

---

# Towards Better Generalization of Adaptive Gradient Methods

---

Anonymous Author(s)

Affiliation

Address

email

## Abstract

Adaptive gradient methods such as AdaGrad, RMSprop and Adam have been optimizers of choice for deep learning due to their fast training speed. However, it was recently observed that their generalization performance is often worse than that of SGD for over-parameterized neural networks. While new algorithms such as AdaBound, SWAT, and Padam were proposed to improve the situation, the provided analyses are only committed to optimization bounds for the training objective, leaving critical generalization capacity unexplored. To close this gap, we propose *Stable Adaptive Gradient Descent* (SAGD) for nonconvex optimization which leverages differential privacy to boost the generalization performance of adaptive gradient methods. Theoretical analyses show that SAGD has high-probability convergence to a population stationary point. We further conduct experiments on various popular deep learning tasks and models. Experimental results illustrate that SAGD is empirically competitive and often better than baselines.

## 1 Introduction

We consider in this paper, the following minimization problem:

$$\min_{\mathbf{w} \in \mathcal{W}} f(\mathbf{w}) \triangleq \mathbb{E}_{z \sim \mathcal{P}}[\ell(\mathbf{w}, z)], \quad (1)$$

where the *population loss*  $f$  is a (possibly) nonconvex objective function (as for most deep learning tasks),  $\mathcal{W} \subset \mathbb{R}^d$  is the parameter set and  $z$  is the vector of data samples distributed according to an unknown data distribution  $\mathcal{P}$ . We assume that we have access to an oracle that, given  $n$  i.i.d. samples  $(\mathbf{z}_1, \dots, \mathbf{z}_n)$ , returns the stochastic objectives  $(\ell(\mathbf{w}, \mathbf{z}_1), \dots, \ell(\mathbf{w}, \mathbf{z}_n))$ . Our goal is to find critical points of the population loss function (1). Given the unknown data distribution, a natural approach towards solving (1) is empirical risk minimization (ERM) [31], which minimizes the *empirical loss*  $\hat{f}(\mathbf{w})$  as follows:  $\min_{\mathbf{w} \in \mathcal{W}} \hat{f}(\mathbf{w}) \triangleq \frac{1}{n} \sum_{j=1}^n \ell(\mathbf{w}, \mathbf{z}_j)$ , when  $n$  samples  $\mathbf{z}_1, \dots, \mathbf{z}_n$  are observed. Stochastic gradient descent (SGD) [30] which iteratively updates the parameter of a model by descending in the direction of the negative gradient, computed on a single sample or a mini-batch of samples, has been the most dominant algorithm for solving the ERM problem, e.g., training deep neural networks. To automatically tune the learning-rate decay in SGD, adaptive gradient methods, such as AdaGrad [8], RMSprop [33], and Adam [17], have emerged leveraging the curvature of the objective function resulting in adaptive coordinate-wise learning rates for faster convergence.

However, the generalization ability of these adaptive methods is often worse than that of SGD for over-parameterized neural networks, e.g., convolutional neural network (CNN) for image classification and recurrent neural network (RNN) for language modeling [37]. To mitigate this issue, several recent algorithms were proposed to combine adaptive methods with SGD. For example, AdaBound [22] and SWAT [16] switch from Adam to SGD as the training proceeds, while Padam [5, 39] unifies AMSGrad [29] and SGD with a partially adaptive parameter. Despite much efforts on deriving theoretical convergence results of the objective function [38, 36, 41, 6], these newly

proposed adaptive gradient methods are often misunderstood regarding their generalization abilities, which is the ultimate goal. On the other hand, current adaptive gradient methods [8, 17, 33, 29, 36] follow a typical stochastic optimization (SO) oracle paradigm [30, 13] which uses stochastic gradients to update the parameters. The SO oracle requires *new samples* at every iteration to get the stochastic gradient such that, in expectation, it equals the population gradient. In practice, however, only finite training samples are available and reused by the optimization oracle for a certain number of times (*i.e.*, epochs). Hardt et al. [14] found that the generalization error increases with the number of times the optimization oracle passes over the training data. It is thus expected that gradient descent algorithms will be much more well-behaved if we have access to an infinite number of fresh samples. Re-using data samples is therefore a caveat for the generalization of a given algorithm.

In order to tackle the above issues, we propose *Stable Adaptive Gradient Descent* (SAGD) which aims at improving the generalization of general adaptive gradient descent algorithms. SAGD behaves similarly to the aforementioned ideal case of infinite fresh samples borrowing ideas from *adaptive data analysis* [10] and *differential privacy* [9]. The main idea of our method is that, at each iteration, SAGD accesses the observations  $z$  through a differentially private mechanism and computes an estimated gradient  $\nabla \ell(\mathbf{w}, z)$  of the objective function  $\nabla f(\mathbf{w})$ . It then uses the estimated gradient to perform a descent step using adaptive stepsize. We prove that the reused data points in SAGD nearly possess the statistical nature of *fresh samples* yielding to high concentration bounds of the population gradients through the iterations. Our contributions can be summarized as follows:

- We derive a novel adaptive gradient method, namely SAGD, leveraging ideas of differential privacy and adaptive data analysis aiming at improving the generalization of current baseline methods. A mini-batch variant is also introduced for large-scale learning tasks.
- Our differentially private mechanism, embedded in the SAGD, explores the idea of Laplace Mechanism (adding Laplace noises to gradients) and THRESHOLDOUT [9] leading to DPG-LAP and DPG-SPARSE methods saving privacy cost. In particular, we show that differentially private gradients stay close to the population gradients with high probability.
- We establish various theoretical guarantees for our algorithm. We derive a concentration bound on the generalization error and show that the  $\ell_2$ -norm of the *population gradient*, *i.e.*,  $\|\nabla f(\mathbf{w})\|$  obtained by the SAGD converges in  $\mathcal{O}(1/n^{2/3})$  with high probability.
- We conduct several experimental applications based on training neural networks for image classification and language modeling indicating that SAGD outperforms existing adaptive gradient methods in terms of the generalization and over-fitting performance.

**Roadmap:** The SAGD algorithm, including the differentially private mechanisms, and its mini-batch variant are described in Section 3. Numerical experiments are presented Section 4. Section 5 concludes our work. Due to space limit, most of the proofs are relegated to supplementary material.

**Notations:** We use  $\mathbf{g}_t$  and  $\nabla f(\mathbf{w})$  interchangeably to denote the *population gradient* such that  $\mathbf{g}_t = \nabla f(\mathbf{w}_t) = \mathbb{E}_{\mathbf{z} \in \mathcal{P}}[\nabla \ell(\mathbf{w}_t, \mathbf{z})]$ .  $S = \{\mathbf{z}_1, \dots, \mathbf{z}_n\}$  denotes the  $n$  available training samples.  $\hat{\mathbf{g}}_t$  denotes the sample gradient evaluated on  $S$  such that  $\hat{\mathbf{g}}_t = \nabla \hat{f}(\mathbf{w}) = \frac{1}{n} \sum_{j=1}^n \nabla \ell(\mathbf{w}_t, \mathbf{z}_j)$ . For a vector  $\mathbf{v}$ ,  $\mathbf{v}^2$  represents that  $\mathbf{v}$  is element-wise squared. We use  $\mathbf{v}^i$  or  $[\mathbf{v}]_i$  to denote the  $i$ -th coordinate of  $\mathbf{v}$  and  $\|\mathbf{v}\|_2$  is the  $\ell_2$ -norm of  $\mathbf{v}$  and denote  $[d] = \{1, \dots, d\}$ .

## 2 Preliminaries

**Adaptive Gradient Methods:** In the nonconvex setting, existing work on SGD [13] and adaptive gradient methods [38, 36, 41, 6] show convergence to a stationary point with a rate of  $\mathcal{O}(1/\sqrt{T})$  where  $T$  is the number of stochastic gradient computations. Given  $n$  samples, a stochastic oracle can obtain at most  $n$  stochastic gradients, which implies convergence to the population stationarity with a rate of  $\mathcal{O}(1/\sqrt{n})$ . In addition, Kuzborskij and Lampert [19], Raginsky et al. [28], Hardt et al. [14], Mou et al. [25], Pensia et al. [26], Chen et al. [6], Li et al. [21] study the generalization of gradient-based optimization algorithms using the generalization property of stable algorithms [2]. In particular, Raginsky et al. [28], Mou et al. [25], Li et al. [21], Pensia et al. [26] focus on noisy gradient algorithms, *e.g.*, SGLD, and provide a generalization bound in  $\mathcal{O}(\sqrt{T}/n)$ . This type of bounds usually has a dependence on the training data and has a polynomial dependence on  $T$ .

---

**Algorithm 1** SAGD with DGP-LAP

---

```

1: Input: Dataset  $S$ , certain loss  $\ell(\cdot)$ , initial point  $\mathbf{w}_0$  and noise level  $\sigma$ .
2: Set noise level  $\sigma$ , iteration number  $T$ , and stepsize  $\eta_t$ .
3: for  $t = 0, \dots, T - 1$  do
4:   DPG-LAP: Compute full batch gradient on  $S$ :
      
$$\hat{\mathbf{g}}_t = \frac{1}{n} \sum_{j=1}^n \nabla \ell(\mathbf{w}_t, z_j).$$

5:   Set  $\tilde{\mathbf{g}}_t = \hat{\mathbf{g}}_t + \mathbf{b}_t$ , where  $\mathbf{b}_t^i$  is drawn i.i.d from  $\text{Lap}(\sigma)$  for all  $i \in [d]$ .
6:    $\mathbf{m}_t = \tilde{\mathbf{g}}_t$  and  $\mathbf{v}_t = (1 - \beta_2) \sum_{i=1}^t \beta_2^{t-i} \tilde{\mathbf{g}}_i^2$ .
7:    $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \mathbf{m}_t / (\sqrt{\mathbf{v}_t} + \nu)$ .
8: end for

```

---

**Differential Privacy and Adaptive Data Analysis:** Differential privacy [9] was originally studied for preserving the privacy of individual data in the statistical query. Recently, differential privacy has been widely used for stochastic optimization. Some pioneering work [4, 1, 35] introduce differential privacy to empirical risk minimization (ERM) to protect sensitive information of the training data. The popular differentially private algorithms include the gradient perturbation that adds noise to the gradient in gradient descent algorithms [4, 1, 34]. Moreover, in Adaptive Data Analysis ADA [10, 11, 12], the same holdout set is used multiple times to test the hypotheses which are generated based on previous test results. It has been shown that reusing the holdout set via a differentially private mechanism ensures the validity of the test. In other words, the differentially private reused dataset maintains the statistical nature of fresh samples and improves generalization [40].

