UNIVERSITY of HOUSTON
CULLEN COLLEGE of ENGINEERING

ANTS LAB

Towards Better Generalization of Adaptive Gradient Methods

Yingxue Zhou, Belhal Karimi, Jinxing Yu, Zhiqiang Xu and Ping Li

Baidu Research

NeurIPS 2020

First-order gradient optimizers for deep learning

- SGD (Robbins & Monro, 1951)
  - + Momentum (Qian, 1999)
  - + Nesterov (Nesterov, 1983)
- AdaGrad (Duchi et al., 2011)
- RMSprop (Tieleman & Hinton, 2012)
- Adam (Kingma & Lei Ba, 2015)

Name	Update Rule
SGD	$\Delta\theta_t = -\alpha g_t$
Momentum	$m_t = \gamma m_{t-1} + (1-\gamma) g_t,$ $\Delta\theta_t = -\alpha\, m_t$
Adagrad	$G_t = G_{t-1} + g_t^2,$ $\Delta\theta_t = -\alpha\, g_t G_t^{-1/2}$
RMSprop	$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2,$ $\Delta\theta_t = -\alpha\, g_t v_t^{-1/2}$
Adam	$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t,$ $v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2,$ $\hat{m}_t = m_t/(1-\beta_1^t),$ $\hat{v}_t = v_t/(1-\beta_2^t),$ $\Delta\theta_t = -\alpha\, \hat{m}_t \hat{v}_t^{-1/2}$

	Acceler (e.g. SGD, Nester	t Methods n, RMSProp)
Fast convergence		
Good generalization		
Stability for complex settings such as GAN	✗	✓

Stochastic non-convex optimization

★ Minimize the *population loss* $f(\mathbf{w})$ given n i.i.d. samples $\mathbf{z}_1, \ldots, \mathbf{z}_n$ from unknown distribution $\mathcal{P}$:

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

- $\ell : \mathcal{W} \times \mathcal{Z} \mapsto \mathbb{R}$: non-convex loss function
- $\mathbf{z} \in \mathcal{Z}$: data point following unknown distribution $\mathcal{P}$

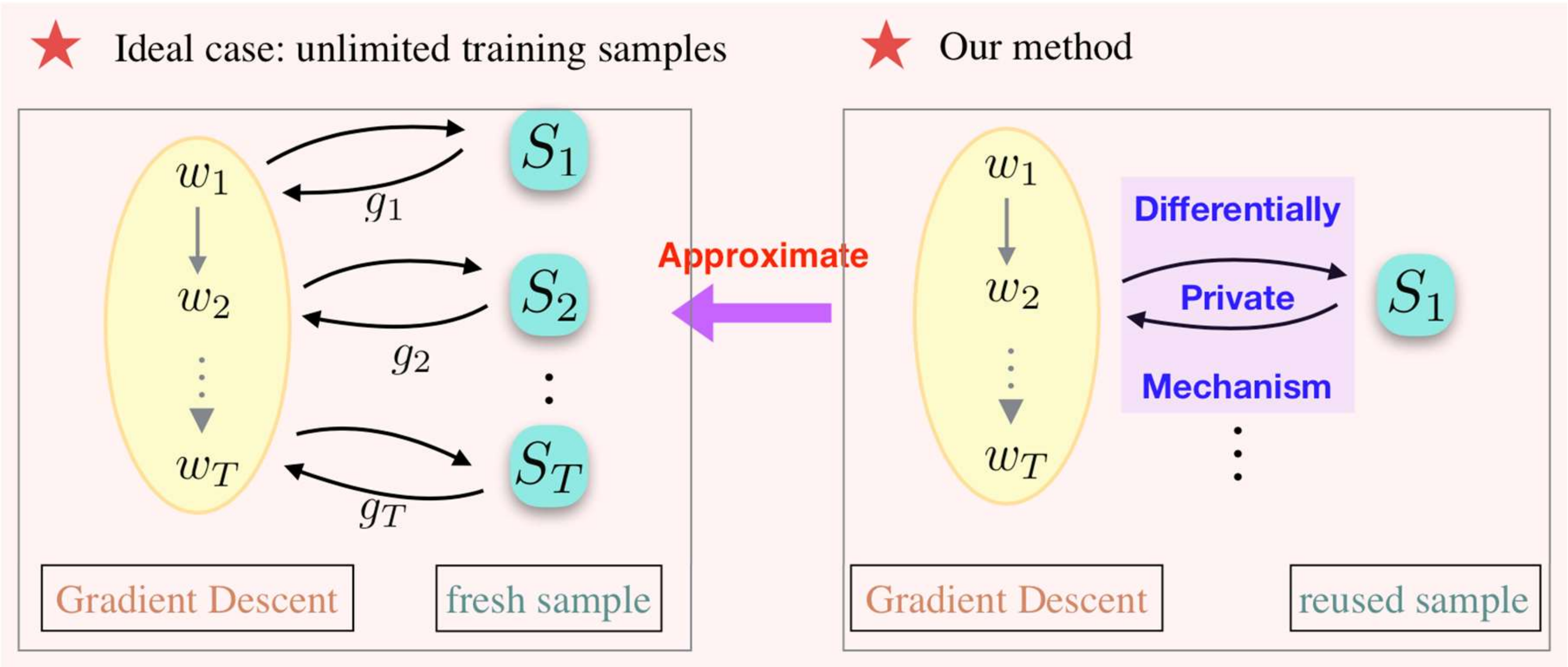
★ Minimize the empirical risk (ERM):

