Automatic Clipping: Differentially Private Deep Learning Made Easy and Stronger

Anonymous Author(s)
Affiliation
Address
email

Abstract

Per-example gradient clipping is a key algorithmic step that enables practical differential private (DP) training for deep learning models. The choice of clipping norm R, however, is shown to be vital for achieving high accuracy under DP. We propose an easy-to-use replacement, called AutoClipping, that eliminates the need to tune R for any DP optimizers, including DP-SGD, DP-Adam, DP-LAMB and many others. The automatic variants are as private and computationally efficient as existing DP optimizers, but require no DP-specific hyperparameters and thus make DP training as amenable as the standard non-private training. We give a rigorous convergence analysis of automatic DP-SGD in the non-convex setting, which shows that it enjoys an asymptotic convergence rate that matches the standard SGD. We also demonstrate on various language and vision tasks that automatic clipping outperforms or matches the state-of-the-art, and can be easily employed with minimal changes to existing codebases.

1 Introduction

Deep learning has achieved impressive progress in a wide range of computer vision and natural language processing tasks. These successes are made available, in part, by the collection of large datasets, sometimes containing sensitive private information of individual data points (e.g., chest scan images, DNA sequences). Prior works have illustrated that deep learning models pose severe privacy risks to individual subjects in the training data and are susceptible to various practical attacks. For example, machine learning services such as Google Prediction API and Amazon Machine Learning can leak membership information from the purchase records [58]; if one feeds the GPT2 language model with some specific prefix, the model will autocomplete texts that contain the full name, phone number, email address, etc., from the training data that it memorizes [10].

Differential privacy (DP) [20, 22, 21] is a formal definition of privacy that has been shown to prevent the aforementioned privacy risks in deep learning [1]. On a high level, the key difference between the DP deep learning and the regular one is whether the gradient is privately released. In other words, while the standard optimizers update on the summed gradient $\sum_i \boldsymbol{g}_i$, and DP optimizers update on the *private gradient*:

$$\text{DP Optimizer}(\{\boldsymbol{g}_i\}_{i=1}^B) = \text{Optimizer}(\overbrace{\sum\nolimits_i \boldsymbol{g}_i \cdot \texttt{Clip}(\|\boldsymbol{g}_i\|; R) + \sigma R \cdot \mathcal{N}(0, \mathbf{I})}^{\text{private gradient}}) \tag{1.1}$$

$$\text{Standard Optimizer}(\{\boldsymbol{g}_i\}_{i=1}^B) = \text{Optimizer}(\sum\nolimits_i \boldsymbol{g}_i) \tag{1.2}$$

Here $\boldsymbol{g}_i \in \mathbb{R}^d$ is the per-sample gradient of loss l_i, $\mathcal{N}$ is the standard normal random variable, σ is the noise multiplier, and R is the clipping norm. The clipping function $\texttt{Clip} : \mathbb{R}^d \to \mathbb{R}$ is defined

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

such that $\|\boldsymbol{g}_i \cdot \texttt{Clip}(\boldsymbol{g}_i; R)\| \leq R$. For instance, the DP-SGD in [1] on batch B_t is

$$\text{DP-SGD}_{\text{Abadi}}: \quad \boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \frac{\eta}{|B_t|}\Big(\sum_{i\in B_t} \frac{\partial l_i}{\partial \boldsymbol{w}_t} \min\Big(R/\Big\|\frac{\partial l_i}{\partial \boldsymbol{w}_t}\Big\|, 1\Big) + \sigma R \cdot \mathcal{N}(0, \mathbf{I})\Big) \tag{1.3}$$

In comparison to the regular training (1.2), two additional DP-specific hyperparameters R and σ need to be determined in DP learning (1.1). On the one hand, setting the noise multiplier σ is easy and can be derived analytically prior to the training. Whenever the privacy budget (ϵ, δ) is determined, one can apply off-the-shelf privacy accounting tools in Section 2.1 to determine σ, based on the subsampling probability p (e.g. batch size over sample size) and the number of iterations T:

$$\text{privacy_accountant}(\sigma, p, T; \delta) = \epsilon$$

On the other hand, the choice of clipping norm R is crucial to the performance of DP models, yet the hyperparameter tuning is much labor intensive. Recent advances of DP deep learning on ImageNet [34] and on E2E datasets [36], using ResNet18 and GPT2 respectively, illustrate that the performance is very sensitive to R, as we have reproduced in Figure 1. A closer look at Figure 1 shows that, on ImageNet, ResNet18 can drop from the highest 45% accuracy to 31% if R is chosen 2 times larger, and to 0.1% if R is chosen 4 times larger. Similar drastic drop can also be observed in [34, Figure 3] even if the noise multiplier $\sigma = 0$. Unlike the noise multiplier σ, the clipping norm R cannot be inferred from the privacy budget (ϵ, δ) and have to be tuned with caution, as R affects both the clipping operation and the noise addition in (1.1). Consequently, DP training necessarily requires a 2D grid search for (R, η), like the right plot of Figure 1, whereas the regular training only requires an easy 1D grid search for η. Even worse, the difficulty of tuning a per-layer clipping norm vector [43], i.e. one clipping norm for one layer, may increase exponentially to the number of layers.

To save the effort of tuning R, previous researches have proposed different approaches to efficiently set R. In [3, 51, 24], researchers advocate to use data-adaptive information to select R, such as a specified quantile of the gradient norm distribution. These adaptive clipping methods can be a little ad-hoc: they often replace the the need to tune R by the need to tune one or more new hyperparameters, e.g. the quantile to use and the ratio to split the privacy budget between the quantile decision and the gradient perturbation. Another approach used by the practitioners is to replace an expensive 2D grid search by multiple cheaper 1D grid searches. For example, the researchers propose, in [34, Section 3.3] to fine-tune η with non-DP SGD, fix η and sweep over various values of the clipping norm R with DP-SGD, then further fix R and do one more grid search on η. However, tuning R formally in a data-dependent way (e.g. through cross-validation) introduces additional privacy loss [48], and most existing empirical work does not privately conduct hyperparameter tuning.

We take a completely different route by proposing a new clipping principle that removes R, instead of coming up with methods to find the appropriate R. We term our method as the automatic clipping and summarize our contributions as follows.