### 3 Stable Adaptive Gradient Descent Algorithm

Beforehand, we recall the definition of a  $(\epsilon, \delta)$ -differentially private algorithm:

**Definition 1.** (Differential Privacy [9]) A randomized algorithm  $\mathcal{M}$  is  $(\epsilon, \delta)$ -differentially private if

$$\mathbb{P}\{\mathcal{M}(\mathcal{D}) \in \mathcal{Y}\} \leq \exp(\epsilon) \mathbb{P}\{\mathcal{M}(\mathcal{D}') \in \mathcal{Y}\} + \delta$$

holds for all  $\mathcal{Y} \subseteq \text{Range}(\mathcal{M})$  and all pairs of adjacent datasets  $\mathcal{D}, \mathcal{D}'$  that differ on a single sample.

The general approach for achieving  $(\epsilon, \delta)$ -differential privacy when estimating a deterministic real-valued function  $q : \mathcal{Z}^n \rightarrow \mathbb{R}^d$  is Laplace Mechanism [9], which adds Laplace noise calibrated to the function  $q$ , i.e.,  $\mathcal{M}(\mathcal{D}) = q(\mathcal{D}) + \mathbf{b}$ , where for all  $i \in [d]$ ,  $\mathbf{b}^i \sim \text{Laplace}(0, \sigma^2)$ . We present SAGD with two different Differential Private Gradient (DPG) computing methods that provide an estimate of the gradient  $\nabla f(\mathbf{w})$ , namely DPG-LAP based on the *Laplace Mechanism* [9], see Section 3.1 and an improvement named DPG-SPARSE motivated by sparse vector technique [9] in Section 3.2.

#### 3.1 SAGD with DGP-LAP

In most deep learning applications, a training set  $S$  of size  $n$  is observed. Then, at each iteration  $t \in [T]$ , SAGD, described in Algorithm 1, calls DPG-LAP (Line 5 in Algorithm 1), that computes the empirical gradient noted  $\tilde{\mathbf{g}}_t$  and updates the model parameter  $\mathbf{w}_{t+1}$  using adaptive stepsize. Note that the noise variance  $\sigma^2$ , step-size  $\eta_t$ , iteration number  $T$ ,  $\beta_2$  are parameters and play an important role for our theoretical study presented in the sequel. We first consider DPG-LAP which adds Laplace noise  $\mathbf{b}_t \in \mathbb{R}^d$  to the empirical gradient  $\hat{\mathbf{g}}_t = \frac{1}{n} \sum_{j=1}^n \nabla \ell(\mathbf{w}_t, z_j)$  and returns a noisy gradient  $\tilde{\mathbf{g}}_t = \hat{\mathbf{g}}_t + \mathbf{b}_t$  to the optimization oracle Algorithm 1. Throughout the paper, assume:

**A1.** The objective function  $f : \mathbb{R}^d \rightarrow \mathbb{R}$  is bounded from below by  $f^*$  and is  $L$ -smooth ( $L$ -Lipschitz gradients), i.e.,  $\|\nabla f(\mathbf{w}) - \nabla f(\mathbf{w}')\| \leq L\|\mathbf{w} - \mathbf{w}'\|$ , for all  $\mathbf{w}, \mathbf{w}' \in \mathcal{W}$ .

**A2.** The gradient of  $\ell$  and its noisy approximation are bounded: For all  $\mathbf{w} \in \mathcal{W}$ ,  $\mathbf{z} \in \mathcal{Z}$   $\|\nabla \ell(\mathbf{w}, z)\|_1 \leq G_1$  and for all  $t \in [T]$ ,  $\|\tilde{\mathbf{g}}_t\|_2 \leq G$ .

**High-probability bound:** We first show that the noisy gradient  $\tilde{\mathbf{g}}_t$  approximates the population gradient  $\mathbf{g}_t$  with high probability. A general approach for analyzing such estimation error  $|\tilde{\mathbf{g}}_t - \mathbf{g}_t|$  is the Hoeffding's bound. Indeed, given training set  $S \in \mathcal{Z}^n$ , where  $\mathcal{Z} \subset \mathbb{R}$ , and a fixed  $\mathbf{w}_0$  chosen to be independent of the dataset  $S$ , denote the empirical gradient  $\hat{\mathbf{g}}_0 = \mathbb{E}_{z \in S} \nabla \ell(\mathbf{w}_0, z)$  and population gradient  $\mathbf{g}_0 = \mathbb{E}_{z \sim \mathcal{P}} [\nabla \ell(\mathbf{w}_0, z)]$  then, Hoeffding's bound implies for coordinate  $i \in [d]$  and  $\mu > 0$ :

$$P\{|\hat{\mathbf{g}}_0^i - \mathbf{g}_0^i| \geq \mu\} \leq 2 \exp\left(\frac{-2n\mu^2}{4G_\infty^2}\right), \quad (2)$$

where  $G_\infty$  is the maximal value of the  $\ell_\infty$ -norm of the gradient  $\mathbf{g}_0$ . Generally, if  $\mathbf{w}_1$  is updated using the gradient computed on training set  $S$ , i.e.,  $\mathbf{w}_1 = \mathbf{w}_0 - \eta \hat{\mathbf{g}}_0$ , concentration inequality (2) will not hold for  $\hat{\mathbf{g}}_1 = \mathbb{E}_{z \in S} \nabla_i \ell(\mathbf{w}_1, z)$ , because  $\mathbf{w}_1$  is no longer independent of  $S$ . For any differentially private algorithm, Lemma 1 provides the following high probability concentration bound:

**Lemma 1.** *Let  $\mathcal{A}$  be an  $(\epsilon, \delta)$ -differentially private gradient descent algorithm with access to training set  $S$  of size  $n$ . Let  $\mathbf{w}_t = \mathcal{A}(S)$  be the parameter generated at iteration  $t \in [T]$  and  $\hat{\mathbf{g}}_t$  the empirical gradient on  $S$ . For any  $\sigma > 0$ ,  $\beta > 0$ , if the privacy cost of  $\mathcal{A}$  satisfies  $\epsilon \leq \sigma/13$ ,  $\delta \leq \sigma\beta/(26 \ln(26/\sigma))$ , and sample size  $n \geq 2 \ln(8/\delta)/\epsilon^2$ , we then have*

$$\mathbb{P} \{ |\hat{\mathbf{g}}_t^i - \mathbf{g}_t^i| \geq \sigma \} \leq \beta \quad \text{for every } i \in [d] \text{ and every } t \in [T].$$

Lemma 1 is an instance of Theorem 8 from [10] and illustrates that, if the privacy cost  $\epsilon$  is bounded by the estimation error, the differential privacy mechanism enables the reused training samples set to maintain statistical guarantees as if they were fresh samples. Then, we establish in Lemma 2, that SAGD with DPG-LAP is a differentially private algorithm with the following privacy cost:

**Lemma 2.** *SAGD with DPG-LAP (Alg. 1) is  $(\frac{\sqrt{T \ln(1/\delta)} G_1}{n\sigma}, \delta)$ -differentially private.*

In order to achieve a gradient concentration bound for SAGD with DPG-LAP as described in Lemma 1, we set  $\sqrt{T \ln(1/\delta)} G_1/(n\sigma) \leq \sigma/13$ ,  $\delta \leq \sigma\beta/(26 \ln(26/\sigma))$ , and sample size  $n \geq 2 \ln(8/\delta)/\epsilon^2$ . Then, the following result shows that across all iterations, gradients produced by SAGD with DPG-LAP maintain high probability concentration bounds.

**Theorem 1.** *Given  $\sigma > 0$ , let  $\tilde{\mathbf{g}}_1, \dots, \tilde{\mathbf{g}}_T$  be gradients computed by DPG-LAP in SAGD. Set the number of iterations  $2n\sigma^2/G_1^2 \leq T \leq n^2\sigma^4/(169 \ln(1/(\sigma\beta))G_1^2)$ , then for  $t \in [T]$ ,  $\beta > 0$ ,  $\mu > 0$ :*

$$\mathbb{P} \{ \|\tilde{\mathbf{g}}_t - \mathbf{g}_t\| \geq \sqrt{d}\sigma(1 + \mu) \} \leq d\beta + d \exp(-\mu).$$

Note that given the concentration error bound of  $\sqrt{d}\sigma(1 + \mu)$ , Theorem 1 indicates that a higher noise level  $\sigma$ , implying a better privacy guarantee and a larger number of iterations  $T$ , would meanwhile incur a larger concentration error. Thus, there is a trade-off between noise and accuracy illustrated by the positive numbers  $\beta$  and  $\mu$ . A larger  $\mu$  brings a larger concentration error but a smaller probability. A larger  $\beta$  implies a larger upper bound on  $T$ , yet also a larger probability bound. Note that although the probability  $d\beta + d \exp(-\mu)$  has a dependence on dimension  $d$ , we can choose appropriate  $\beta$  and  $\mu$  to make the probability arbitrarily small when analyzing the convergence to a stationary point.

**Non-asymptotic convergence rate:** We derive the optimal values of  $\sigma$  and  $T$  to improve the trade-off between the statistical rate and the optimization rate and we obtain a novel finite-time bound in Theorem 2. Denote  $\rho_{n,d} \triangleq \mathcal{O}(\ln n + \ln d)$ , we prove that SAGD with DPG-LAP converges to a population stationary point with high probability at the following rate:

**Theorem 2.** *Given training set  $S$  of size  $n$ , for  $\nu > 0$ , if  $\eta_t = \eta$  with  $\eta \leq \nu/(2L)$ ,  $\sigma = 1/n^{1/3}$ , iteration number  $T = n^{2/3}/(169G_1^2(\ln d + 7 \ln n/3))$ ,  $\mu = \ln(1/\beta)$  and  $\beta = 1/(dn^{5/3})$ , then SAGD with DPG-LAP algorithm yields:*

$$\min_{1 \leq t \leq T} \|\nabla f(\mathbf{w}_t)\|^2 \leq \mathcal{O} \left( \frac{\rho_{n,d} (f(\mathbf{w}_1) - f^*)}{n^{2/3}} \right) + \mathcal{O} \left( \frac{d\rho_{n,d}^2}{n^{2/3}} \right),$$

with probability at least  $1 - \mathcal{O}(1/(\rho_{n,d}n))$ .

Theorem 2 shows that, given  $n$  samples, SAGD converges to a stationary point at a rate of  $\mathcal{O}(1/n^{2/3})$  where we use the  $\ell_2$  norm of the gradient of the objective function as a convergence criterion. Particularly, the first term of the bound corresponds to the optimization error  $\mathcal{O}(1/T)$  with  $T = \mathcal{O}(n^{2/3})$ , while the second is the statistical error depending on available sample size  $n$  and dimension  $d$ . The current optimization analyses [38, 36, 41, 6] show that adaptive gradient descent algorithms converge to the stationary point of the objective function with a rate of  $\mathcal{O}(1/\sqrt{T})$  with  $T$  stochastic gradient computations. Given  $n$  samples, their analyses yield a rate of  $\mathcal{O}(1/\sqrt{n})$ . Thus, the SAGD achieves a sharper bound compared to the previous analyses.

