabavirazavi, S. Zad, and S. S. Iyengar, "Privacy and security challenges in large language models," in *2025 IEEE 15th Annual Computing and Communication Workshop and Conference (CCWC)*. IEEE, 2025, pp. 00 746–00 752.

[158] Y. Yao, J. Duan, K. Xu, Y. Cai, Z. Sun, and Y. Zhang, "A survey on large language model (llm) security and privacy: The good, the bad, and the ugly," *High-Confidence Computing*, vol. 4, no. 2, p. 100211, 2024.

[159] Q. Li, J. Hong, C. Xie, J. Tan, R. Xin, J. Hou, X. Yin, Z. Wang, D. Hendrycks, Z. Wang, B. Li, B. He, and D. Song, "Llm-pbe: Assessing data privacy in large language models," *arXiv preprint arXiv:2408.12787*, 2024.

[160] M. Duan, A. Suri, N. Mireshghallah, S. Min, W. Shi, L. Zettlemoyer, Y. Tsvetkov, Y. Choi, D. Evans, and H. Hajishirzi, "Do membership inference attacks work on large language models?" *arXiv preprint arXiv:2402.07841*, 2024.

[161] M. Meeus, I. Shilov, S. Jain, M. Faysse, M. Rei, and Y.-A. de Montjoye, "Sok: Membership inference attacks on llms are rushing nowhere (and how to fix it)," in *2025 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*. IEEE, 2025, pp. 385–401.

[162] D. Wang, C. Chen, and J. Xu, "Differentially private empirical risk minimization with non-convex loss functions," in *International Conference on Machine Learning*. PMLR, 2019, pp. 6526–6535.

[163] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization." *Journal of machine learning research*, vol. 12, no. 7, 2011.

[164] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[165] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural networks for machine learning*, vol. 4, no. 2, pp. 26–31, 2012.

[166] M. Bun and T. Steinke, "Concentrated differential privacy: Simplifications, extensions, and lower bounds," in *Theory of Cryptography Conference*. Springer, 2016, pp. 635–658.

[167] "Opacus," https://github.com/pytorch/opacus, 2023.

[168] D. Lakens, "Calculating and reporting effect sizes to facilitate cumulative science: a practical primer for t-tests and anovas," *Frontiers in psychology*, vol. 4, p. 62627, 2013.

[169] T. K. Kim, "T test as a parametric statistic," *Korean journal of anesthesiology*, vol. 68, no. 6, p. 540, 2015.

[170] R. Wetzels, D. Matzke, M. D. Lee, J. N. Rouder, G. J. Iverson, and E.-J. Wagenmakers, "Statistical evidence in experimental psychology: An empirical comparison using 855 t tests," *Perspectives on Psychological Science*, vol. 6, no. 3, pp. 291–298, 2011.

[171] P. C. Austin, "A critical appraisal of propensity-score matching in the medical literature between 1996 and 2003," *Statistics in medicine*, vol. 27, no. 12, pp. 2037–2049, 2008.

[172] M. L. Head, L. Holman, R. Lanfear, A. T. Kahn, and M. D. Jennions, "The extent and consequences of p-hacking in science," *PLoS biology*, vol. 13, no. 3, p. e1002106, 2015.

[173] Y. Grandvalet and Y. Bengio, "Hypothesis testing for cross-validation," *Montreal Universite de Montreal, Operationnelle DdIeR*, vol. 1285, 2006.

[174] P. Bayle, A. Bayle, L. Janson, and L. Mackey, "Cross-validation confidence intervals for test error," *Advances in Neural Information Processing Systems*, vol. 33, pp. 16 339–16 350, 2020.

[175] K. L. Lange, R. J. Little, and J. M. Taylor, "Robust statistical modeling using the t distribution," *Journal of the American Statistical Association*, vol. 84, no. 408, pp. 881–896, 1989.

[176] S. Goodman, "A dirty dozen: twelve p-value misconceptions," in *Seminars in hematology*, vol. 45, no. 3. Elsevier, 2008, pp. 135–140.

