R+R: Towards Reliable and Generalizable Differentially Private Machine Learning

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 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 numerous 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 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

*. We open source our codebase at: https://github.com/wenxuan-Bao/Reliable-and-Generalizable-DPML.

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:

- 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

We provide background related to supervised learning, ML reproducibility, and DPML. Appendix A provides additional background.

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 [29]. 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 (Appendix C). 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. 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. [30] suggest a taxonomy for reproducibility with three levels. Raff et al. [31] 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.

2.3. Differential Privacy

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 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.

2.4. Differentially Private Machine Learning

There are numerous randomized learning algorithms that meet differential privacy such as output perturbation [10] or objective perturbation [32]. However, the most widely used technique is the Differentially Private Stochastic Gradient Descent (DP-SGD) of Abadi et al. [13].

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

TABLE 1: 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 15.

Techniques	Paper	Year	Reproducibility	
			G	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	●	●

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 (Appendix C). 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. Selected papers

3.1. 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. Due to space constraints we do not provide here a comprehensive overview of the DPML literature. We refer interested readers to recent surveys [33], [34], [35], [36], [37].

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 1. 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.

3.2. Selected Papers

Here is a brief overview of the selected papers.

Tuning existing architectures. Klause et al. [14] proposed an architecture modification on ResNet, adding a normalization layer after the residual block called ScaleNorm. They claim that ScaleNorm can improve the speed of convergence, and they 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, they find that using normalization before ReLU leads to improved performance compared to using ReLU before batch normalization.

Hyperparameter Tuning. 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 hyperparameters tuning method using a large batch size and high noise multiplier to achieve SoTA performance.

Data Augmentation: Augmentation multiplicity. 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.

Data Augmentation: Mixup. Bao et al. [18] introduced two techniques, DP-MIX$_{\text{SELF}}$ and DP-MIX$_{\text{DIFF}}$, to integrate mixup [38] into DP-SGD. DP-MIX$_{\text{SELF}}$ is similar to De et al. [17] but employs mixup as an augmentation, while DP-MIX$_{\text{DIFF}}$ utilizes a text-to-image diffusion model to create class-specific synthetic examples that are then mixed up with actual training data. Both achieve SoTA performance across various datasets and settings.

Feature selection. 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.

Clipping: Auto-Clipping. Bu et al. [21] and Yang et al. [39] 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. They add a parameter γ which can be relatively stable from case to case to preserve the gradients' magnitude information. They show the convergence of DP-SGD with Auto clipping is the same as SGD. In experiments, they show that DP-SGD with Auto clipping has a slightly better performance than DP-SGD with basic clipping. They claim to match or outperform SoTA.

Clipping: Mixed Ghost Clipping. By combining and extending previous works [40], [41], [42], Bu et al. [20] propose a technique called Mixed Ghost Gradient Clipping which does not require computing per-sample gradients. They use ghost norm to compute gradient norm without actually computing gradients and second back-propagation to transfer weighted loss to weighted 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.

Fine-tuning: Selective Fine-tuning. Luo et al. [23] propose a new fine-tuning technique targeted at the normalization layer and a percentage of parameters in convolution layers that have high magnitudes. 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.

Fine-tuning: Random Process Pre-training. Tang et al. [24] introduce a method to pre-train models on random process data [43], [44], 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 SoTA results, underlining its effectiveness.

4. Methodology

In this section, we describe 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).

4.1. 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?

4.2. 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 B.

Setup. To ensure a fair comparison, all our experiments are conducted using PyTorch and Opacus [45]. 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 [45] 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.

4.3. 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 [46] to measure *effect size* (improvement size) and paired t-tests [47] to measure statistical significance. These methods are well regarded and used across diverse scientific research areas, including psychology [48] and medicine [49].

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 [50]. Such tests are seldom used in machine learning practice, although there are numerous methods to use them [51], [52]. 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 4.4). This is especially striking when the privacy budget is small (e.g., $\varepsilon = 0.1$) or the training dataset is small.[2] 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. 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 [53], the *t-statistic* is calculated as: $t = \frac{\mu_d}{\sigma_d / \sqrt{n}}$, and from it a *p-value* is obtained [54]. 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 the result of chance. In our experiments, we use the default parameters of scipy.stats.ttest_rel to compute the *t-statistic* and p-value.[3]

We also propose to 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.[4] 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 [55]). 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.