### 3.2 SAGD with DPG-SPARSE

In this section, we consider the SAGD with an advanced version of DPG named DPG-SPARSE motivated by the sparse vector technique [9] aiming to provide a sharper result on the privacy cost  $\epsilon$  and  $\delta$ . Lemma 2 shows that the privacy cost of SAGD with DPG-LAP scales with  $\mathcal{O}(\sqrt{T})$ . In order to guarantee the generalization of SAGD as stated in Theorem 1, we need to control the privacy cost below a certain threshold i.e.,  $\sqrt{T \ln(1/\delta)} G_1 / (n\sigma) \leq \sigma/13$ . However, it limits the iteration number  $T$  of SAGD, leading to a compromised optimization term in Theorem 2. In order to relax the upper bound on  $T$ , we propose the SAGD with DPG-SPARSE in Algorithm 2. Given  $n$  samples, Algorithm 2 splits the dataset evenly into two parts  $S_1$  and  $S_2$ . At each iteration  $t$ , Algorithm 2 computes gradients on both datasets:  $\hat{\mathbf{g}}_{S_1,t} = \frac{1}{|S_1|} \sum_{\mathbf{z}_j \in S_1} \nabla \ell(\mathbf{w}_t, \mathbf{z}_j)$  and  $\hat{\mathbf{g}}_{S_2,t} = \frac{1}{|S_2|} \sum_{\mathbf{z}_j \in S_2} \nabla \ell(\mathbf{w}_t, \mathbf{z}_j)$ . It then validates  $\hat{\mathbf{g}}_{S_1,t}$  with  $\hat{\mathbf{g}}_{S_2,t}$ , i.e., if the norm of their difference is greater than a random threshold  $\tau - \gamma$ , it returns  $\tilde{\mathbf{g}}_t = \hat{\mathbf{g}}_{S_1,t} + \mathbf{b}_t$ , otherwise  $\tilde{\mathbf{g}}_t = \hat{\mathbf{g}}_{S_2,t}$ .

---

#### Algorithm 2 SAGD with DPG-SPARSE

---

```

1: Input: Dataset  $S$ , certain loss  $\ell(\cdot)$ , initial point  $\mathbf{w}_0$ .
2: Set noise level  $\sigma$ , iteration number  $T$ , and stepsize  $\eta_t$ .
3: Split  $S$  randomly into  $S_1$  and  $S_2$ .
4: for  $t = 0, \dots, T - 1$  do
5:   DPG-SPARSE: Compute full batch gradient on  $S_1$  and  $S_2$ :
        $\hat{\mathbf{g}}_{S_1,t} = \frac{1}{|S_1|} \sum_{\mathbf{z}_j \in S_1} \nabla \ell(\mathbf{w}_t, \mathbf{z}_j)$ ,  $\hat{\mathbf{g}}_{S_2,t} = \frac{1}{|S_2|} \sum_{\mathbf{z}_j \in S_2} \nabla \ell(\mathbf{w}_t, \mathbf{z}_j)$ .
6:   Sample  $\gamma \sim \text{Lap}(2\sigma)$ ,  $\tau \sim \text{Lap}(4\sigma)$ .
7:   if  $\|\hat{\mathbf{g}}_{S_1,t} - \hat{\mathbf{g}}_{S_2,t}\| + \gamma > \tau$  then
8:      $\tilde{\mathbf{g}}_t = \hat{\mathbf{g}}_{S_1,t} + \mathbf{b}_t$ , where  $\mathbf{b}_t^i$  is drawn i.i.d from  $\text{Lap}(\sigma)$ , for all  $i \in [d]$ .
9:   else
10:     $\tilde{\mathbf{g}}_t = \hat{\mathbf{g}}_{S_2,t}$ 
11:   end if
12:    $\mathbf{m}_t = \tilde{\mathbf{g}}_t$  and  $\mathbf{v}_t = (1 - \beta_2) \sum_{i=1}^t \beta_2^{t-i} \tilde{\mathbf{g}}_i^2$ .
13:    $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \mathbf{m}_t / (\sqrt{\mathbf{v}_t} + \nu)$ .
14: end for
15: Return:  $\tilde{\mathbf{g}}_t$ .

```

---

Following THRESHOLDOUT, Zhou et al. [40] propose a stable gradient descent algorithm which uses a similar framework as DPG-SPARSE to compute an estimated gradient by validating coordinates of  $\hat{\mathbf{g}}_{S_1,t}$  and  $\hat{\mathbf{g}}_{S_2,t}$ . However, their method is computationally expensive in high-dimensional settings such as deep neural networks. Ours are particularly suited for those models, as observed in Section 4.

**High-probability bound:** To analyze the privacy cost of DPG-SPARSE, let  $C_s$  be the number of times the validation fails, i.e.,  $\|\hat{\mathbf{g}}_{S_1,t} - \hat{\mathbf{g}}_{S_2,t}\| + \gamma > \tau$  is true, over  $T$  iterations in SAGD. The following Lemma establishes the privacy cost of the SAGD with DPG-SPARSE algorithm.

**Lemma 3.** SAGD with DPG-SPARSE (Alg. 2) is  $(\frac{\sqrt{C_s \ln(2/\delta) 2G_1}}{n\sigma}, \delta)$ -differentially private.

Lemma 3 shows that the privacy cost of SAGD with DPG-SPARSE scales with  $\mathcal{O}(\sqrt{C_s})$  where  $C_s \leq T$ . In other words, DPG-SPARSE procedure improves the privacy cost of the SAGD algorithm. Indeed, in order to achieve the generalization guarantee of SAGD with DPG-SPARSE, stated in Lemma 1 and by considering the result of Lemma 3, we only need to set  $\sqrt{C_s \ln(1/\delta)} G_1 / (n\sigma) \leq \sigma/13$ , which potentially improves the upper bound on  $T$ . We derive the generalization guarantee of  $\tilde{\mathbf{g}}_t$  generated by the SAGD with DPG-SPARSE algorithm in the following result:

**Theorem 3.** Given  $\sigma > 0$ , let  $\tilde{\mathbf{g}}_1, \dots, \tilde{\mathbf{g}}_T$  be the gradients computed by DPG-SPARSE in SAGD. With a budget  $n\sigma^2 / (2G_1^2) \leq C_s \leq n^2\sigma^4 / (676 \ln(1/(\sigma\beta)) G_1^2)$ , then for  $t \in [T], \beta > 0, \mu > 0$ :

$$\mathbb{P} \left\{ \|\tilde{\mathbf{g}}_t - \mathbf{g}_t\| \geq \sqrt{d}\sigma(1 + \mu) \right\} \leq d\beta + d \exp(-\mu).$$

In the worst case  $C_s = T$ , we recover the bound of  $T \leq n^2\sigma^4 / (676 \ln(1/(\sigma\beta)) G_1^2)$  of DPG-LAP.

**Non-asymptotic convergence rate:** The finite-time upper bound on the convergence criterion of interest for the SAGD with DPG-SPARSE algorithm (Algorithm 2) is stated as follows:



**Theorem 4.** Given training set  $S$  of size  $n$ , for  $\nu > 0$ , if  $\eta_t = \eta$  which are chosen with  $\eta \leq \nu/(2L)$ , noise level  $\sigma = 1/n^{1/3}$ , and iteration number  $T = n^{2/3}/(676G_1^2(\ln d + \frac{7}{3}\ln n))$ , then SAGD with DPG-SPARSE algorithm yields:

$$\min_{1 \leq t \leq T} \|\nabla f(\mathbf{w}_t)\|^2 \leq \mathcal{O}\left(\frac{\rho_{n,d}(f(\mathbf{w}_1) - f^*)}{n^{2/3}}\right) + \mathcal{O}\left(\frac{d\rho_{n,d}^2}{n^{2/3}}\right),$$

with probability at least  $1 - \mathcal{O}(1/(\rho_{n,d}n))$ .

Theorem 4 displays a similar rate of  $\mathcal{O}(1/n^{2/3})$  for the SAGD with DGP-SPARSE as Theorem 2. A sharper bound can be achieved when the number of validation failures  $C_s$  is smaller than  $T$ . For example, if  $C_s = \mathcal{O}(\sqrt{T})$ , the upper bound of  $T$  can be improved from  $T \leq \mathcal{O}(n^2)$  to  $T \leq \mathcal{O}(n^4)$ .

### 3.3 Mini-batch Stable Adaptive Gradient Descent Algorithm

For large-scale learning we derive the mini-batch variant of SAGD in Algorithm 3. The training set  $S$  is first partitioned into  $B$  batches with  $m$  samples for each batch. At each iteration  $t$ , Algorithm 3 uses any DPG procedure to compute a differential private gradient  $\tilde{\mathbf{g}}_t$  on each batch and updates  $\mathbf{w}_t$ .

---

#### Algorithm 3 Mini-Batch SAGD

---

```

1: Input: Dataset  $S$ , certain loss  $\ell(\cdot)$ , initial point  $\mathbf{w}_0$ .
2: Set noise level  $\sigma$ , epoch number  $T$ , batch size  $m$ , and stepsize  $\eta_t$ .
3: Split  $S$  into  $B = n/m$  batches:  $\{s_1, \dots, s_B\}$ .
4: for epoch = 1, ...,  $T$  do
5:   for  $k = 1, \dots, B$  do
6:     Call DPG-LAP or DPG-SPARSE to compute  $\tilde{\mathbf{g}}_t$ .
7:      $\mathbf{m}_t = \tilde{\mathbf{g}}_t$  and  $\mathbf{v}_t = (1 - \beta_2) \sum_{i=1}^t \beta_2^{t-i} \tilde{\mathbf{g}}_i^2$ .
8:      $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \mathbf{m}_t / (\sqrt{\mathbf{v}_t} + \nu)$ .
9:   end for
10: end for

```

---

**Theorem 5.** Consider the mini-batch SAGD with DPG-LAP. Given  $S$  of size  $n$ , with  $\nu > 0$ ,  $\eta_t = \eta \leq \nu/(2L)$ , noise level  $\sigma = 1/n^{1/3}$ , and epoch  $T = m^{4/3}/(n169G_1^2(\ln d + \frac{7}{3}\ln n))$ , then:

$$\min_{t=1, \dots, T} \|\nabla f(\mathbf{w}_t)\|^2 \leq \mathcal{O}\left(\frac{\rho_{n,d}(f(\mathbf{w}_1) - f^*)}{(mn)^{1/3}}\right) + \mathcal{O}\left(\frac{d\rho_{n,d}^2}{(mn)^{1/3}}\right),$$

with probability at least  $1 - \mathcal{O}(1/(\rho_{n,d}n))$ .