1. We propose the automatic clipping in (4.1) that expunge the clipping norm from general DP optimizers, allowing DP learning to be as amenable as regular learning.
2. We show that automatic DP optimizers are as private and efficient as existing DP optimizers.
3. We show that automatic DP-SGD converges in the non-convex setting, at the same asymptotic convergence rate as the standard SGD. Our theoretical analysis successfully explains the training behaviors in previous empirical works.
4. We demonstrate the superiority of automatic clipping on extensive vision and language tasks, especially with large models including ResNet, RoBERTa and GPT2.
5. In Appendix K, we publicize the modification that easily switches from Abadi's clipping to our automatic clipping in popular codebases, e.g. Opacus and Objax.

2 Preliminaries

2.1 Differential Privacy

We consider the (ϵ, δ)-DP in Definition 2.1, where smaller (ϵ, δ) means stronger privacy guarantee.

Definition 2.1 ([21])**.** A randomized algorithm M is (ε, δ)-differentially private (DP) if for any two neighboring[1] datasets S, S', and for any event E,

$$\mathbb{P}[M(S) \in E] \leqslant \mathrm{e}^{\varepsilon}\mathbb{P}\left[M\left(S'\right) \in E\right] + \delta. \tag{2.1}$$

[1] S' is a neighbor of S if one can obtain S' by adding or removing one data point from S.

In words, DP restricts the influence of an arbitrary sample, so that the information contributed by such sample is limited and less vulnerable to privacy attacks. In deep learning, DP is generally achieved by applying the subsampled Gaussian mechanism in Lemma A.1 with "Amplification by Sampling" to privatize the minibatch gradients during training.

As illustrated in Equation (1.1), the subsampled Gaussian mechanism involves (1) Sampling a minibatch by including each data point iid with probability p (2) per-sample gradient clipping to bound the l_2 norm sensitivity at R and (3) adding independent Gaussian noise proportional to the sensitivity R and σ, which is derived from the privacy loss ϵ. This can be realized by leveraging a variety of modern privacy accounting tools, such as those based on Renyi DP (or moments accountant) [1, 45, 63], Privacy Loss distribution (Fourier accountants) [33, 26, 70], or Gaussian DP [15, 7].

2.2 Differentially Private optimizers with general clipping operations

Applying the Gaussian mechanism to deep learning leads to various DP optimizers, including DP-SGD in (1.3), DP-HeavyBall, DP-AdaGrad, DP-Adam, DP-FedAvg, and DP-FedSGD [43]. To improve the performance of DP optimizers, previous researches can be classified into two categories.

The first category, where the majority of researches lie in, works with Abadi's clipping and focuses on better design of R. To name a few examples, one can adaptively design R_t for each iteration t [3, 51, 24], or design the per-layer clipping norm vector $\boldsymbol{R} \in \mathbb{R}^L$ for L layers [1, 43] so as to apply different clipping norm for each layer.

Much fewer works fall into the second category that proposes new clipping method. In fact, any function $\texttt{Clip} : \mathbb{R}^d \to \mathbb{R}$ satisfying $\|\texttt{Clip}(\boldsymbol{g}) \cdot \boldsymbol{g}\| \leq R$ can serve as a valid clipping function besides Abadi's. For instance, the global clipping [9] proposes $\texttt{Clip}_{\text{global}}(\boldsymbol{g}) := \mathbb{I}(\|\boldsymbol{g}\| < Z) \cdot R/Z$ to correct the bias in the private gradient and alleviate the mis-calibration issue of DP classifiers. Our automatic clipping also belongs to this category. We note that different clipping methods work orthogonally to optimizers, network architectures and gradient norm computation (see Section 7).

3 Motivation

3.1 Small clipping norm works best

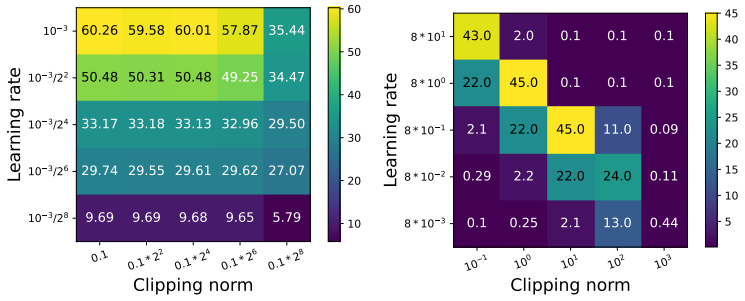

Figure 1: Ablation study of clipping norm and learning rate that achieves SOTA results. Left: GPT2 on E2E dataset, adapted from [36], trained with DP-AdamW. Right: ResNet18 on ImageNet dataset, adapted from [34], trained with DP-Heavyball (SGD with momentum).

Recent studies have witnessed that DP optimizers (1.1) achieve state-of-the-art (SOTA) results with small clipping norms R, in both computer vision and language tasks. In [36], GPT2 (over 800 million parameters) and RoBERTa models (over 400 millions parameters) achieve the best results under DP on QNLI, MNLI, SST-2, QQP, E2E, and DART datasets, with each per-sample gradient clipped to length $R = 0.1$. In [34, 13, 44], ResNets and Vision Transformers achieve the best DP results on ImageNet with $R = 1$; in [60], the best DP results on CIFAR10 use $R = 0.1$ with ResNeXt-29 and SimCLRv2 [11]. The effectiveness of small clipping norm together with proper learning rate is depicted in Figure 1.

Intuitively, smaller R implies that the Abadi's clipping (3.1) happens, which means $\min\big(R/\|\boldsymbol{g}_i\|, 1\big) = R/\|\boldsymbol{g}_i\|$. Given that the clipping norm R is so small compared to the number of parameters in large neural networks, and that strong DP is guaranteed when the number of training iterations is small (i.e. $\|\boldsymbol{g}_i\|$ has not converged to small values yet), we expect and empirically observe that the clipping happens on a large proportion of per-sample gradients at all iterations. For

instance, we find in the GPT2 generation experiments in [36] that 100% of per-sample gradients are clipped at all iterations; in classification tasks such as QQP/QNLI/MNLI, the percentage of clipping is about $20 \sim 60\%$ on average by Appendix H.1.