[177] G. Van Belle, *Statistical rules of thumb*. John Wiley & Sons, 2011.

[178] D. Picard, "Torch. manual_seed (3407) is all you need: On the influence of random seeds in deep learning architectures for computer vision," *arXiv preprint arXiv:2109.08203*, 2021.

[179] J. Gardner, Y. Yang, R. Baker, and C. Brooks, "Enabling end-to-end machine learning replicability: A case study in educational data mining," *arXiv preprint arXiv:1806.05208*, 2018.

[180] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, "Deep reinforcement learning that matters," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[181] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[182] Y. Ding and X. Wu, "Revisiting hyperparameter tuning with differential privacy," *arXiv preprint arXiv:2211.01852*, 2022.

[183] A. Koskela and T. D. Kulkarni, "Practical differentially private hyperparameter tuning with subsampling," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[184] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," 2009.

[185] Y. LeCun, "The mnist database of handwritten digits," *http://yann.lecun. com/exdb/mnist/*, 1998.

[186] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[187] P. Helber, B. Bischke, A. Dengel, and D. Borth, "Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 12, no. 7, pp. 2217–2226, 2019.

[188] J. Yang, R. Shi, and B. Ni, "Medmnist classification decathlon: A lightweight automl benchmark for medical image analysis," in *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*. IEEE, 2021, pp. 191–195.

[189] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," 2007.

[190] J. Xiao, K. A. Ehinger, J. Hays, A. Torralba, and A. Oliva, "Sun database: Exploring a large collection of scene categories," *International Journal of Computer Vision*, vol. 119, pp. 3–22, 2016.

[191] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "Sun database: Large-scale scene recognition from abbey to zoo," in *2010 IEEE computer society conference on computer vision and pattern recognition*. IEEE, 2010, pp. 3485–3492.

[192] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. Jawahar, "Cats and dogs," in *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 2012, pp. 3498–3505.

[193] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *2017 IEEE symposium on security and privacy (SP)*. IEEE, 2017, pp. 3–18.

[194] J. Ye, A. Maddi, S. K. Murakonda, V. Bindschaedler, and R. Shokri, "Enhanced membership inference attacks against machine learning models," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, 2022, pp. 3093–3106.

[195] N. Carlini, S. Chien, M. Nasr, S. Song, A. Terzis, and F. Tramer, "Membership inference attacks from first principles," in *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2022, pp. 1897–1914.

[196] T. Steinke, M. Nasr, and M. Jagielski, "Privacy auditing with one (1) training run," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

Appendix A. Datasets

CIFAR-10. We use the CIFAR-10 dataset [184], which contains 60,000 images with 10 classes. We use 50,000 as the training set and 10,000 as the test set as following most papers do. Each image in CIFAR-10 has 3 RGB channels and its size is 32×32 pixels.

MNIST. It [185] contains 70,000 28×28 gray scale handwritten digit images. We use 60,000 for training and 10,000 for testing.

Fashion-MNIST. It [186] contains 70,000 28×28 grayscale images of clothing. We use 60,000 for training and 10,000 for testing.

EuroSAT. This dataset [187] contains Sentinel-2 satellite images with 10 classes. It has 27,000 64×64 labeled color images. We use 21600 as the training set and 5400 as a test set.

ISIC 2018. We use task 3 of this 2018 year's dataset [18] published by the International Skin Imaging Collaboration (ISIC) for lesion classification challenges. It contains 10,015 images which we use 9,015 images for training and 1,000 for testing.

PathMNIST. This dataset is part of MedMNIST[188]. It contains 107,180 RGB images with 9 classes. The image size is 28×28. We use 89,996 images as the training set and 7180 as the test set.

Caltech 256. The Caltech 256 dataset [189] is frequently utilized for image classification tasks, consisting of 30,607 RGB images across 257 diverse object categories. In our experiments, we allocated 80% of these images for training purposes and the remaining 20% for evaluation.