Theorem 5 describes the convergence rate of the mini-batch SAGD algorithm in terms of batch size  $m$  and sample size  $n$ , i.e.,  $\mathcal{O}(1/(mn)^{1/3})$ . When  $m = \sqrt{n}$ , mini-batch SAGD achieves the convergence of rate  $\mathcal{O}(1/\sqrt{n})$ . When  $m = n$ , i.e., in the full batch setting, Theorem 5 recovers SAGD's convergence rate  $\mathcal{O}(1/n^{2/3})$ . In terms of computational complexity, the mini-batch SAGD requires  $\mathcal{O}(m^{7/3}/n)$  stochastic gradient computations for  $\mathcal{O}(m^{4/3}/n)$  passes over  $m$  samples, while SAGD requires  $\mathcal{O}(n^{5/3})$  stochastic gradient computations. Thus, the mini-batch SAGD has the advantage of decreasing the computation complexity, but displays a slower convergence than SAGD.

## 4 Numerical Experiments

In this section, we evaluate our proposed mini-batch SAGD algorithm on various deep learning models against popular optimization methods: SGD with momentum [27], Adam [17], Padam [5], AdaGrad [8], RMSprop [33], and Adabound [22]. We consider three tasks: the classification tasks on MNIST [20] and CIFAR-10 [18], and the language modeling task on Penn Treebank [23]. The setup of each task is given in the following table:

Dataset	Network Type	Architectures
MNIST	Feedforward	2-Layer with ReLU and 2-Layer with Sigmoid
CIFAR-10	Deep Convolutional	VGG-19 and ResNet-18
Penn Treebank	Recurrent	2-Layer LSTM and 3-Layer LSTM

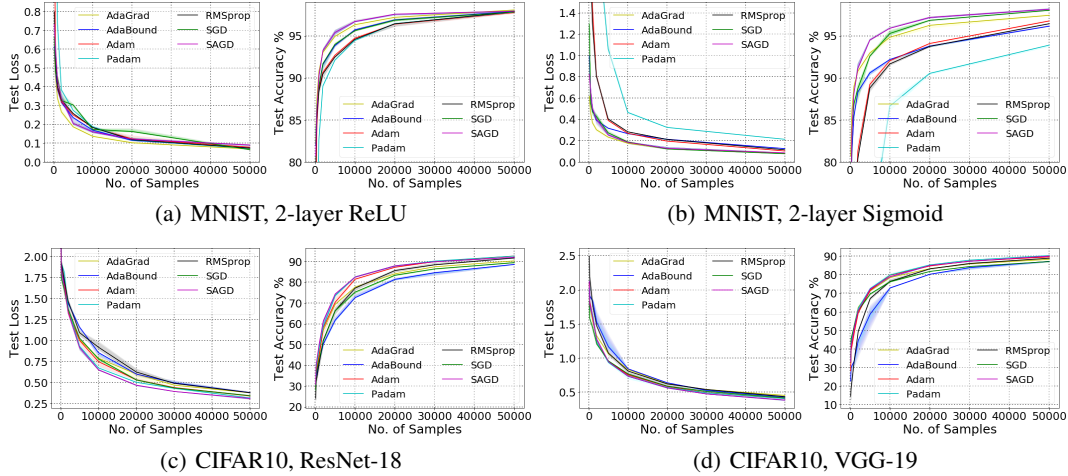


Figure 1: **Top row:** Test loss and accuracy of (a) ReLU neural network and (b) Sigmoid neural network on MNIST. The X-axis is the number of train samples, and the Y-axis is the loss/accuracy. In both cases, SAGD obtains the best test accuracy among all the methods. **Bottom row:** Test loss and accuracy of ResNet-18 and VGG-19 on CIFAR10. SAGD achieves the lowest test loss. For VGG-19, SAGD achieves the best test accuracy among all the methods.

## 4.1 Environmental Settings

**Datasets and Evaluation Metrics:** The MNIST dataset has a training set of 60000 examples and a test set of 10000 examples. The CIFAR-10 dataset consists of 50000 training images and 10000 test images. The Penn Treebank dataset contains 929589, 73760, and 82430 tokens for training, validation, and test, respectively. To better understand the generalization ability of each optimization algorithm with an increasing training sample size  $n$ , for each task, we construct multiple training sets of different size by sampling from the original training set. For MNIST, training sets of size  $n \in \{50, 100, 200, 500, 10^3, 2 \cdot 10^3, 5 \cdot 10^3, 10^4, 2 \cdot 10^4, 5 \cdot 10^4\}$  are constructed. For CIFAR10, training sets of size  $n \in \{200, 500, 10^3, 2 \cdot 10^3, 5 \cdot 10^3, 10^4, 2 \cdot 10^4, 3 \cdot 10^4, 5 \cdot 10^4\}$  are constructed. For each  $n$ , we train the model and report the loss and accuracy on the test set. For Penn Treebank, all training samples are used to train the model and we report the training perplexity and the test perplexity across epochs. Cross-entropy is used as the loss function throughout experiments. The mini-batch size is set to be 128 for CIFAR10 and MNIST, 20 for Penn Treebank. We repeat each experiment 5 times and report the mean and standard deviation of the results.

**Hyper-parameter setting:** Optimization hyper-parameters affect the quality of solutions. Particularly, Wilson et al. [37] highlight that the initial stepsize and the scheme of decaying stepsizes have a considerable impact on the performance. We follow the logarithmically-spaced grid method in Wilson et al. [37] to tune the stepsize. If the parameter performs best at an extreme end of the grid, a new grid will be tried until the best parameter lies in the middle of the grid. Once the interval of the best stepsize is located, we change to the linear-spaced grid to further search for the optimal one. We specify the strategy of decaying stepsizes in the subsections of each task. For each experiment, we set  $\sigma^2 = 1/n^{2/3}$ , where  $n$  is the size of the training set, as stated in Theorem 5. Parameters  $\nu$ ,  $\beta_2$ , and  $T$  follow the default settings as adaptive algorithms such as RMSprop.

## 4.2 Numerical results

**Feedforward Neural Network.** For image classification on MNIST, we focus on two 2-layer fully connected neural networks with either ReLU or Sigmoid activation functions. We run 100 epochs and decay the learning rate by 0.5 every 30 epochs. Figure 1 presents the loss and accuracy on the test set given different training set sizes. Since all algorithms attain the 100% training accuracy, the performance on the training set is omitted. Figure 1 (a) shows that, for ReLU neural network, SAGD performs slightly better than the other algorithms in terms of test accuracy. When  $n = 50000$ , SAGD gets a test accuracy of  $98.38 \pm 0.13\%$ . Figure 1 (b) presents the results on Sigmoid neural network where SAGD achieves the best test accuracy among all the algorithms. When  $n = 50000$ , SAGD reaches the highest test accuracy of  $98.14 \pm 0.11\%$ , outperforming other adaptive algorithms.

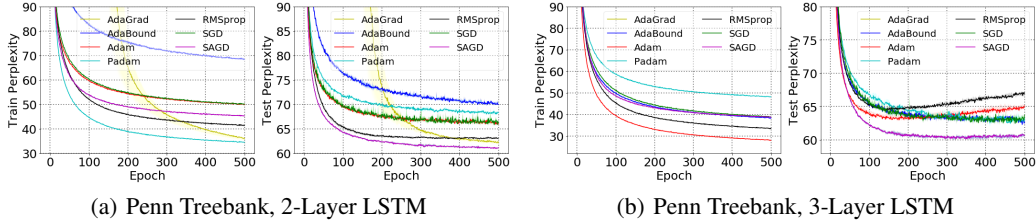


Figure 2: Train and test perplexity of 2-layer LSTM and 3-layer LSTM. Although adaptive methods such as AdGrad, Padam, Adam, and RMSprop achieves better training performance than SAGD, SAGD performs the best in terms of the test perplexity among all the methods.

**Convolutional Neural Network.** We use ResNet-18 [15] and VGG-19 [32] for the CIFAR-10 image classification task. We run 100 epochs and decay the learning rate by 0.1 every 30 epochs. The results are presented in Figure 1. Figure 1 (c) shows that SAGD has higher test accuracy than the other algorithms when the sample size is small *i.e.*,  $n \leq 20000$ . When  $n = 50000$ , SAGD achieves nearly the same test accuracy,  $92.48 \pm 0.09\%$ , as Adam, Padam, and RMSprop. Non-adaptive algorithm SGD performs better than the other algorithms in terms of test loss. Figure 1 (d) reports the results on VGG-19. Although SAGD has a higher test loss than the other algorithms, it achieves the best test accuracy, especially when  $n$  is small. Non-adaptive algorithm SGD performs better than the other adaptive gradient algorithms regarding the test accuracy. When  $n = 50000$ , SGD has the best test accuracy  $91.36 \pm 0.04\%$ . SAGD achieves accuracy  $91.26 \pm 0.05\%$ .

**Recurrent Neural Network.** Finally, an experiment on Penn Treebank is conducted for the language modeling task with 2-layer Long Short-Term Memory (LSTM) [24] network and 3-layer LSTM. We train them for a fixed budget of 500 epochs and omit the learning-rate decay. Perplexity is used as the metric to evaluate the performance and learning curves are plotted in Figure 2. Figure 2 (a) shows that for the 2-layer LSTM, AdaGrad, Padam, RMSprop and Adam achieve a lower training perplexity than SAGD. However, SAGD performs the best in terms of the test perplexity. Specifically, SAGD achieves  $61.02 \pm 0.08$  test perplexity. In particular, we observe that after 200 epochs, the test perplexity of AdaGrad and Adam starts increasing, but the training perplexity continues decreasing (over-fitting occurs). Figure 2 (b) reports the results for the 3-layer LSTM. We can see that the perplexity of AdaGrad, Padam, Adam, and RMSprop start increasing significantly after 150 epochs (*over-fitting*) while the perplexity of SAGD keeps decreasing. SAGD, SGD and AdaBounds perform better than AdaGrad, Padam, Adam, and RMSprop in terms of over-fitting. Table 1 shows the best test perplexity of 2-layer LSTM and 3-layer LSTM for all the algorithms. We can observe that the SAGD achieves the best test perplexity  $59.43 \pm 0.24$  among all the algorithms.

Table 1: Test Perplexity of LSTMs on Penn Treebank. Bold number indicates the best result.

	RMSprop	Adam	AdaGrad	Padam	AdaBound	SGD	SAGD
2-layer LSTM	$62.87 \pm 0.05$	$60.58 \pm 0.37$	$62.20 \pm 0.29$	$62.85 \pm 0.16$	$65.82 \pm 0.08$	$65.96 \pm 0.23$	<b><math>61.02 \pm 0.08</math></b>
3-layer LSTM	$63.97 \pm 0.18$	$63.23 \pm 0.04$	$66.25 \pm 0.31$	$66.45 \pm 0.28$	$62.33 \pm 0.07$	$62.51 \pm 0.11$	<b><math>59.43 \pm 0.24</math></b>

## 5 Conclusion

In this paper, we focus on the generalization ability of adaptive gradient methods. Concerned with the observation that adaptive gradient methods generalize worse than SGD for over-parameterized neural networks and given the limited theoretical understanding of the generalization of those methods, we propose **Stable Adaptive Gradient Descent (SAGD)** methods, which boost the generalization performance in both theory and practice through a novel use of differential privacy. The proposed algorithms generalize well with provable high-probability convergence bounds of the population gradient. Experimental studies highlight that the proposed algorithms are competitive and often better than baseline algorithms for training deep neural networks and demonstrate the aptitude of our method to avoid over-fitting through a differential privacy mechanism.