$$\min_{\mathbf{w} \in \mathcal{W}} \hat{f}(\mathbf{w}) \triangleq \frac{1}{n} \sum_{j=1}^{n} \ell\left(\mathbf{w}, \mathbf{z}_j\right)$$

★ Adaptive Gradient Methods: AdaGrad, RMSprop, Adam, AdaBound, etc

- Optimization bounds for the training objective, e.g., norm of the *empirical gradient*.
- Generalization bound, e.g., norm of the *population gradient*.

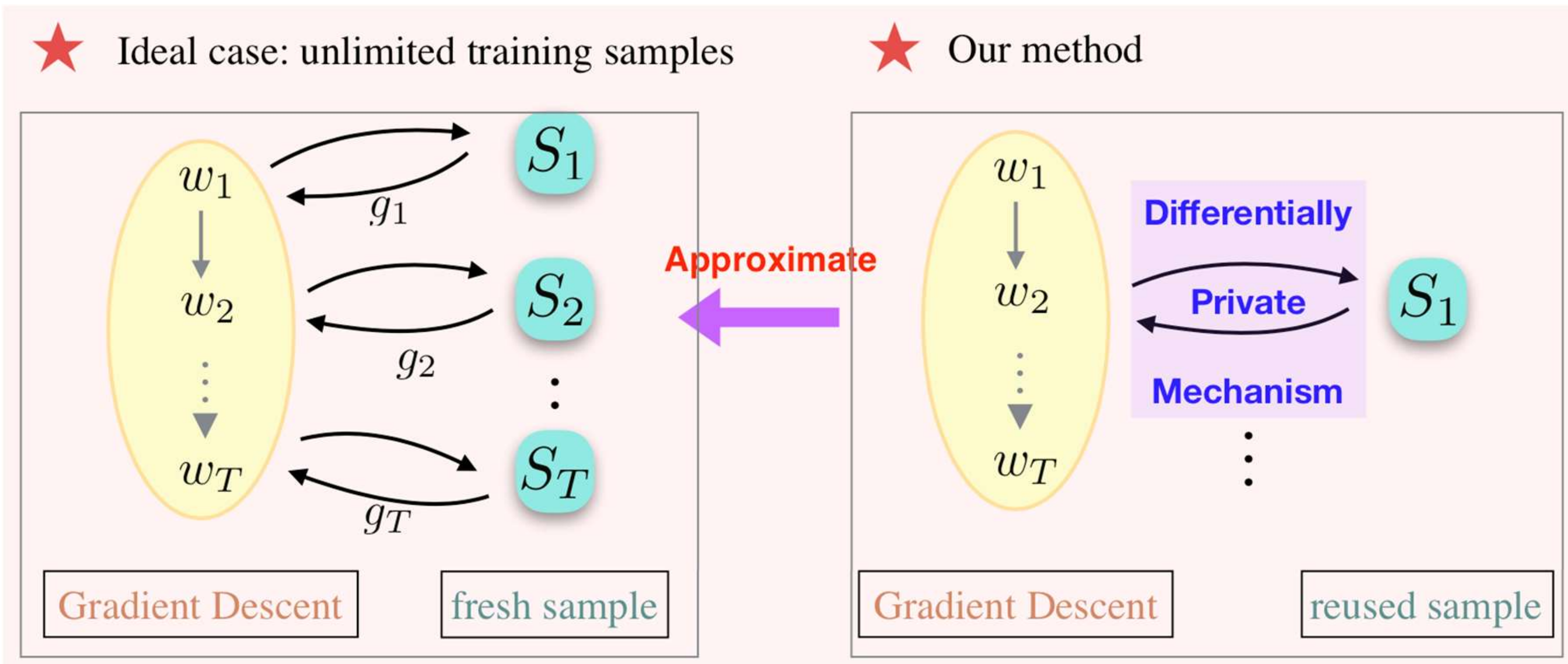
Main Idea