3.2 Per-sample gradient normalization as new clipping: a similarity viewpoint

In the small clipping norm regime, we can approximately view

$$\texttt{Clip}_{\text{Abadi}}(\boldsymbol{g}_i; R) = \min\left(R/||\boldsymbol{g}_i||, 1\right) \approx R/||\boldsymbol{g}_i|| =: \texttt{Clip}_{\text{AUTO-V}}(\boldsymbol{g}_i; R) \tag{3.1}$$

and thus derive a novel private gradient $\sum_i R\frac{\boldsymbol{g}_i}{\|\boldsymbol{g}_i\|} + \sigma R \cdot \mathcal{N}(0, \mathbf{I})$. Here AUTO-V stands for the vanilla automatic clipping, which essentially performs the gradient normalization on each per-sample gradient. As a specific example, we can write the R-dependent automatic DP-SGD as

$$R\text{-dependent DP-SGD}_{\text{AUTO-V}} : \boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \frac{\eta}{|B_t|}\Big(\sum_{i\in B_t} R\frac{\partial l_i}{\partial \boldsymbol{w}_t}/\|\frac{\partial l_i}{\partial \boldsymbol{w}_t}\| + \sigma R \cdot \mathcal{N}(0, \mathbf{I})\Big) \tag{3.2}$$

We may view our AUTO-V clipping as to maximize the dot-product similarity between the clipped gradient and the regular gradient, a commonly used similarity measure, e.g. in the attention block in transformers [61]. Suppose we want

$$\max_{C_i} \left\langle \sum_i C_i \boldsymbol{g}_i, \sum_j \boldsymbol{g}_j \right\rangle \quad \text{s.t. } 0 \le C_i \le R/\|\boldsymbol{g}_i\|$$

Note that the constraint is a sufficient condition for clipping, as discussed in Section 2.2. It is not hard to see that the optimal clipping factor is

$$C_i = \begin{cases} R/\|\boldsymbol{g}_i\| & \text{if } \langle \boldsymbol{g}_i, \sum_j \boldsymbol{g}_j \rangle > 0 \\ 0 & \text{if } \langle \boldsymbol{g}_i, \sum_j \boldsymbol{g}_j \rangle \le 0 \end{cases}$$

If the per-sample gradients are indeed concentrated in the sense $\forall i, \langle \boldsymbol{g}_i, \sum_j \boldsymbol{g}_j \rangle \ge 0$, then AUTO-V is the optimal per-sample gradient clipping. We compare with Abadi's clipping in Figure 2, where the dot-product similarity is significantly magnified by our AUTO-V clipping.

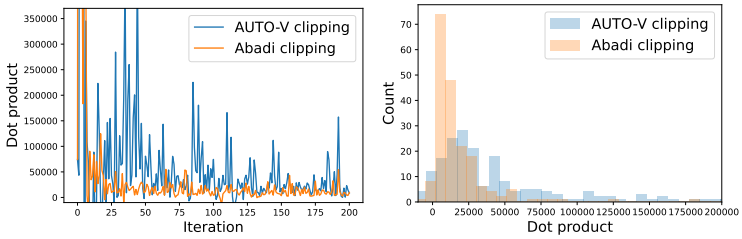

Figure 2: RoBERTa-base with DP-Adam ($\epsilon = 3$) on SST2 dataset, as in Section 6.2.

3.3 Stability constant breaks scale-invariance and remains stationary

One potential drawback of AUTO-V clipping is that all gradients lose their magnitudes information completely, since $\|\boldsymbol{g}_i \cdot \texttt{Clip}_{\text{AUTO-V}}(\boldsymbol{g}_i; R)\| = R, \forall i$. This scale-invariance in AUTO-V and partially in Abadi's clipping (when $\|\boldsymbol{g}_i\| > R$) leads to the "lazy region" issue: the parameters will not be updated by DP-GD even if the true gradients are non-zero. In Figure 3, we illustrate in a logistic regression[2] that AUTO-V and Abadi's clipping have zero clipped gradient for the trainable parameter $\theta \in [-2, 2]$, as the per-sample gradients from two classes cancel each other. Another benefit of γ is to remain stationary as $\boldsymbol{g}_i \to 0$, i.e. making $C_i\boldsymbol{g}_i \to \boldsymbol{g}_i/\gamma$ small rather than having magnitude R in AUTO-V. We elaborate this point in Section 4.3.

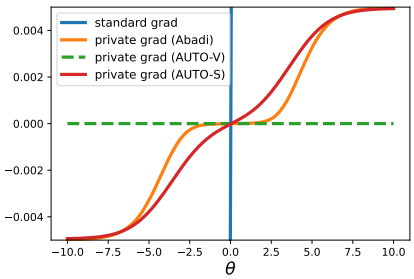

Figure 3: Scalar gradient $\frac{\partial \mathcal{L}}{\partial \theta}$ at each θ.

[2]The settings are in Appendix F, where the lazy region issues also emerge in the mean estimation problem.

To preserve the magnitude information and thus escape the lazy region, we propose the AUTO-S clipping, with a positive stability constant γ:

$$\texttt{Clip}_{\text{AUTO-S}}(\boldsymbol{g}_i; R) := R/(||\boldsymbol{g}_i|| + \gamma) \tag{3.3}$$

We visualize in Figure 4 that AUTO-S allows larger per-sample gradients to have larger magnitudes after the clipping, which in turn allows DP-SGD$_{\text{AUTO-S}}$ (but not DP-SGD$_{\text{AUTO-V}}$) to converge to zero gradient norms in Section 5.

4 Automatic DP Training

One may wonder why our clipping (3.1)(3.3) is automatic at all, if the hyperparameter R is still present and there is an additional parameter γ to choose. It turns out that any constant choice of $R > 0$ is equivalent to choosing $R = 1$, and common deep learning optimizers are insensitive to the choice of γ (see ablation study in Figure 13). Consequently, we choose $\gamma = 0.01$ in all our experiments. Specifically, let us redefine

$$\texttt{Clip}_{\text{AUTO-S}}(\boldsymbol{g}_i) := 1/(||\boldsymbol{g}_i|| + \gamma). \tag{4.1}$$

With this clipping, we can design automatic DP optimizers similar to (1.1):

$$\begin{aligned} \text{Automatic DP Optimizer}(\{\boldsymbol{g}_i\}_{i=1}^B) &= \text{Optimizer}(\hat{\boldsymbol{g}}_t) \\ \text{where } \hat{\boldsymbol{g}}_t &:= \sum_{i \in B_t} \frac{\boldsymbol{g}_{t,i}}{||\boldsymbol{g}_{t,i}|| + \gamma} + \sigma \cdot \mathcal{N}(0, \mathbf{I}) \end{aligned} \tag{4.2}$$

Clearly, the new private gradient $\hat{\boldsymbol{g}}_t$ from our automatic clipping is R-independent, in contrast to the one used in (1.1). A concrete example (in the case of $\gamma = 0$) that is comparable to (3.2) will be

$$R\text{-independent DP-SGD}_{\text{AUTO-V}}: \quad \boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \frac{\eta}{|B_t|}\Big(\sum_{i \in B_t} \frac{\partial l_i}{\partial \boldsymbol{w}_t} / \Big\| \frac{\partial l_i}{\partial \boldsymbol{w}_t} \Big\| + \sigma \cdot \mathcal{N}(0, \mathbf{I}) \Big) \tag{4.3}$$

Leveraging the private gradient $\hat{\boldsymbol{g}}_t$ in (4.2), we can train DP neural networks without tuning DP-specific hyperparamters R and σ, as demonstrated in Algorithm 1.