4.4. 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 [56] who studied 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 2.

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 2) 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

[2] If the training dataset is huge, as is often the case in deep learning research, statistical measurements are 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.

[3] https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.ttest_rel.html

[4] The paired t-test's statistical power increases with the correlation.

TABLE 2: 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%

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 4.5.

Another relevant observation from Table 2 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.

4.5. Seed Hacking?

In this section, we explore the concept of *seed hacking*, inspired by the work of Picard [56]. 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 [57], [58]. 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 [50] but also to further drive home the point that the increased randomness in DPML matters.

We use data from our random seed experiment (Table 2) 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.

TABLE 3: 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

Results are shown in Table 3, 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.

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.

TABLE 4: 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%)

TABLE 5: Sander et al. [15] performance for varying ε.

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

5. 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 4).[5] 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.

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%.[6] 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 6).

We empirically explored four different ordering schemes as represented in their code.[7] Results are shown in Table 6. 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 5. 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].[8]

For reproducibility, the results we obtain are shown in Table 6. 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 8. 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.[9] The reproducible results of DP-MIX$_{\text{SELF}}$ we obtain are shown in Table 6. 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 7. 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 8. We can observe that the performance boost from their proposed method decreases significantly as the privacy budget decreases, similar to the method proposed by De et al.[17].

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.*

[5] https://github.com/OsvaldFrisk/dp-not-all-noise-is-equal

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

[7] 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

[8] https://github.com/facebookresearch/tan

[9] https://github.com/wenxuan-Bao/DP-Mix

TABLE 6: 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 7: 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)

TABLE 8: 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%)

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 6). 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.[10] We find that the method only slightly improved performance (Table 6).

TABLE 9: 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 Mixed Ghost clipping	94.71%(±0.14%) **94.77%**(±0.13%)	0.78	0.52	0.44
Vit_base_patch16_224	Basic clipping Mixed Ghost clipping	**97.34%**(±0.20%) 95.05%(±0.13%)	-16.61	**0.004**	-13.57

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[11], and fine-tuned the models on CIFAR-10. We compared Mixed Ghost clipping and basic gradient clipping, with $\varepsilon = 1$ and two training epochs, as suggested in the paper (Table 9). Mixed Ghost Clipping *did not* outperform basic gradient clipping in a statistically significant way (Section 4.3). 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 10 shows our results, which are

[10] https://github.com/woodyx218/private_vision

[11] https://github.com/huggingface/pytorch-image-models

TABLE 10: Reproduced results for Bu et al. [21]. According to the framework of Section 4.3, 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

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.

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 14: 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.[12] We then compare the results to De et al. [17] under varying privacy budgets (Table 11). 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 performance plateauing for Tramèr and Boneh whereas the test accuracy for De et al. keeps increasing.

TABLE 11: 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

TABLE 12: 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 13: 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

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 12.[13]

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) [59] as a measure of the domain gap.

As shown in Table 12, the performance obtained when pre-training on EuroSAT is analogous to that achieved

[12] https://github.com/ftramer/Handcrafted-DP

[13] https://github.com/inspire-group/DP-RandP

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 13. 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.

6. 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 A.

6.1. 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 14. 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 14) 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.*

6.2. 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 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.*

6.3. 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 16. 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 16 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 4.3, a large enough effect size can overcome

TABLE 14: 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%)

TABLE 15: 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. The average running time per epoch and its standard deviation are reported. 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.

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[14]	×	0.44	1.76	1.20 [15]
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 [16]
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 [17]
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 [18]
Tang et al. [24]	Fine-tuning technique	N/A	✓	✓	✓	✓	✓	✓	✓	46.76	93.16	10.92

TABLE 16: 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

a small number of runs. Power analysis on the results of Table 16 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 7).

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], [60], 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[61], [62], [63], 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 [64], [65], [66] 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, both De et al. and Bao et al. 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.

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

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

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

[16] We use this method as the DP-SGD baseline as it is well-known.

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

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

TABLE 17: 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	□

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.

6.4. 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 15.

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.

7. 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 17.

Selected papers. We grade our 11 selected papers according to our checklist. Results are shown in Table 15. 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 D.