SUN397. The Scene UNderstanding (SUN) [190], [191] dataset comprises 108,754 RGB images spanning 397 distinct classes. For our experimental framework, 80% of these images were used for training, with the balance of 20% reserved for testing.

Oxford Pet. This dataset [192] features 37 categories of cats and dogs, totaling 7,349 images. We used 3,680 images for the training set and 3,669 for the test set.

18. https://challenge.isic-archive.com

Appendix B. Additional Experiments

B.1. Empirical Privacy Measurements

TABLE 23: Membership Inference Attacks AUC for different methods for varying privacy budgets from $\varepsilon = 0.1$ to $\varepsilon = 8$.

	ε	P-Attack	S-Attack	R-Attack	C-Attack
Basline	0.1	0.50	0.50	0.49	0.50
	0.5	0.50	0.50	0.48	0.50
	1	0.49	0.50	0.48	0.50
	8	0.50	0.50	0.48	0.50
De et al. [17]	0.1	0.50	0.50	0.50	0.50
	0.5	0.49	0.49	0.48	0.49
	1	0.50	0.50	0.48	0.50
	8	0.50	0.50	0.49	0.50
Bao et al. [18]	0.1	0.50	0.50	0.50	0.50
	0.5	0.50	0.50	0.49	0.50
	1	0.50	0.50	0.48	0.50
	8	0.50	0.51	0.49	0.50

Since DP is a worst-case notion, different methods providing the same DP guarantee could provide different empirical privacy, as measured by membership inference attacks [193], [194], [195]. We use the popular Privacy Meter tool to run four different attacks.[19] The attacks are P-Attack (Population), R-Attack (Reference), S-Attack (Shadow Models) based on [194], and C-Attack (Carlini et al. [195]). We report the Area Under the Curve (AUC) as a measure of the attack success rate.

We consider three methodologies: a baseline approach which involves training a WRN-16-4 network with vanilla-DP-SGD on CIFAR-10 from scratch, utilizing technique from De et al. [17], and another leveraging method based on Bao et al. [18]. Membership inference attacks require held-out data points from the training set, thus we limit the training dataset to 30,000 samples. The privacy budgets tested range from $\varepsilon = 0.1$ to $\varepsilon = 8$. Results are shown in Table 23.

Despite the diversity in techniques, all methods maintain analogous levels of empirical privacy, achieving AUC close to 50%—comparable to random guessing, even when using a relatively high privacy budget (i.e., $\varepsilon = 8$).

Empirical privacy and accuracy trade-off. Results from Tables 13 and 23 suggest that even loose privacy guarantees (e.g., $\varepsilon = 8$) may offer meaningful protection. However, we caution that the attacks we perform are all based on the black-box setting, and that the conclusions may not hold with stronger attacks or in the white-box setting. We refer readers to the (growing) literature on privacy auditing (e.g.,[196], [91]) for a more nuanced discussion.

Appendix C. Checklist

In this section, we describe the rationale behind the items in our proposed checklist alongside examples.

19. https://github.com/privacytrustlab/ml_privacy_meter

C.1. Generalizability

Evaluation in different settings. If the proposed method can be used in multiple settings (e.g., train from scratch and pre-trained), it should be evaluated in different settings.

- *Rationale*: A method could provide a substantial improvement in one setting but no improvement in another setting.
- *Examples*: De et al. [17] advocate for self-augmentations, showcasing notable performance in both from-scratch training and fine-tuning, a finding corroborated by our reproduction experiments.

Evaluation with different datasets. Methods should be evaluated with multiple different domain datasets to uncover whether improvements persists.

- *Rationale*: Some methods may only provide improvements for some datasets because different datasets involve tasks of varying difficulty. For example, a method may work only on specific datasets, or datasets with few classes or few input features. Moreover, when considering pre-trained models, a method may appear to work particularly well because the pre-trained model was trained on data similar to the target dataset.
- *Examples*: Some papers only report results for few datasets like CIFAR-10 and CIFAR-100. When applying their methods to datasets from a different domain such as EuroSAT, we did not observe their claimed improvements.