Algorithm 1 Automatic Deep Learning with DP

Parameters: initial weights $\boldsymbol{w}_0$, learning rate η_t, sampling probability p, number of iterations T.

1: Find σ such that $\epsilon_{\text{Accountant}}(\delta, \sigma, p, T) \leq \epsilon$ from any privacy accountant.
2: **for** iteration $t = 1, \cdots, T$ **do**
3: Sample a batch B_t by including each data point iid with probability p ($\mathbb{E}[\text{BatchSize}] = pn$).
4: Apply automatic clipping to per-sample gradients $\{\boldsymbol{g}_i\}_{i \in B_t}$: $\hat{\boldsymbol{g}}_i = \boldsymbol{g}_i / (\|\boldsymbol{g}_i\|_2 + 0.01)$.
5: Add Gaussian noise to the sum of clipped gradients: $\hat{\boldsymbol{g}} = \sum_i \hat{\boldsymbol{g}}_i + \sigma \cdot \mathcal{N}(0, \mathbf{I})$.
6: Update $\boldsymbol{w}_t$ by any optimizer on the private gradient $\hat{\boldsymbol{g}}$ with learning rate η_t.

We will elaborate two distinct reasons in each section for the following statement:

$$\boxed{\text{DP Optimizer}_{\text{Abadi}} \approx R\text{-dependent DP Optimizer}_{\text{AUTO}} \equiv R\text{-independent DP Optimizer}_{\text{AUTO}}}$$

which reduces the hyperparameter tuning of DP training to that of the regular training, i.e. only on learning rate, weight decay, etc. The significant save in the tuning effort is illustrated in Figure 14.

4.1 Non-adaptive optimizer couples clipping norm with learning rate

With R-dependent automatic clipping, DP-SGD becomes

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \frac{\eta}{|B_t|}\Big(\sum_{i \in B_t} \boldsymbol{g}_{t,i} \cdot \frac{R}{||\boldsymbol{g}_{t,i}|| + \gamma} + \sigma R \cdot \mathcal{N}(0, \mathbf{I}) \Big) = \boldsymbol{w}_t - \frac{\eta R}{|B_t|} \hat{\boldsymbol{g}}_t.$$

We can view $\eta_{\text{effective}} := \eta R$ as a whole: increasing R has the same effect as increasing η, which explains the diagonal pattern in Figure 1(right plot) where DP-SGD$_{\text{Abadi}}$ is applied with small clipping norm[3]. We extend to general non-adaptive optimizers in Theorem 1, with proof in Appendix B.1.

Theorem 1. *Non-adaptive R-dependent automatic DP optimizers (including SGD, Heavyball[52] and NAG[47]), with learning rate η and weight decay λ, is equivalent to R-independent automatic DP optimizers, with learning rate $\eta' = \eta R$ and weight decay $\lambda' = \lambda / R$.*

[3] When we further consider weight decay in automatic clipping (included in Theorem 1), increasing R is no longer equivalent to increasing η, as η also couples with the weight decay constant λ.

4.2 Adaptive optimizer can be insensitive to clipping norm

Adaptive automatic DP optimizers are different than the non-adaptive ones, as the clipping norm cancels out instead of being coupled with learning rate. To see this, we scrutinize DP-Adam$_{\text{Abadi}}$ (which is similar to DP-Adam$_{\text{AUTO-V}}$) in Figure 1(left plot), where columns to the left are almost identical. Further evidence is observed in [44, Table 5] that shrinking R has zero effect on LAMB. We now give a simple explanation using AdaGrad [18]:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta \frac{\boldsymbol{g}_t}{\sqrt{G_t}}$$

where $\boldsymbol{g}_t = \sum_i \boldsymbol{g}_{t,i}$ is the gradient sum and $G_t = \sum_{\tau<t} \boldsymbol{g}_\tau^2$ is sum of gradient square by Hadamard product over the past iterations. In R-dependent DP-AdaGrad$_{\text{AUTO-V}}$, the private gradient is $R\hat{\boldsymbol{g}}_t$ in place of the standard gradient sum $\boldsymbol{g}_t$, and $\hat{G}_t = R^2 \sum_{\tau\leq t} \hat{\boldsymbol{g}}_\tau^2$:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta \frac{R\hat{\boldsymbol{g}}_t}{\sqrt{\hat{G}_t}} = \boldsymbol{w}_t - \eta \frac{\hat{\boldsymbol{g}}_t}{\sqrt{\sum_{\tau<t} (\hat{\boldsymbol{g}}_\tau)^2}}.$$

We generalize to the general adaptive optimizers in Theorem 2, with proof in Appendix B.2.

Theorem 2. *Adaptive R-dependent automatic DP optimizers (including AdaGrad[18], AdaDelta[69], AdaMax/Adam[31], NAdam[16], RAdam[39], LARS[65], LAMB[66]), with learning rate η and weight decay λ is equivalent to R-independent automatic DP optimizers with learning rate η and weight decay $\lambda' = \lambda/R$. With decoupled weight decay[42], R-dependent automatic DP-AdamW is equivalent to R-independent automatic DP-AdamW with the same η and λ.*

Similarly, we demonstrate the automatic DP optimizers with per-layer clipping style in Appendix B.3.

4.3 Automatic clipping guarantees the same level of privacy while maximizes utility

In Theorem 3 (proved in Appendix A), we show that the new private gradient $\hat{\boldsymbol{g}}_t$ in (4.2) has the same level of privacy guarantee as the existing one in (1.1), since the global sensitivity remains the same (see Figure 4). We note that as long as $\gamma > 0$, the magnitude information of per-sample gradients is preserved by AUTO-S, in the sense that $\|\boldsymbol{g}_i\| > \|\boldsymbol{g}_j\| \iff \|C_i\boldsymbol{g}_i\| > \|C_j\boldsymbol{g}_j\|$, whereas this can be violated in both the AUTO-V and Abadi's clipping (as depicted by the flat curve in Figure 4 when $\|\boldsymbol{g}_i\| > 1$). Additionally, note that when γ is small, almost all data points "max out" the signal relative to the amount of noise we add. To say it differently, for the same amount of noise, AUTO-S with small γ allows more signal to be pushed through a differentially private channel. Towards the end of the training, i.e., at the limit when $\|\boldsymbol{g}_i\| \to 0$ for all i, then we have $\sum_i \frac{\boldsymbol{g}_i}{\|\boldsymbol{g}_i\|+\gamma} \to \frac{1}{\gamma}\sum_i \boldsymbol{g}_i$. In words, the clipped gradients become closer to the standard SGD, thus do not suffer from the instability of AUTO-V.