## 6 Broader Impact

We believe that our work stands in the line of several papers towards improving generalization and avoiding over-fitting. Indeed, the basic principle of our method is to fit any given model, in particular deep model, using an intermediate differentially-private mechanisms allowing the model to fit fresh samples while passing over the same batch of  $n$  observations. The impact of such work is straightforward and could avoid learning, and thus reproducing at testing phase, the bias existent in the training dataset.

## References

- [1] R. Bassily, A. Smith, and A. Thakurta. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 464–473. IEEE, 2014.
- [2] O. Bousquet and A. Elisseeff. Stability and generalization. *Journal of machine learning research*, 2(Mar):499–526, 2002.
- [3] S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*, 2015.
- [4] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12(Mar):1069–1109, 2011.
- [5] J. Chen and Q. Gu. Closing the generalization gap of adaptive gradient methods in training deep neural networks. *arXiv preprint arXiv:1806.06763*, 2018.
- [6] X. Chen, S. Liu, R. Sun, and M. Hong. On the convergence of a class of adam-type algorithms for non-convex optimization. In *International Conference on Learning Representations*, 2019.
- [7] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*, 2017.
- [8] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.
- [9] C. Dwork, A. Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.
- [10] C. Dwork, V. Feldman, M. Hardt, T. Pitassi, O. Reingold, and A. Roth. Generalization in adaptive data analysis and holdout reuse. In *Advances in Neural Information Processing Systems*, pages 2350–2358, 2015.
- [11] C. Dwork, V. Feldman, M. Hardt, T. Pitassi, O. Reingold, and A. Roth. The reusable holdout: Preserving validity in adaptive data analysis. *Science*, 349(6248):636–638, 2015.
- [12] C. Dwork, V. Feldman, M. Hardt, T. Pitassi, O. Reingold, and A. L. Roth. Preserving statistical validity in adaptive data analysis. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 117–126. ACM, 2015.
- [13] S. Ghadimi and G. Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.
- [14] M. Hardt, B. Recht, and Y. Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *International Conference on Machine Learning*, pages 1225–1234, 2016.
- [15] 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.

- 333 [16] N. S. Keskar and R. Socher. Improving generalization performance by switching from adam  
334 to sgd. *arXiv preprint arXiv:1712.07628*, 2017.
- 335 [17] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *In Proceedings of*  
336 *the 3rd International Conference on Learning Representations (ICLR)*, 2015.
- 337 [18] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. Tech-  
338 nical report, Citeseer, 2009.
- 339 [19] I. Kuzborskij and C. Lampert. Data-dependent stability of stochastic gradient descent. In  
340 *International Conference on Machine Learning*, pages 2820–2829, 2018.
- 341 [20] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, et al. Gradient-based learning applied to document  
342 recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- 343 [21] J. Li, X. Luo, and M. Qiao. On generalization error bounds of noisy gradient methods for  
344 non-convex learning. *arXiv preprint arXiv:1902.00621*, 2019.
- 345 [22] L. Luo, Y. Xiong, and Y. Liu. Adaptive gradient methods with dynamic bound of learning rate.  
346 In *International Conference on Learning Representations*, 2019.
- 347 [23] M. Marcus, B. Santorini, and M. A. Marcinkiewicz. Building a large annotated corpus of  
348 english: the penn treebank. *Computational linguistics-Association for Computational Lin-*  
349 *guistics*, 19(2):313–330, 1993.
- 350 [24] S. Merity, N. S. Keskar, and R. Socher. Regularizing and optimizing LSTM language models.  
351 In *International Conference on Learning Representations*, 2018.
- 352 [25] W. Mou, L. Wang, X. Zhai, and K. Zheng. Generalization bounds of sgld for non-convex  
353 learning: Two theoretical viewpoints. In *Conference On Learning Theory*, pages 605–638,  
354 2018.
- 355 [26] A. Pensia, V. Jog, and P.-L. Loh. Generalization error bounds for noisy, iterative algorithms.  
356 In *2018 IEEE International Symposium on Information Theory (ISIT)*, pages 546–550. IEEE,  
357 2018.
- 358 [27] N. Qian. On the momentum term in gradient descent learning algorithms. *Neural networks*,  
359 12(1):145–151, 1999.
- 360 [28] M. Raginsky, A. Rakhlin, and M. Telgarsky. Non-convex learning via stochastic gradient  
361 langevin dynamics: a nonasymptotic analysis. In *Conference on Learning Theory*, pages 1674–  
362 1703, 2017.
- 363 [29] S. J. Reddi, S. Kale, and S. Kumar. On the convergence of adam and beyond. In *International*  
364 *Conference on Learning Representations*, 2018.
- 365 [30] H. Robbins and S. Monro. A stochastic approximation method. *The annals of mathematical*  
366 *statistics*, pages 400–407, 1951.
- 367 [31] S. Shalev-Shwartz and S. Ben-David. *Understanding machine learning: From theory to algo-*  
368 *rithms*. Cambridge university press, 2014.
- 369 [32] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recog-  
370 nition. *arXiv preprint arXiv:1409.1556*, 2014.
- 371 [33] T. Tieleman and G. Hinton. Rmsprop: Divide the gradient by a running average of its recent  
372 magnitude. *COURSERA: Neural networks for machine learning*, 2012.
- 373 [34] D. Wang and J. Xu. Differentially private empirical risk minimization with smooth non-convex  
374 loss functions: A non-stationary view. In *Proceedings of the AAAI Conference on Artificial*  
375 *Intelligence*, volume 33, pages 1182–1189, 2019.

- 376 [35] D. Wang, M. Ye, and J. Xu. Differentially private empirical risk minimization revisited: Faster  
377 and more general. In *Advances in Neural Information Processing Systems*, pages 2722–2731,  
378 2017.
- 379 [36] R. Ward, X. Wu, and L. Bottou. Adagrad stepsizes: sharp convergence over nonconvex land-  
380 scapes. In *International Conference on Machine Learning*, pages 6677–6686, 2019.
- 381 [37] A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, and B. Recht. The marginal value of adaptive  
382 gradient methods in machine learning. In *Advances in Neural Information Processing Systems*,  
383 pages 4148–4158, 2017.
- 384 [38] M. Zaheer, S. Reddi, D. Sachan, S. Kale, and S. Kumar. Adaptive methods for nonconvex  
385 optimization. In *Advances in Neural Information Processing Systems*, pages 9793–9803, 2018.
- 386 [39] D. Zhou, Y. Tang, Z. Yang, Y. Cao, and Q. Gu. On the convergence of adaptive gradient  
387 methods for nonconvex optimization. *arXiv preprint arXiv:1808.05671*, 2018.
- 388 [40] Y. Zhou, S. Chen, and A. Banerjee. Stable gradient descent. In *UAI*, pages 766–775, 2018.
- 389 [41] F. Zou, L. Shen, Z. Jie, W. Zhang, and W. Liu. A sufficient condition for convergences of  
390 adam and rmsprop. In *Proceedings of the IEEE Conference on Computer Vision and Pattern*  
391 *Recognition*, pages 11127–11135, 2019.

## A Differential Privacy and Generalization Analysis

### A.1 Proof of Lemma 1

By applying Theorem 8 from Dwork et al. [10] to gradient computation, we can get the Lemma 1.

**Lemma 1.** *Let  $\mathcal{A}$  be an  $(\epsilon, \delta)$ -differentially private gradient descent algorithm with access to training set  $S$  of size  $n$ . Let  $\mathbf{w}_t = \mathcal{A}(S)$  be the parameter generated at iteration  $t \in [T]$  and  $\hat{\mathbf{g}}_t$  the empirical gradient on  $S$ . For any  $\sigma > 0$ ,  $\beta > 0$ , if the privacy cost of  $\mathcal{A}$  satisfies  $\epsilon \leq \sigma/13$ ,  $\delta \leq \sigma\beta/(26 \ln(26/\sigma))$ , and sample size  $n \geq 2 \ln(8/\delta)/\epsilon^2$ , we then have*

$$\mathbb{P} \{ |\hat{\mathbf{g}}_t^i - \mathbf{g}_t^i| \geq \sigma \} \leq \beta \quad \text{for every } i \in [d] \text{ and every } t \in [T].$$

**Proof** Theorem 8 in Dwork et al. [10] shows that in order to achieve generalization error  $\tau$  with probability  $1 - \rho$  for a  $(\epsilon, \delta)$ -differentially private algorithm (i.e., in order to guarantee for every function  $\phi_t$ ,  $\forall t \in [T]$ , we have  $\mathbb{P} [|\mathcal{P}[\phi_t] - \mathcal{E}_S[\phi_t]| \geq \tau] \leq \rho$ ), where  $\mathcal{P}[\phi_t]$  is the population value,  $\mathcal{E}_S[\phi_t]$  is the empirical value evaluated on  $S$  and  $\rho$  and  $\tau$  are any positive constant, we can set the  $\epsilon \leq \frac{\tau}{13}$  and  $\delta \leq \frac{\tau\rho}{26 \ln(26/\tau)}$ . In our context,  $\tau = \sigma$ ,  $\beta = \rho$ ,  $\phi_t$  is the gradient computation function  $\nabla \ell(\mathbf{w}_t, \mathbf{z})$ ,  $\mathcal{P}[\phi_t]$  represents the population gradient  $\mathbf{g}_t^i$ ,  $\forall i \in [p]$ , and  $\mathcal{E}_S[\phi_t]$  represents the sample gradient  $\hat{\mathbf{g}}_t^i$ ,  $\forall i \in [p]$ . Thus we have  $\mathbb{P} \{ |\hat{\mathbf{g}}_t^i - \mathbf{g}_t^i| \geq \tau \} \leq \rho$  if  $\epsilon \leq \frac{\sigma}{13}$ ,  $\delta \leq \frac{\sigma\beta}{26 \ln(26/\sigma)}$ .

### A.2 Proof of Lemma 2

**Lemma 2.** *SAGD with DPG-LAP (Alg. 1) is  $(\frac{\sqrt{T \ln(1/\delta)} G_1}{n\sigma}, \delta)$ -differentially private.*

**Proof** At each iteration  $t$ , the algorithm is composed of two sequential parts: DPG to access the training set  $S$  and compute  $\tilde{\mathbf{g}}_t$ , and parameter update based on estimated  $\tilde{\mathbf{g}}_t$ . We mark the DPG as part  $\mathcal{A}$  and the gradient descent as part  $\mathcal{B}$ . We first show  $\mathcal{A}$  preserves  $\frac{G_1}{n\sigma}$ -differential privacy. Then according to the *post-processing property* of differential privacy (Proposition 2.1 in [9]) we have  $\mathcal{B} \circ \mathcal{A}$  is also  $\frac{G_1}{n\sigma}$ -differentially private.