Evaluation with different architectures. Methods should be evaluated with different model architectures, if applicable.

- *Rationale*: Some methods may only provide improvements with specific model architectures because of the nature of the method or task. Moreover, performance on a task can greatly differ from one architecture to another. For example, a method may perform well using a model architecture with relatively few parameters because that architecture may be well-suited for the considered tasks. For different tasks, however, the method may falter if such tasks require models with much larger parameter counts.
- *Examples*: Some papers only evaluate their methods on a particular model such as WRN-28-10 model. When applying their method to VIT model, we are not able to reproduce their claimed performance.

Evaluation for different privacy requirements. Methods should be evaluated in different privacy regimes, that is with different range of values for ε (and δ if applicable). The privacy parameters range considered should be appropriate for the given setting. For example, pre-trained models fine-tuned with large datasets may tolerate much lower ε values than models trained from scratch on small datasets.

- *Rationale*: Improvements provided from a method may not be uniform across all privacy regimes. Typically the less stringent the privacy requirement the less improvement there is, in part because DPML performance is

closer to the non-private setting than for more stringent requirements.
- *Examples*: Tramèr and Boneh [19] present their method's results for low privacy budgets, specifically values smaller than 3. In our reproduction of their experiments with a larger privacy budget, such as 8, we observed that the performance of their proposed method plateaued as the privacy budget increased.

Evaluation of combination with other techniques. Authors should evaluate or discuss whether their proposed methods can be combined with other methods.

- *Rationale*: Some combinations of seemingly orthogonal methods actually decrease performance.
- *Examples*: Combining the methods of De et al. [17] with Sander et al. [15] yields improved performance. However, merging Klause et al. [14] with Sander et al. [15] leads to a decrease in performance.

C.2. Reliability

Code open sourcing Authors should open-source their code to facilitate reproducibility.

- *Rationale*: Differences in the implementation of the same technique or the use of different codebases can yield significant differences. Open-sourcing code is a straightforward way to mitigate such concerns.

Results of multiple runs are reported. Our experiments on the randomness of DPML show that the variability of DPML is significant (Section 7.5). Reporting the result of multiple runs is a way to mitigate this problem.

- *Rationale*: Providing both an aggregate measure and a measure of variability across runs (e.g., mean and std) facilitates scientifically valid comparisons. The number of runs performed should be appropriate for the setting and privacy budget.
- *Examples*: Some papers do not present results from multiple runs, making it challenging to discern if the performance improvement is due to their proposed method or due to randomness.

Improvement is statistically significant The proposed techniques should show statistically significant improvement beyond baselines using our proposed framework.

- *Rationale*: If the performance of two methods being compared is similar, then we cannot conclusively determine which method (if any) is superior.
- *Examples*: The performance boost from some papers is not statistically significant.

Account for hyperparameter search. The cost and benefit of hyperparameter search need to be taken into account.

- *Rationale*: To avoid unfair comparisons any hyperparameter search must be accounted for. Ideally, separate validation and test sets should be used and the privacy cost of the search should be reported.
- *Examples*: None of our selected papers pay the privacy budget to hyperparameter search. We also find that only De et al.[17] and Tang et al. [24] have validation set for hyperparameter tuning.

Evaluation includes ablation experiments. Authors should ensure that improvements observed can be attributed to the proposed methods, e.g., through the use of an ablation study or experimental methodology to exclude other factors.

- *Rationale*: Some papers combine multiple methods without independent evaluation, or otherwise evaluate their approach in a way that measured improvements cannot be conclusively tied back to the proposed technique.
- *Examples*: Some papers do not provide a comprehensive ablation study. For example, when using a more complex pre-trained model than prior work to achieve new SoTA results, observed improvements could be due to a pre-trained model, the proposed method, or both.