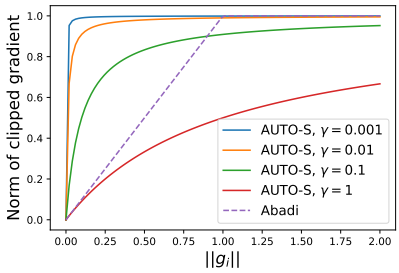

Figure 4: Gradient norms before and after being clipped by different methods at $R = 1$.

Theorem 3. *Under the noise multiplier σ, number of iterations T, subsampling probability B/n, DP optimizers using AUTO-V or AUTO-S clipping satisfy $(\epsilon_{\textit{Accountant}}(\delta, \sigma, B/n, T), \delta)$-DP as much as those using Abadi's clipping, regardless of the choice of privacy accountant.*

5 Convergence analysis of DP-SGD with automatic clipping

5.1 Convergence theory of DP-SGD to stationary points

We highlight that automatic clipping can be more amenable to analysis than Abadi's clipping in [12], since we no longer need to decide whether each per-sample gradient is clipped.

To analyze the convergence of automatic DP-SGD (4.2) in the non-convex setting, we follow the standard assumptions in the SGD literature [23, 2, 6], with one additional symmetry assumption on the gradient noise.

Assumption 5.1 (Lower bound of loss)**.** For all $\boldsymbol{w}$ and some constant $\mathcal{L}_*$, we have $\mathcal{L}(\boldsymbol{w}) \geq \mathcal{L}_*$.

Assumption 5.2 (Smoothness)**.** Let $\boldsymbol{g}(\boldsymbol{w})$ denote the gradient of the objective $\mathcal{L}(\boldsymbol{w})$. Then $\forall \boldsymbol{w}, \boldsymbol{v}$, there is an non-negative constant L such that

$$\mathcal{L}(\boldsymbol{v}) - \left[\mathcal{L}(\boldsymbol{w}) + \boldsymbol{g}(\boldsymbol{w})^\top(\boldsymbol{v} - \boldsymbol{w})\right] \leq \frac{L}{2}\|\boldsymbol{w} - \boldsymbol{v}\|^2. \tag{5.1}$$

Assumption 5.3 (Gradient noise)**.** The per-sample gradient noise $\tilde{\boldsymbol{g}}_{t,i} - \boldsymbol{g}_t$ is i.i.d. from some ditribution such that

$$\mathbb{E}(\tilde{\boldsymbol{g}}_{t,i} - \boldsymbol{g}_t) = 0, \mathbb{E}\|\tilde{\boldsymbol{g}}_{t,i} - \boldsymbol{g}_t\|^2 \leq \xi^2,$$

and $\tilde{\boldsymbol{g}}_{t,i}$ is centrally symmetric[4] about $\boldsymbol{g}_t$ in distribution:

$$\tilde{\boldsymbol{g}}_{t,i} - \boldsymbol{g}_t \overset{\mathcal{D}}{=} \boldsymbol{g}_t - \tilde{\boldsymbol{g}}_{t,i}.$$

We show in Theorem 4 that DP-SGD with AUTO-S clipping allows the true gradient norm to converge to zero, but not so with AUTO-V clipping. We leave the proof in Appendix C.1.

Theorem 4. *Under Assumption 5.1, 5.2, 5.3, running DP-SGD with automatic clipping for T iterations and setting the learning rate $\eta \propto 1/\sqrt{T}$ give*

$$\min_{0\leq t\leq T} \mathbb{E}(\|\boldsymbol{g}_t\|) \leq \mathcal{G}\left(\frac{4}{\sqrt{T}}\sqrt{(\mathcal{L}_0 - \mathcal{L}_*)L\left(1 + \frac{\sigma^2 d}{B^2}\right)}; \xi, \gamma\right) := \min_{r>0} \frac{\xi}{r} + \mathcal{F}\left(\cdots; r, \xi, \gamma\right). \tag{5.2}$$

Here $\cdots$ represents the first argument of $\mathcal{G}$, and $\mathcal{G}$ is increasing and positive. As $T \to \infty$, we have $\min_t \mathbb{E}(\|\boldsymbol{g}_t\|) = O(T^{-1/4})$, the same rate as the standard SGD given in Theorem 8.

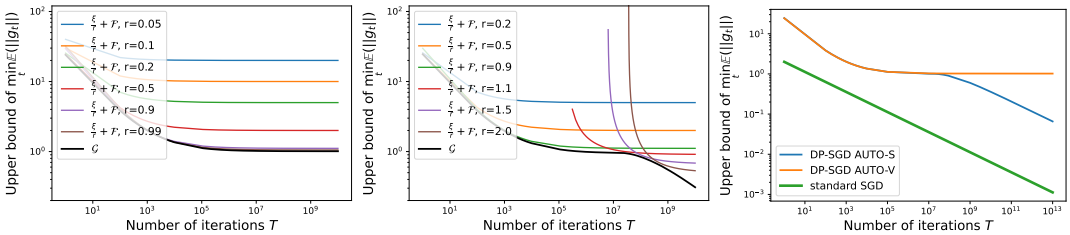

Figure 5: Left: DP-SGD with AUTO-V clipping. Middle: DP-SGD with AUTO-S clipping. Right: Log-log plot of convergence rate in comparison to standard SGD. Here $\xi = 1, \gamma = 0.01$, and the $O(1/\sqrt{T})$ term is set to 10 for DP-SGD and to 2 for standard SGD.

Remark 5.4. In Theorem 4, the upper bound takes an implicit form of $\mathcal{G}(\cdot; \xi, \gamma)$ because it is a lower envelope of functions $\frac{\xi}{r} + \mathcal{F}(\cdot; r, \xi, \gamma)$ over all possible $r > 0$, whose forms are detailed in Theorem 5. Notice that $\mathcal{G}$ results only from the clipping operation, not from the noise addition.

Remark 5.5. We show in Theorem 5 and demonstrate in Figure 5 that the upper bound (5.2) is always larger than ξ with AUTO-V ($\gamma = 0$), and can only be reduced to zero with AUTO-S ($\gamma > 0$). We provide real data evidence in Figure 12 that strictly positive γ reduces the gradient norm significantly.