8. 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-205512 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, A. Roth *et al.*, “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] 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.

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

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

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

[33] 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.

[34] 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.

[35] 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.

[36] 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.

[37] 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.

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

[39] 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.

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

[41] 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.

[42] 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.

[43] 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.

[44] 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.

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

[46] 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.

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

[48] 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.

[49] 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.

[50] 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.

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

[52] 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.

[53] 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.

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

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

[56] 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.

[57] 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.

[58] 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.

[59] 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.

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

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

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

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

[64] 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.

[65] 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.

[66] 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.

[67] 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.

[68] 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.

[69] 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.

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

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

[72] A. Rényi *et al.*, "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.

[73] 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.

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

[75] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

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

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

[78] 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.

[79] 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.

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

[81] 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.

[82] 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.

[83] 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.

[84] 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.

[85] 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.

[86] 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.

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

[88] 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.

Appendix

Appendix A. Additional Background

A.1. 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. [67] 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 some classification layers), and then run the training process on the new task to fine-tune the existing pre-trained layers alongside with the new layers.

A.2. Differential Privacy and Variants

Rényi DP (RDP). There are several variants (both generalization and relaxations) of differential privacy [68], [69], [70] such as approximate DP [10], Rényi DP (RDP) [71]. Among those DP variants, Rényi DP is the most widely used. It is based on Rényi divergence [72] and is defined as follows (Mironov [71]).

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 [73], [74] and its composition properties. We refer the reader to Mironov [71] 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 $\delta \in (0, 1)$.*

A.3. Learning Algorithms

For completeness, we provide a description of SGD (Algorithm 1) and DP-SGD (Algorithm 2).

Algorithm 1 SGD

```
Input: Training dataset $D$, loss function $\mathcal{L}(\theta)$. Parameters: learning rate $\eta_t$, mini-batch size $L$. $N = |D|$ is the number of training data points.
  Initialize $\theta_0$ 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: $\theta_T$
```

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 $\eta_t$, noise scale $\sigma$, mini-batch/lot size $L$, gradient norm bound $C$.
  Initialize $\theta_0$ 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: $\theta_T$ and and compute the overall privacy cost $(\varepsilon, \delta)$ using a privacy accounting method
```

Appendix B. Datasets

CIFAR-10. We use the CIFAR-10 dataset [75], 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 [76] contains 70,000 28 × 28 gray scale handwritten digit images. We use 60,000 for training and 10,000 for testing.

Fashion-MNIST. It [77] contains 70,000 28 × 28 grayscale images of clothing. We use 60,000 for training and 10,000 for testing.

EuroSAT. This dataset [78] 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 [19] 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[79]. 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. [80]. The Caltech 256 dataset is frequently utilized for image classification tasks, consisting of 30,607 RGB images across 257 diverse object categories. In our experiments, we allocated 80

SUN397. The Scene UNderstanding (SUN) [81], [82] dataset comprises 108,754 RGB images spanning 397 distinct classes. For our experimental framework, 80

Oxford Pet. [83] 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.

Appendix C. Additional Experiments

C.1. Empirical Privacy Measurements

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 [84], [85], [86]. We use the popular Privacy Meter tool[20] to run four different attacks. The attacks are P-Attack (Population), R-Attack (Reference), S-Attack (Shadow Models) based on [85], and C-Attack (Carlini et al. [86]). We report the Area Under the Curve (AUC) as a measure of the attack success rate.

TABLE 18: 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

We consider three methodologies: a baseline approach which involves training a WRN-16-4 network with vanilla-DP-SGD on CIFAR-10 from scratch[21], utilizing technique from De et al. [17], and another leveraging method based on Bao et al. [18]. The privacy budgets tested range from $\varepsilon = 0.1$ to $\varepsilon = 8$. Results are shown in Table 18.

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 8 and 18 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.,[87], [88]) for a more nuanced discussion.

Appendix D. Checklist

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

D.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.

[19] https://challenge.isic-archive.com

[20] https://github.com/privacytrustlab/ml_privacy_meter

[21] Membership inference attacks require held-out data points from the training set, thus we limit the training dataset to 30,000 samples.

- *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.

D.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 4.4). 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 falls without statistically significant 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. However, 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.