Ideal case: we have access to fresh samples in each iteration

- Sample gradients stay close to the population gradient across all iterations
- Leading to high probability bounds on the population stationary point

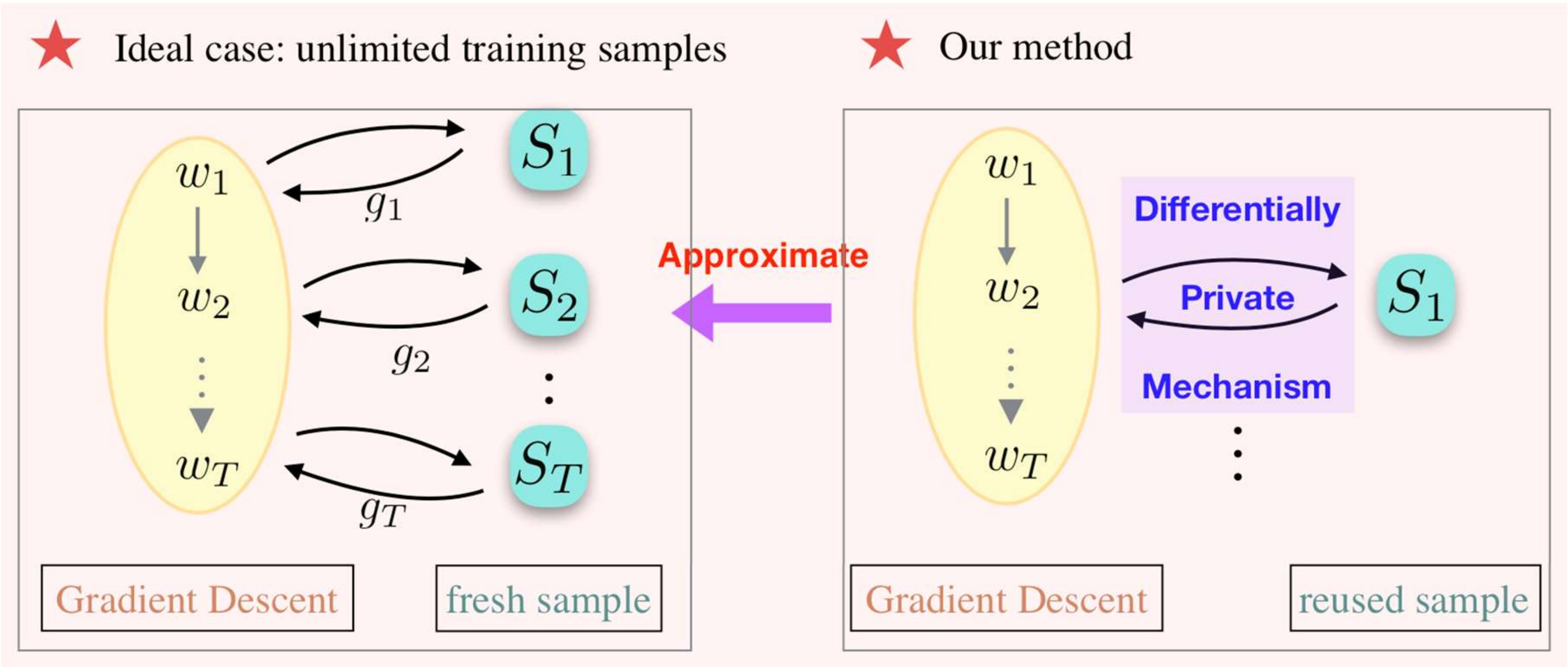
Main Idea



Our method: Stable Adaptive Gradient Descent Algorithm (SAGD)

- Training set S_t maintains the statistical nature of fresh data
- StGD is running multiple passes over the training data, but not doing ERM.