5.2 Analysis of factors affecting the convergence

We now analyze the many factors that affect the convergence in Theorem 4, from a unified viewpoint of both the convergence and the privacy.

We start with the stability constant γ and the learning rate η_t, both only affect the convergence not the privacy. We empirically observe in Figure 7 that small γ benefits the convergence at initial iterations (when the privacy guarantee is strong) but larger γ converges faster asymptotically. For η_t, the optimal is in fact the miminizer of the hyperbola in (C.4), that is unique and tunable.

Next, we focus on the hyperparameters that affect both convergence and privacy: the batch size B, the noise multiplier σ, and the number of iterations T. These hyperparameters have to be considered along the privacy-accuracy tradeoff, not just from a convergence perspective. We denote the first argument of $\mathcal{G}$ in Theorem 4 as $X := \frac{4}{\sqrt{T}}\sqrt{(\mathcal{L}_0 - \mathcal{L}_*)L\left(1 + \frac{\sigma^2 d}{B^2}\right)}$, which is to be minimized.

[4]The symmetry assumption has been empirically verified in [12, Figure 3]. For theoretical analysis, it can be extended to mirror symmetry about the hyperplane normal to $\boldsymbol{g}_t$, that is $\{\boldsymbol{v} : \boldsymbol{g}_t^\top \boldsymbol{v} = 0\}$.

Given a fixed privacy budget (ϵ, δ), we leverage μ-GDP [15, 7] with $\mu \cdot n = B\sqrt{T(e^{1/\sigma^2} - 1)}$ for theoretical analysis only [5]. We summarized the following guidelines for DP training, which match the training behaviors observed in SOTA empirical work [36, 34, 13, 60, 44, 68].

1. **[Train longer with larger noise]** Fixing B, we have $X \propto \sqrt{1 + \frac{\sigma^2 d}{B^2}}/\sqrt{T} \approx \frac{\sigma\sqrt{d}}{B\sqrt{T}} \propto \sigma\sqrt{(e^{1/\sigma^2} - 1)}$, which is decreasing in σ. Hence larger σ and larger T are preferred.
2. **[Larger batch size helps]** Fixing iterations T or epochs $E = BT$, we again have $X \propto \sigma\sqrt{e^{1/\sigma^2} - 1}$. Hence larger σ and larger B are preferred.
3. **[Pretraining is key]** Pretraining can boost the accuracy by starting with a much smaller initial loss $\mathcal{L}_0$ and from a smooth (small L) and flat (small ξ, c.f. Figure 7(left)) initialization.
4. **[Learning rate needs tuning]** The optimal learning rate by minimizing (C.4) is $\sqrt{2(\mathcal{L}_0 - \mathcal{L}_*)/L/\left(1 + \frac{\sigma^2 d}{B^2}\right)}$. We should use larger learning rate for smaller model, weaker privacy (larger ϵ or small σ), or larger batch size (c.f. [36, Figure 2]).

6 Experiments

We evaluate our automatic DP training on image classification, sentence classification, and table-to-text generation tasks. Detailed settings including hyperparameters can be found in Appendix G.

6.1 Image classification

For MNIST/FashionMNIST, we use the same setup as in [49, 60, 57] with a simple CNN. For CIFAR10, we use the same setup as in [60] with pretrained SimCLRv2 [11]. For ImageNette [28], a 10-class sub-task of ImageNet [14], we use the same setup as in [32] without the learning rate decay. For CelebA [41], the real human face dataset, we train ResNet9 [27] with group normalization to replace the batch normalization. Notice that CelebA contains high-resolution (178x218) images, each with 40 labels. We consider CelebA for either multi-class classification on one label, e.g. 'Smiling' and 'Male', or for multi-label/multi-task problem to learn all labels simultaneously.

Task	Model	(ϵ, δ)	Accuracy %		
			Abadi's clipping	AUTO-S clipping	non-DP ($\epsilon = \infty$)
MNIST	4-layer CNN	$(3, 1e\text{-}5)$	98.04 ± 0.09	98.15 ± 0.07	99.11 ± 0.07
FashionMNIST	4-layer CNN	$(3, 1e\text{-}5)$	86.04 ± 0.26	86.36 ± 0.18	89.57 ± 0.13
CIFAR10 pretrained	SimCLRv2	$(2, 1e\text{-}5)$	92.44 ± 0.13	92.70 ± 0.02	94.42 ± 0.01
ImageNette	ResNet9	$(8, 1e\text{-}4)$	60.29 ± 0.53	60.71 ± 0.48	71.11 ± 0.37
CelebA [Smiling]	ResNet9	$(8, 5e\text{-}6)$	90.75 ± 0.11	91.08 ± 0.08	92.61 ± 0.20
CelebA [Male]	ResNet9	$(8, 5e\text{-}6)$	95.54 ± 0.14	95.70 ± 0.07	97.90 ± 0.04
CelebA Multi-label	ResNet9	$(3, 5e\text{-}6)$	86.81 ± 0.03	87.05 ± 0.01	90.30 ± 0.02
CelebA Multi-label	ResNet9	$(8, 5e\text{-}6)$	87.52 ± 0.15	87.58 ± 0.04	90.30 ± 0.02

Table 1: Average test accuracy and 95% confidence interval on image tasks over 5 runs.

In Table 1, we observe that AUTO-S clipping outperforms existing clipping in all datasets with statistical significance. Interestingly, the standard deviation from different runs is smaller for automatic DP optimizers, indicating better reproducibility and stability. We additionally experiment 40 binary classification problems on CelebA with respect to each label, and observe that the mean accuracy further improves to 91.63% at $\epsilon = 8$ for AUTO-S (see Appendix J).

6.2 Sentence classification

On five benchmark language datasets (MNLI(m/mm)[64], QQP[30], QNLI[55], SST2[59]), we compare our automatic DP training with reparameterized gradient perturbation (RGP, [68]) and full-parameter finetuning (full, [36]) using RoBERTa models [40]. These methods use the same experimental setup. For language models, our automatic training is based on the codebase of [36][6].

[5] Note for experiments, we use RDP [45] as the accountant. We use GDP for analysis because it is more analytic than RDP. From [7], μ can be uniquely determined from (ϵ, δ) via $\delta = \Phi(\mu/2 - \epsilon/\mu) - e^{\epsilon}\Phi(-\mu/2 - \epsilon/\mu)$. Similar to (ϵ, δ), smaller μ means stronger privacy guarantee.

[6] See `https://github.com/lxuechen/private-transformers` and the detailed modification in Appendix K.3.