The part  $\mathcal{A}$  (DPG-Lap) uses the basic tool from differential privacy, the ‘‘Laplace Mechanism’’ (Definition 3.3 in [9]). The Laplace Mechanism adds i.i.d. Laplace noise to each coordinate of the output. Adding noise from  $\text{Lap}(\sigma)$  to a query of  $G_1/n$  sensitivity preserves  $G_1/n\sigma$ -differential privacy by (Theorem 3.6 in [9]). Over  $T$  iterations, we have  $T$  applications of a DPG-Lap. By the advanced composition theorem (Theorem 3.20 in [9]),  $T$  applications of a  $\frac{G_1}{n\sigma}$ -differentially private algorithm is  $(\frac{\sqrt{T \ln(1/\delta)} G_1}{n\sigma}, \delta)$ -differentially private. So SAGD with DPG-Lap is  $(\frac{\sqrt{T \ln(1/\delta)} 2G_1}{n\sigma}, \delta)$ -differentially private.  $\square$

### A.3 Proof of Theorem 1

**Theorem 1.** *Given  $\sigma > 0$ , let  $\tilde{\mathbf{g}}_1, \dots, \tilde{\mathbf{g}}_T$  be gradients computed by DPG-LAP in SAGD. Set the number of iterations  $2n\sigma^2/G_1^2 \leq T \leq n^2\sigma^4/(169 \ln(1/(\sigma\beta))G_1^2)$ , then for  $t \in [T]$ ,  $\beta > 0$ ,  $\mu > 0$ :*

$$\mathbb{P} \{ \|\tilde{\mathbf{g}}_t - \mathbf{g}_t\| \geq \sqrt{d}\sigma(1 + \mu) \} \leq d\beta + d \exp(-\mu).$$

**Proof** The concentration bound is decomposed into two parts:

$$\mathbb{P} \{ \|\tilde{\mathbf{g}}_t - \mathbf{g}_t\| \geq \sqrt{d}\sigma(1 + \mu) \} \leq \underbrace{\mathbb{P} \{ \|\tilde{\mathbf{g}}_t - \hat{\mathbf{g}}_t\| \geq \sqrt{d}\sigma\mu \}}_{T_1: \text{empirical error}} + \underbrace{\mathbb{P} \{ \|\hat{\mathbf{g}}_t - \mathbf{g}_t\| \geq \sqrt{d}\sigma \}}_{T_2: \text{generalization error}}.$$

In the above inequality, there are two types of error we need to control. The first type of error, referred to as empirical error  $T_1$ , is the deviation between the differentially private estimated gradient  $\tilde{\mathbf{g}}_t$  and the empirical gradient  $\hat{\mathbf{g}}_t$ . The second type of error, referred to as generalization error  $T_2$ , is the deviation between the empirical gradient  $\hat{\mathbf{g}}_t$  and the population gradient  $\mathbf{g}_t$ .

The second term  $T_2$  can be bounded thorough the generalization guarantee of differential privacy. Recall that from Lemma 1, under the condition in Theorem 3, we have for all  $t \in [T]$ ,  $i \in [d]$ :

$$\mathbb{P} \{ |\hat{\mathbf{g}}_t^i - \mathbf{g}_t^i| \geq \sigma \} \leq \beta.$$

430 So that we have

$$\mathbb{P} \left\{ \|\hat{\mathbf{g}}_t - \mathbf{g}_t\| \geq \sqrt{d}\sigma \right\} \leq \mathbb{P} \left\{ \|\hat{\mathbf{g}}_t - \mathbf{g}_t\|_\infty \geq \sigma \right\} \leq d\mathbb{P} \left\{ |\hat{g}_t^i - g_t^i| \geq \sigma \right\} \leq d\beta. \quad (3)$$

431 Now we bound the second term  $T_1$ . Recall that  $\tilde{\mathbf{g}}_t = \hat{\mathbf{g}}_t + \mathbf{b}_t$ , where  $\mathbf{b}_t$  is a noise vector with each  
 432 coordinate drawn from Laplace noise  $\text{Lap}(\sigma)$ . In this case, we have

$$\mathbb{P} \left\{ \|\tilde{\mathbf{g}}_t - \hat{\mathbf{g}}_t\| \geq \sqrt{d}\sigma\mu \right\} \leq \mathbb{P} \left\{ \|\mathbf{b}_t\| \geq \sqrt{d}\sigma\mu \right\} \leq \mathbb{P} \left\{ \|\mathbf{b}_t\|_\infty \geq \sigma\mu \right\} \quad (4)$$

$$\leq d\mathbb{P} \left\{ |\mathbf{b}_t^i| \geq \sigma\mu \right\} = d\exp(-\mu). \quad (5)$$

433 The second inequality comes from  $\|\mathbf{b}_t\| \leq \sqrt{d}\|\mathbf{b}_t\|_\infty$ . The last equality comes from the property  
 434 of Laplace distribution. Combine (3) and (4), we complete the proof.  $\square$

#### 435 A.4 Proof of Lemma 3

436 **Lemma 3.** SAGD with DPG-SPARSE (Alg. 2) is  $(\frac{\sqrt{C_s \ln(2/\delta)2G_1}}{n\sigma}, \delta)$ -differentially private.

437 **Proof** At each iteration  $t$ , the algorithm is composed of two sequential parts: DPG-Sparse (part  $\mathcal{A}$ )  
 438 and parameter update based on estimated  $\tilde{\mathbf{g}}_t$  (part  $\mathcal{B}$ ). We first show  $\mathcal{A}$  preserves  $\frac{2G_1}{n\sigma}$ -differential  
 439 privacy. Then according to the *post-processing property* of differential privacy (Proposition 2.1  
 440 in [9]) we have  $\mathcal{B} \circ \mathcal{A}$  is also  $\frac{2G_1}{n\sigma}$ -differentially private.

441 The part  $\mathcal{A}$  (DPG-Sparse) is a composition of basic tools from differential privacy, the ‘‘Sparse  
 442 Vector Algorithm’’ (Algorithm 2 in [9]) and the ‘‘Laplace Mechanism’’ (Definition 3.3 in [9]). In  
 443 our setting, the sparse vector algorithm takes as input a sequence of  $T$  sensitivity  $G_1/n$  queries,  
 444 and for each query, attempts to determine whether the value of the query, evaluated on the private  
 445 dataset  $S_1$ , is above a fixed threshold  $\gamma + \tau$  or below it. In our instantiation, the  $S_1$  is the private data  
 446 set, and each function corresponds to the gradient computation function  $\hat{\mathbf{g}}_t$  which is of sensitivity  
 447  $G_1/n$ . By the privacy guarantee of the sparse vector algorithm, the sparse vector portion of SAGD  
 448 satisfies  $G_1/n\sigma$ -differential privacy. The Laplace mechanism portion of SAGD satisfies  $G_1/n\sigma$ -  
 449 differential privacy by (Theorem 3.6 in [9]). Finally, the composition of two mechanisms satisfies  
 450  $\frac{2G_1}{n\sigma}$ -differential privacy. For the sparse vector technique, only the query that fails the validation,  
 451 corresponding to the ‘above threshold’, release the privacy of private dataset  $S_1$  and pays a  $\frac{2G_1}{n\sigma}$   
 452 privacy cost. Over all the iterations  $T$ , We have  $C_s$  queries fail the validation. Thus, by the advanced  
 453 composition theorem (Theorem 3.20 in [9]),  $C_s$  applications of a  $\frac{2G_1}{n\sigma}$ -differentially private algorithm  
 454 is  $(\frac{\sqrt{C_s \ln(2/\delta)2G_1}}{n\sigma}, \delta)$ -differentially private. So SAGD with DPG-Sparse is  $(\frac{\sqrt{C_s \ln(2/\delta)2G_1}}{n\sigma}, \delta)$ -  
 455 differentially private.  $\square$

#### 456 A.5 Proof of Theorem 3:

457 **Theorem 3.** Given  $\sigma > 0$ , let  $\tilde{\mathbf{g}}_1, \dots, \tilde{\mathbf{g}}_T$  be the gradients computed by DPG-SPARSE in SAGD.  
 458 With a budget  $n\sigma^2/(2G_1^2) \leq C_s \leq n^2\sigma^4/(676 \ln(1/(\sigma\beta))G_1^2)$ , then for  $t \in [T], \beta > 0, \mu > 0$ :

$$\mathbb{P} \left\{ \|\tilde{\mathbf{g}}_t - \mathbf{g}_t\| \geq \sqrt{d}\sigma(1 + \mu) \right\} \leq d\beta + d\exp(-\mu).$$

459 **Proof** The concentration bound can be decomposed into two parts:

$$\mathbb{P} \left\{ \|\tilde{\mathbf{g}}_t - \mathbf{g}_t\| \geq \sqrt{d}\sigma(1 + \mu) \right\} \leq \underbrace{\mathbb{P} \left\{ \|\tilde{\mathbf{g}}_t - \hat{\mathbf{g}}_{s_1,t}\| \geq \sqrt{d}\sigma\mu \right\}}_{T_1: \text{empirical error}} + \underbrace{\mathbb{P} \left\{ \|\hat{\mathbf{g}}_{s_1,t} - \mathbf{g}_t\| \geq \sqrt{d}\sigma \right\}}_{T_2: \text{generalization error}},$$

460 which yields

$$\mathbb{P} \left\{ \|\hat{\mathbf{g}}_{s_1,t} - \mathbf{g}_t\| \geq \sqrt{d}\sigma \right\} \leq \mathbb{P} \left\{ \|\hat{\mathbf{g}}_{s_1,t} - \mathbf{g}_t\|_\infty \geq \sigma \right\} \leq d\mathbb{P} \left\{ |\hat{g}_{s_1,t}^i - g_t^i| \geq \sigma \right\} \leq d\beta. \quad (6)$$



461 Now we bound the second term  $T_1$  by considering two cases, by depending on whether DPG-3  
 462 answers the query  $\tilde{\mathbf{g}}_t$  by returning  $\tilde{\mathbf{g}}_t = \hat{\mathbf{g}}_{s_1,t} + \mathbf{v}_t$  or by returning  $\tilde{\mathbf{g}}_t = \hat{\mathbf{g}}_{s_2,t}$ . In the first case, we  
 463 have

$$\|\tilde{\mathbf{g}}_t - \hat{\mathbf{g}}_{s_1,t}\| = \|\mathbf{v}_t\|$$

464 and

$$\mathbb{P}\left\{\|\tilde{\mathbf{g}}_t - \hat{\mathbf{g}}_{s_1,t}\| \geq \sqrt{d}\sigma\mu\right\} = \mathbb{P}\left\{\|\mathbf{v}_t\| \geq \sqrt{d}\sigma\mu\right\} \leq d \exp(-\mu).$$

465 The last inequality comes from the  $\|\mathbf{v}_t\| \leq \sqrt{d}\|\mathbf{v}_t\|_\infty$  and properties of the Laplace distribution.