Method	$\epsilon = 3$				$\epsilon = 8$				$\epsilon = \infty$ (non-DP)			
	MNLI	QQP	QNLI	SST2	MNLI	QQP	QNLI	SST2	MNLI	QQP	QNLI	SST2
RGP [68]	-	-	-	-	80.5/79.6	85.5	87.2	91.6	83.6/83.2	89.3	91.3	92.9
full [36]	82.45/82.99	85.56	**87.42**	91.86	83.20/83.46	86.08	**87.94**	92.09				
full Auto-V	81.21/82.03	84.72	86.56	91.86	82.18/82.64	**86.23**	87.24	92.09	85.91/86.14	87.34	91.40	94.49
full Auto-S	**83.22/83.21**	**85.76**	86.91	**92.32**	**83.82/83.55**	**86.58**	87.85	**92.43**				

Table 2: Test accuracy on language tasks with RoBERTa-base (12 blocks, 163 million parameters).

Method	$\epsilon = 3$				$\epsilon = 8$				$\epsilon = \infty$ (non-DP)			
	MNLI	QQP	QNLI	SST2	MNLI	QQP	QNLI	SST2	MNLI	QQP	QNLI	SST2
RGP [68]	-	-	-	-	86.1/86.0	86.7	90.0	93.0	-	-	-	-
full [36]	**86.43**/86.46	86.43	90.76	93.04	87.02/**87.26**	87.47	91.10	93.81				
full Auto-V	85.33/85.61	**86.61**	89.99	**93.12**	85.91/86.10	86.86	90.55	93.35	90.33/90.03	87.90	93.61	96.21
full Auto-S	86.27/**86.67**	**86.76**	**91.01**	**93.92**	**87.07**/87.16	**87.47**	**91.45**	**94.61**				

Table 3: Test accuracy on language tasks with RoBERTa-large (24 blocks, 407 million parameters).

In Table 2 and Table 3, we note that full parameter finetuning with AUTO-S outperforms or at least matches SOTA on all tasks. We use *exactly the same* hyperparameters as in [36].

6.3 Table-to-text generation

We compare our automatic DP training with a variety of fine-tuning methods, for table-to-text generation task on E2E dataset [19], where the goal is to generate texts about different aspects of a restaurant's data. We measure the success on this task by BLEU, ROUGE-L (in Table 4), METEOR, NIST, CIDEr (extended in Table 7), with higher value meaning better model quality.

Metric	DP guarantee	GPT2 large	GPT2 medium	GPT2							
		full AUTO-S	full AUTO-S	full AUTO-S	full AUTO-V	full [36]	LoRA [29]	RGP [68]	prefix [37]	top2 [36]	retrain [36]
BLEU	$\epsilon = 3$	**64.180**	**63.850**	**61.340**	**61.519**	**61.519**	58.153	58.482	47.772	25.920	15.457
	$\epsilon = 8$	**64.640**	**64.220**	**63.600**	63.189	63.189	**63.389**	58.455	49.263	26.885	24.247
	non-DP	66.840	68.500	69.463	69.463	69.463	69.682	68.328	68.845	65.752	65.731
ROGUE-L	$\epsilon = 3$	**67.857**	**67.071**	**65.872**	65.670	65.670	**65.773**	65.560	58.964	44.536	35.240
	$\epsilon = 8$	**68.968**	**67.533**	**67.073**	66.429	66.429	**67.525**	65.030	60.730	46.421	39.951
	non-DP	70.384	71.458	71.359	71.359	71.359	71.709	68.844	70.805	68.704	68.751

Table 4: Test performance on E2E dataset with GPT2. Additional performance measures are included in Table 7. The best two GPT2 models for each row are marked in bold.

Competitive methods include low-rank adaption (LoRA), prefix-tuning (prefix), RGP, only fine-tuning the top 2 Transformer blocks (top2), and training from scratch (retrain), as were recorded in [36]. Again, we use the *exactly the same* hyperparameters as in [36]. For GPT2 (163 million parameters), GPT2 medium (406 million), and GPT2 large (838 million), Table 4 shows that AUTO-S is scalable with stronger performance on larger models. Our automatic full-parameter finetuning has the best overall performance. Additionally, we highlight that AUTO-S and methods like LoRA are not mutually exclusive and can be combined to yield strong performance, since AUTO-S modifies the optimizers and LoRA modifies the architecture.

7 Discussion

In this work, we proposed AutoClipping as a drop-in replacement to the standard per-example clipping differentially private training. This is the first technique that eliminate the need to tune the clipping norm R, thus making DP deep learning as easy as regular learning. Our AUTO-S method enjoys both theoretical guarantee of convergence in non-convex problems (under various conditions), and strong empirical performance that advances the state-of-the-art (SOTA) of DP learning on both computer vision and language tasks.

We are excited about the future of automatic DP training, especially along with other working techniques. Notably, our automatic clipping applies compatibly with general optimizers (e.g. [8, 17]), clipping styles (all-layer or per-layer), architecture modifications (e.g. LoRA, RGP, prefix), and data augmentation (e.g. adversarial training [25] and multiple augmentation [13]). Thus, we expect to achieve comparable results to all SOTA in a lightweight fashion.

References

[1] 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*, pages 308–318, 2016.

[2] Z. Allen-Zhu. Natasha 2: Faster non-convex optimization than sgd. *Advances in neural information processing systems*, 31, 2018.

[3] G. Andrew, O. Thakkar, B. McMahan, and S. Ramaswamy. Differentially private learning with adaptive clipping. *Advances in Neural Information Processing Systems*, 34, 2021.

[4] S. Banerjee and A. Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.

[5] J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar. signsgd: Compressed optimisation for non-convex problems. In *International Conference on Machine Learning*, pages 560–569. PMLR, 2018.

[6] L. Bottou, F. E. Curtis, and J. Nocedal. Optimization methods for large-scale machine learning. *Siam Review*, 60(2):223–311, 2018.

[7] Z. Bu, J. Dong, Q. Long, and W. J. Su. Deep learning with gaussian differential privacy. *Harvard data science review*, 2020(23), 2020.

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

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

[10] N. Carlini, F. Tramer, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, U. Erlingsson, et al. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650, 2021.

[11] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[12] X. Chen, S. Z. Wu, and M. Hong. Understanding gradient clipping in private sgd: A geometric perspective. *Advances in Neural Information Processing Systems*, 33:13773–13782, 2020.

[13] 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.

[14] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[15] J. Dong, A. Roth, and W. J. Su. Gaussian differential privacy. *arXiv preprint arXiv:1905.02383*, 2019.

[16] T. Dozat. Incorporating nesterov momentum into adam. 2016.