466 In the second case, we have

$$\|\tilde{\mathbf{g}}_t - \hat{\mathbf{g}}_{s_1,t}\| = \|\hat{\mathbf{g}}_{s_2,t} - \hat{\mathbf{g}}_{s_1,t}\| \leq |\gamma| + |\tau|$$

467 and

$$\begin{aligned} \mathbb{P}\left\{\|\tilde{\mathbf{g}}_t - \hat{\mathbf{g}}_{s_1,t}\| \geq \sqrt{d}\sigma\mu\right\} &= \mathbb{P}\left\{|\gamma| + |\tau| \geq \sqrt{d}\sigma\mu\right\} \\ &\leq \mathbb{P}\left\{|\gamma| \geq \frac{2}{6}\sqrt{d}\sigma\mu\right\} + \mathbb{P}\left\{|\tau| \geq \frac{4}{6}\sqrt{d}\sigma\mu\right\} \\ &= 2 \exp(-\sqrt{d}\mu/6). \end{aligned}$$

468 Combining these two cases, we have

$$\begin{aligned} \mathbb{P}\left\{\|\tilde{\mathbf{g}}_t - \hat{\mathbf{g}}_{s_1,t}\| \geq \sqrt{d}\sigma\mu\right\} &\leq \max\left\{\mathbb{P}\left\{\|\mathbf{v}_t\| \geq \sqrt{d}\sigma\mu\right\}, \mathbb{P}\left\{|\gamma| + |\tau| \geq \sqrt{d}\sigma\mu\right\}\right\} \\ &\leq \max\left\{d \exp(-\mu), 2 \exp(-\sqrt{d}\mu/6)\right\} \\ &= d \exp(-\mu). \end{aligned} \tag{7}$$

469 We complete the proof by combining (6) and (7).

470 □

## 471 B Non-asymptotic Convergence analysis

472 In this section, we present the proof of Theorem 2, 4, 5.

### 473 B.1 Proof of Theorem 2 and Theorem 4

474 The proof of Theorem 2 consists of two parts: We first prove that the convergence rate of a gradient-  
 475 based iterative algorithm is related to the gradient concentration error  $\alpha$  and its iteration time  $T$ .  
 476 Then we combine the concentration error  $\alpha$  achieved by SAGD with DPG-Lap in Theorem 1 with  
 477 the first part to complete the proof of Theorem 2. To simplify the analysis, we first use  $\alpha$  and  $\xi$  to  
 478 denote the generalization error  $\sqrt{d}\sigma(1 + \mu)$  and probability  $d\beta + d \exp(-\mu)$  in Theorem 1 in the  
 479 following analysis. The details are presented in the following theorem.

480 **Theorem 6.** *Let  $\tilde{\mathbf{g}}_1, \dots, \tilde{\mathbf{g}}_T$  be the noisy gradients generated in Algorithm 1 through DPG oracle*  
 481 *over  $T$  iterations. Then, for every  $t \in [T]$ ,  $\tilde{\mathbf{g}}_t$  satisfies*

$$\mathbb{P}\{\|\tilde{\mathbf{g}}_t - \mathbf{g}_t\| \geq \alpha\} \leq \xi,$$

482 where the values of  $\alpha$  and  $\xi$  are given in Section A.

483 With the guarantee of Theorem 6, we have the following theorem showing the convergence of  
 484 SAGD.

485 **Theorem 7.** let  $\eta_t = \eta$ . Further more assume that  $\nu$ ,  $\beta$  and  $\eta$  are chosen such that the following  
486 conditions satisfied:  $\eta \leq \frac{\nu}{2L}$ . Under the Assumption A1 and A2, the Algorithm 1 with  $T$  iterations,  
487  $\phi_t(\tilde{\mathbf{g}}_1, \dots, \tilde{\mathbf{g}}_t) = \tilde{\mathbf{g}}_t$  and  $\mathbf{v}_t = (1 - \beta_2) \sum_{i=1}^t \beta_2^{t-i} \tilde{\mathbf{g}}_i^2$  achieves:

$$\min_{t=1, \dots, T} \|\nabla f(x_t)\|^2 \leq (G + \nu) \times \left( \frac{f(\mathbf{w}_1) - f^*}{\eta T} + \frac{3\alpha^2}{4\nu} \right), \quad (8)$$

488 with probability at least  $1 - T\xi$ .

489 We can now tackle the proof of our result stated in Theorem 7.

490 **Proof** Using the update rule of RMSprop, we have  $\phi_t(\tilde{\mathbf{g}}_1, \dots, \tilde{\mathbf{g}}_t) = \tilde{\mathbf{g}}_t$  and  $\psi_t(\tilde{\mathbf{g}}_1, \dots, \tilde{\mathbf{g}}_t) =$   
491  $(1 - \beta_2) \sum_{i=1}^t \beta_2^{t-i} \tilde{\mathbf{g}}_i^2$ . Thus, we can rewrite the update of Algorithm 1 as:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \tilde{\mathbf{g}}_t / (\sqrt{\mathbf{v}_t} + \nu) \text{ and } \mathbf{v}_t = (1 - \beta_2) \sum_{i=1}^t \beta_2^{t-i} \tilde{\mathbf{g}}_i^2.$$

492 Let  $\Delta_t = \tilde{\mathbf{g}}_t - \mathbf{g}_t$ , we obtain:

$$\begin{aligned} & f(\mathbf{w}_{t+1}) \\ & \leq f(\mathbf{w}_t) + \langle \mathbf{g}_t, \mathbf{w}_{t+1} - \mathbf{w}_t \rangle + \frac{L}{2} \|\mathbf{w}_{t+1} - \mathbf{w}_t\|^2 \\ & = f(\mathbf{w}_t) - \eta_t \langle \mathbf{g}_t, \tilde{\mathbf{g}}_t / (\sqrt{\mathbf{v}_t} + \nu) \rangle + \frac{L\eta_t^2}{2} \left\| \frac{\tilde{\mathbf{g}}_t}{(\sqrt{\mathbf{v}_t} + \nu)} \right\|^2 \\ & = f(\mathbf{w}_t) - \eta_t \left\langle \mathbf{g}_t, \frac{\mathbf{g}_t + \Delta_t}{\sqrt{\mathbf{v}_t} + \nu} \right\rangle + \frac{L\eta_t^2}{2} \left\| \frac{\mathbf{g}_t + \Delta_t}{\sqrt{\mathbf{v}_t} + \nu} \right\|^2 \\ & \leq f(\mathbf{w}_t) - \eta_t \left\langle \mathbf{g}_t, \frac{\mathbf{g}_t}{\sqrt{\mathbf{v}_t} + \nu} \right\rangle - \eta_t \left\langle \mathbf{g}_t, \frac{\Delta_t}{\sqrt{\mathbf{v}_t} + \nu} \right\rangle + L\eta_t^2 \left( \left\| \frac{\mathbf{g}_t}{\sqrt{\mathbf{v}_t} + \nu} \right\|^2 + \left\| \frac{\Delta_t}{\sqrt{\mathbf{v}_t} + \nu} \right\|^2 \right) \\ & = f(\mathbf{w}_t) - \eta_t \sum_{i=1}^d \frac{[\mathbf{g}_t]_i^2}{\sqrt{\mathbf{v}_t^i} + \nu} - \eta_t \sum_{i=1}^d \frac{\mathbf{g}_t^i \Delta_t^i}{\sqrt{\mathbf{v}_t^i} + \nu} + L\eta_t^2 \left( \sum_{i=1}^d \frac{[\mathbf{g}_t]_i^2}{(\sqrt{\mathbf{v}_t^i} + \nu)^2} + \sum_{i=1}^d \frac{[\Delta_t]_i^2}{(\sqrt{\mathbf{v}_t^i} + \nu)^2} \right) \\ & \leq f(\mathbf{w}_t) - \eta_t \sum_{i=1}^d \frac{[\mathbf{g}_t]_i^2}{\sqrt{\mathbf{v}_t^i} + \nu} + \frac{\eta_t}{2} \sum_{i=1}^d \frac{[\mathbf{g}_t]_i^2 + [\Delta_t]_i^2}{\sqrt{\mathbf{v}_t^i} + \nu} + \frac{L\eta_t^2}{\nu} \left( \sum_{i=1}^d \frac{[\mathbf{g}_t]_i^2}{\sqrt{\mathbf{v}_t^i} + \nu} + \sum_{i=1}^d \frac{[\Delta_t]_i^2}{\sqrt{\mathbf{v}_t^i} + \nu} \right) \\ & = f(\mathbf{w}_t) - \left( \eta_t - \frac{\eta_t}{2} - \frac{L\eta_t^2}{\nu} \right) \sum_{i=1}^d \frac{[\mathbf{g}_t]_i^2}{\sqrt{\mathbf{v}_t^i} + \nu} + \left( \frac{\eta_t}{2} + \frac{L\eta_t^2}{\nu} \right) \sum_{i=1}^d \frac{[\Delta_t]_i^2}{\sqrt{\mathbf{v}_t^i} + \nu}. \end{aligned}$$

493 Given the parameter setting from the theorem, we see the following condition hold:

$$\frac{L\eta_t}{\nu} \leq \frac{1}{4}.$$

494 Then we obtain

$$\begin{aligned} f(\mathbf{w}_{t+1}) & \leq f(\mathbf{w}_t) - \frac{\eta}{4} \sum_{i=1}^d \frac{[\mathbf{g}_t]_i^2}{\sqrt{\mathbf{v}_t^i} + \nu} + \frac{3\eta}{4} \sum_{i=1}^d \frac{[\Delta_t]_i^2}{\sqrt{\mathbf{v}_t^i} + \nu} \\ & \leq f(\mathbf{w}_t) - \frac{\eta}{G + \nu} \|\mathbf{g}_t\|^2 + \frac{3\eta}{4\epsilon} \|\Delta_t\|^2. \end{aligned}$$

495 The second inequality follows from the fact that  $0 \leq \mathbf{v}_t^i \leq G^2$ . Using the telescoping sum and  
496 rearranging the inequality, we obtain

$$\frac{\eta}{G + \nu} \sum_{t=1}^T \|\mathbf{g}_t\|^2 \leq f(\mathbf{w}_1) - f^* + \frac{3\eta}{4\epsilon} \sum_{t=1}^T \|\Delta_t\|^2.$$

497 Multiplying with  $\frac{G+\nu}{\eta T}$  on both sides and with the guarantee in Theorem 1 that  $\|\Delta_t\| \leq \alpha$  with  
 498 probability at least  $1 - \xi$ , we obtain

$$\min_{t=1,\dots,T} \|\mathbf{g}_t\|^2 \leq (G + \nu) \times \left( \frac{f(\mathbf{w}_1) - f^*}{\eta T} + \frac{3\alpha^2}{4\nu} \right),$$

499 with probability at least  $1 - T\xi$ .

500

501

□

502 We may now present the proof of our Theorem 2.

503 **Theorem 2.** *Given training set  $S$  of size  $n$ , for  $\nu > 0$ , if  $\eta_t = \eta$  with  $\eta \leq \nu/(2L)$ ,  $\sigma = 1/n^{1/3}$ ,  
 504 iteration number  $T = n^{2/3}/(169G_1^2(\ln d + 7\ln n/3))$ ,  $\mu = \ln(1/\beta)$  and  $\beta = 1/(dn^{5/3})$ , then  
 505 SAGD with DPG-LAP algorithm yields:*

$$\min_{1 \leq t \leq T} \|\nabla f(\mathbf{w}_t)\|^2 \leq \mathcal{O}\left(\frac{\rho_{n,d}(f(\mathbf{w}_1) - f^*)}{n^{2/3}}\right) + \mathcal{O}\left(\frac{d\rho_{n,d}^2}{n^{2/3}}\right),$$

506 with probability at least  $1 - \mathcal{O}(1/(\rho_{n,d}n))$ .

507 **Proof** First consider the gradient concentration bound achieved by SAGD (Theorem 1 and Theorem  
 508 3) that if  $\frac{2n\sigma^2}{G_1^2} \leq T \leq \frac{n^2\sigma^4}{169\ln(1/(\sigma\beta))G_1^2}$ , we have

$$\mathbb{P}\left\{\|\tilde{\mathbf{g}}_t - \mathbf{g}_t\| \geq \sqrt{d}\sigma(1 + \mu)\right\} \leq d\beta + d\exp(-\mu), \quad \forall t \in [T].$$

509 Then bring the setting in Theorem 2 that  $\sigma = 1/n^{1/3}$ , let  $\mu = \ln(1/\beta)$  and  $\beta = 1/(dn^{5/3})$ , we have  
 510

$$\|\tilde{\mathbf{g}}_t - \mathbf{g}_t\|^2 \leq d(1 + \ln d + \frac{5}{3}\ln n)^2/n^{2/3},$$

511 with probability at least  $1 - 1/n^{5/3}$ , when we set  $T = n^{2/3}/(169G_1^2(\ln d + \frac{7}{3}\ln n))$ .

512 Connect this result with Theorem 7, so that we have  $\alpha^2 = d(1 + \ln d + \frac{5}{3}\ln n)^2/n^{2/3}$  and  $\xi = 1/n^{5/3}$ .  
 513 Bring the value  $\alpha^2$ ,  $\xi$  and  $T = n^{2/3}/(169G_1^2(\ln d + \frac{7}{3}\ln n))$  into (8), with  $\rho_{n,d} = \mathcal{O}(\ln n + \ln d)$ ,  
 514 we have

$$\min_{t=1,\dots,T} \|\nabla f(\mathbf{w}_t)\|^2 \leq \mathcal{O}\left(\frac{\rho_{n,d}(f(\mathbf{w}_1) - f^*)}{n^{2/3}}\right) + \mathcal{O}\left(\frac{d\rho_{n,d}^2}{n^{2/3}}\right),$$

515 with probability at least  $1 - \mathcal{O}\left(\frac{1}{\rho_{n,d}n}\right)$  which concludes the proof. □

516 **Theorem 4.** *Given training set  $S$  of size  $n$ , for  $\nu > 0$ , if  $\eta_t = \eta$  which are chosen with  $\eta \leq \nu/(2L)$ ,  
 517 noise level  $\sigma = 1/n^{1/3}$ , and iteration number  $T = n^{2/3}/(676G_1^2(\ln d + \frac{7}{3}\ln n))$ , then SAGD with  
 518 DPG-SPARSE algorithm yields:*

$$\min_{1 \leq t \leq T} \|\nabla f(\mathbf{w}_t)\|^2 \leq \mathcal{O}\left(\frac{\rho_{n,d}(f(\mathbf{w}_1) - f^*)}{n^{2/3}}\right) + \mathcal{O}\left(\frac{d\rho_{n,d}^2}{n^{2/3}}\right),$$

519 with probability at least  $1 - \mathcal{O}(1/(\rho_{n,d}n))$ .

520 **Proof** The proof of Theorem 4 follows the proof of Theorem 2 by considering the case  $C_s = T$ . □

## 521 B.2 Proof of Theorem 5

522 **Theorem 5.** Consider the mini-batch SAGD with DPG-LAP. Given  $S$  of size  $n$ , with  $\nu > 0$ ,  
 523  $\eta_t = \eta \leq \nu/(2L)$ , noise level  $\sigma = 1/n^{1/3}$ , and epoch  $T = m^{4/3}/(n169G_1^2(\ln d + \frac{7}{3}\ln n))$ , then:

$$\min_{t=1,\dots,T} \|\nabla f(\mathbf{w}_t)\|^2 \leq \mathcal{O}\left(\frac{\rho_{n,d}(f(\mathbf{w}_1) - f^*)}{(mn)^{1/3}}\right) + \mathcal{O}\left(\frac{d\rho_{n,d}^2}{(mn)^{1/3}}\right),$$

524 with probability at least  $1 - \mathcal{O}(1/(\rho_{n,d}n))$ .

525 **Proof** When mini-batch SAGD calls **DPG** to access each batch  $s_k$  with size  $m$  for  $T$  times, we  
 526 have mini-batch SAGD preserves  $(\frac{\sqrt{T\ln(1/\delta)}G_1}{m\sigma}, \delta)$ -differential privacy for each batch  $s_k$ . Now  
 527 consider the gradient concentration bound achieved by DPG-Lap (Theorem 1) that if  $\frac{2m\sigma^2}{G_1^2} \leq T \leq$   
 528  $\frac{m^2\sigma^4}{169\ln(1/(\sigma\beta))G_1^2}$ , we have

$$\mathbb{P}\left\{\|\tilde{\mathbf{g}}_t - \mathbf{g}_t\| \geq \sqrt{d}\sigma(1 + \mu)\right\} \leq d\beta + d\exp(-\mu), \quad \forall t \in [T].$$

529 Then bring the setting in Theorem 5 that  $\sigma = 1/(nm)^{1/6}$ , let  $\mu = \ln(1/\beta)$  and  $\beta = 1/(dn^{5/3})$ , we  
 530 have

$$\|\tilde{\mathbf{g}}_t - \mathbf{g}_t\|^2 \leq d(1 + \ln d + \frac{5}{3}\ln n)^2/n^{2/3},$$

531 with probability at least  $1 - 1/n^{5/3}$ , when we set  $T = (mn)^{1/3}/(169G_1^2(\ln d + \frac{7}{3}\ln n))$ .

532 Connect this result with Theorem 7, so that we have  $\alpha^2 = d(1 + \ln d + \frac{5}{3}\ln n)^2/(mn)^{1/3}$  and  
 533  $\xi = 1/n^{5/3}$ . Bring the value  $\alpha^2$ ,  $\xi$  and  $T = (mn)^{1/3}/(169G_1^2(\ln d + \frac{7}{3}\ln n))$  into (8), with  
 534  $\rho_{n,d} = \mathcal{O}(\ln n + \ln d)$ , we have

$$\min_{t=1,\dots,T} \|\nabla f(\mathbf{w}_t)\|^2 \leq \mathcal{O}\left(\frac{\rho_{n,d}(f(\mathbf{w}_1) - f^*)}{(mn)^{1/3}}\right) + \mathcal{O}\left(\frac{d\rho_{n,d}^2}{(mn)^{1/3}}\right),$$

535 with probability at least  $1 - \mathcal{O}\left(\frac{1}{\rho_{n,d}n}\right)$ . Here we complete the proof.

536 □

## 537 C Additional Numerical Experiment

538 We present an additional experiment to evaluate our proposed mini-batch SAGD.

539 In this section, we consider a Natural Language Inference task on the Stanford Natural Language  
 540 Inference (SNLI) dataset [3]. The SNLI corpus is a collection of 570 000 human-written English  
 541 sentence pairs manually labeled for balanced classification. The goal is to predict if an hypothesis  
 542 sentence is an *entailment*, *contradiction* or *neutral* with respect to a given text. This task of natural  
 543 language inference (NLI) is also known as recognizing textual entailment.

544 **Dataset and Evaluation Metrics:** For SNLI, all training samples are used to train the model and  
 545 we report the training perplexity and the test perplexity across epochs. Cross-entropy is used as the  
 546 loss function throughout experiments. The mini-batch size is set to 20 for this dataset. We repeat  
 547 each experiment 5 times and report the mean and standard deviation of the results.

548 **Model and Hyperparameters:** We use a bi-directional LSTM architecture, as the concatenation of  
 549 a forward LSTM and a backward LSTM as described in [7]. We use 300 dimensions as fixed word  
 550 embeddings and set the learning rate following the method described in the main paper.

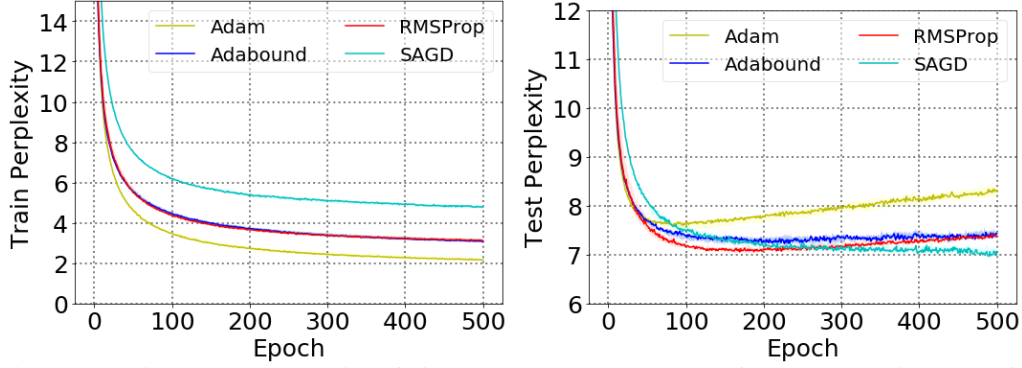


Figure 3: Train and test perplexity of biLSTM on SNLI. SAGD performs the best in terms of the test perplexity among all the methods while showing a worse loss perplexity. SAGD empirically avoids over-fitting.

551 In Figure 3, we compare mini-batch SAGD to the following baselines: Adam [17], RMSprop [33],  
552 and Adabound [22]. As in the NLP task on Penn Treebank, we observe that whilst SAGD displays  
553 a worse loss perplexity than its competition, it succeeds in keeping a low testing perplexity through  
554 the epochs. This phenomena has been observed in all of our experiments (either classification of  
555 images or inference of text) and highlights the advantage of our proposed method to present *reused*  
556 samples to the model as if they were fresh ones. Thus, over-fitting is less likely to happen and testing  
557 loss will remain low. As an example of over-fitting, we observe in In Figure 3 that Adam achieves  
558 the best training perplexity, yet displays an increasing testing perplexity after only a few epochs,  
559 which leads to bad final test accuracy.