[17] J. Du and H. Mi. Dp-fp: Differentially private forward propagation for large models. *arXiv preprint arXiv:2112.14430*, 2021.

[18] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

[19] O. Dusek, J. Novikova, and V. Rieser. Evaluating the State-of-the-Art of End-to-End Natural Language Generation: The E2E NLG Challenge. *Computer Speech & Language*, 59:123–156, Jan. 2020.

[20] C. Dwork. Differential privacy: A survey of results. In *International conference on theory and applications of models of computation*, pages 1–19. Springer, 2008.

[21] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

[22] C. Dwork, A. Roth, et al. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2014.

[23] S. Ghadimi and G. Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

[24] A. Golatkar, A. Achille, Y.-X. Wang, A. Roth, M. Kearns, and S. Soatto. Mixed differential privacy in computer vision. *arXiv preprint arXiv:2203.11481*, 2022.

[25] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[26] S. Gopi, Y. T. Lee, and L. Wutschitz. Numerical composition of differential privacy. *Advances in Neural Information Processing Systems*, 34, 2021.

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

[28] J. Howard. Imagewang.

[29] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[30] S. Iyer, N. Dandekar, and K. Csernai. First quora dataset release: Question pairs, 2017.

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

[32] 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.

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

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

[35] Q. Lhoest, A. Villanova del Moral, Y. Jernite, A. Thakur, P. von Platen, S. Patil, J. Chaumond, M. Drame, J. Plu, L. Tunstall, J. Davison, M. Sasko, G. Chhablani, B. Malik, S. Brandeis, T. Le Scao, V. Sanh, C. Xu, N. Patry, A. McMillan-Major, P. Schmid, S. Gugger, C. Delangue, T. Matussière, L. Debut, S. Bekman, P. Cistac, T. Goehringer, V. Mustar, F. Lagunas, A. Rush, and T. Wolf. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics.

[36] X. Li, F. Tramer, P. Liang, and T. Hashimoto. Large language models can be strong differentially private learners. *arXiv preprint arXiv:2110.05679*, 2021.

[37] X. L. Li and P. Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[38] C.-Y. Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[39] L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and J. Han. On the variance of the adaptive learning rate and beyond. *arXiv preprint arXiv:1908.03265*, 2019.

[40] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[41] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[42] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[43] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang. Learning differentially private recurrent language models. *arXiv preprint arXiv:1710.06963*, 2017.

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

[45] I. Mironov. Rényi differential privacy. In *2017 IEEE 30th computer security foundations symposium (CSF)*, pages 263–275. IEEE, 2017.

[46] D. Misra. Mish: A self regularized non-monotonic activation function. *arXiv preprint arXiv:1908.08681*, 2019.

[47] Y. E. Nesterov. A method for solving the convex programming problem with convergence rate o (1/kˆ 2). In *Dokl. akad. nauk Sssr*, volume 269, pages 543–547, 1983.

[48] N. Papernot and T. Steinke. Hyperparameter tuning with renyi differential privacy. *arXiv preprint arXiv:2110.03620*, 2021.

[49] N. Papernot, A. Thakurta, S. Song, S. Chien, and U. Erlingsson. Tempered sigmoid activations for deep learning with differential privacy.

[50] K. Papineni, S. Roukos, T. Ward, and W. jing Zhu. Bleu: a method for automatic evaluation of machine translation. pages 311–318, 2002.

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

[52] B. T. Polyak. Some methods of speeding up the convergence of iteration methods. *Ussr computational mathematics and mathematical physics*, 4(5):1–17, 1964.

[53] B. T. Polyak and A. B. Juditsky. Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization*, 30(4):838–855, 1992.

[54] S. Qiao, H. Wang, C. Liu, W. Shen, and A. Yuille. Micro-batch training with batch-channel normalization and weight standardization. *arXiv preprint arXiv:1903.10520*, 2019.

[55] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

[56] S. O. Sadjadi, T. Kheyrkhah, A. Tong, C. S. Greenberg, D. A. Reynolds, E. Singer, L. P. Mason, J. Hernandez-Cordero, et al. The 2017 nist language recognition evaluation. In *Odyssey*, pages 82–89, 2018.

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

[58] 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)*, pages 3–18. IEEE, 2017.

[59] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

[60] F. Tramer and D. Boneh. Differentially private learning needs better features (or much more data). *arXiv preprint arXiv:2011.11660*, 2020.

[61] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[62] R. Vedantam, C. Lawrence Zitnick, and D. Parikh. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575, 2015.

[63] Y.-X. Wang, B. Balle, and S. P. Kasiviswanathan. Subsampled rényi differential privacy and analytical moments accountant. In *International Conference on Artificial Intelligence and Statistics*, pages 1226–1235. PMLR, 2019.

[64] A. Williams, N. Nangia, and S. Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics, 2018.

[65] Y. You, I. Gitman, and B. Ginsburg. Scaling sgd batch size to 32k for imagenet training. *arXiv preprint arXiv:1708.03888*, 6(12):6, 2017.

[66] Y. You, J. Li, S. Reddi, J. Hseu, S. Kumar, S. Bhojanapalli, X. Song, J. Demmel, K. Keutzer, and C.-J. Hsieh. Large batch optimization for deep learning: Training bert in 76 minutes. *arXiv preprint arXiv:1904.00962*, 2019.

[67] A. Yousefpour, I. Shilov, A. Sablayrolles, D. Testuggine, K. Prasad, M. Malek, J. Nguyen, S. Ghosh, A. Bharadwaj, J. Zhao, G. Cormode, and I. Mironov. Opacus: User-friendly differential privacy library in PyTorch. *arXiv preprint arXiv:2109.12298*, 2021.

[68] D. Yu, H. Zhang, W. Chen, J. Yin, and T.-Y. Liu. Large scale private learning via low-rank reparametrization. In *International Conference on Machine Learning*, pages 12208–12218. PMLR, 2021.

[69] M. D. Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[70] Y. Zhu, J. Dong, and Y.-X. Wang. Optimal accounting of differential privacy via characteristic function. *arXiv preprint arXiv:2106.08567*, 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [N/A]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Assumptions 5.1-5.3.
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix A.
3. If you ran experiments...

(a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We include our code in Appendix K, and cite all datasets.

(b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See all details in Appendix G.

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Table 1.

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [N/A] This work is unrelated to computational efficiency.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [Yes] We cite all codebases, datasets, models clearly.

(b) Did you mention the license of the assets? [N/A] Experiments are conducted on public data.

(c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We include code snippet in Appendix K and will include a URL to release the full codebase.

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]