Main Idea



SAGD guarantees:

- ☑ Sample gradients concentrate to population gradients across all iterations
- ☑ Norm of population gradient converges with high probability
- ☑ An upper bound on the number of iterations

Differential Privacy

Main Point: There is no much difference between the output of the algorithm over two datasets that differ in one data element

Definition:

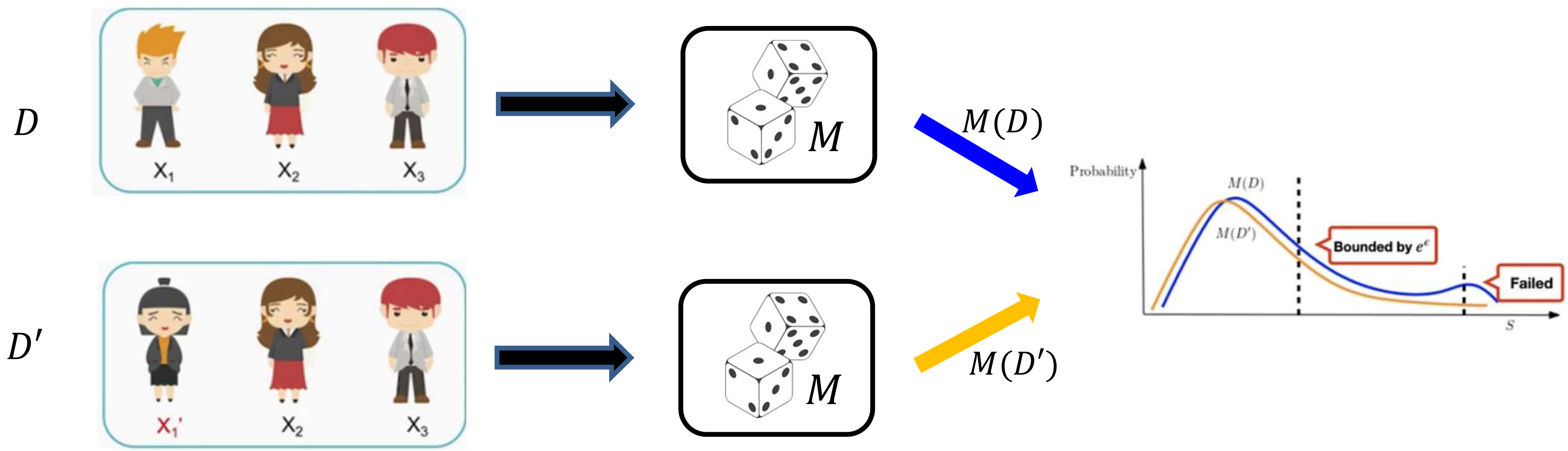
A randomized algorithm M is called (ϵ, δ)-differentially private if for all neighboring datasets D, D' and every outcome $S \subseteq Range(M)$

$$\Pr[\mathcal{M}(D) \in S] \leq e^{\epsilon} \Pr[\mathcal{M}(D') \in S] + \delta.$$

Privacy budget

Probability of failure

Differential Privacy



Basic properties of DP

- For any (ϵ, δ)-DP algorithm $M(\cdot)$, $B(M(\cdot))$ is still (ϵ, δ)-DP for any post-processing $B(\cdot)$

- For any query $q(\cdot)$, adding Gaussian noise $M(\cdot) = q(\cdot) + \mathcal{N}(0,\sigma^2)$ is (ϵ, δ)-DP when $\sigma \propto \frac{\Delta\sqrt{\log 1/\delta}}{\epsilon}$, where $\Delta = \max_{d(D,D')=1} \|q(D) - q(D')\|_2$ is the ℓ_2-norm sensitivity of $q(\cdot)$

- For any query $q(\cdot)$, adding Laplacian noise $M(\cdot) = q(\cdot) + Lap(\sigma)$ is ϵ-DP when $\sigma \propto \frac{\Delta}{\epsilon}$,where $\Delta = \max_{d(D,D')=1} \|q(D) - q(D')\|_1$ is the ℓ_1-norm sensitivity

SAGD with Laplace mechanism

★ SAGD with Laplace mechanism

Algorithm 1 SAGD with DGP-LAP

1: **Input**: Dataset S, certain loss $\ell(\cdot)$, initial point $\mathbf{w}_0$ and noise level σ.
2: Set noise level σ, iteration number T, and stepsize η_t.
3: **for** $t = 0, ..., T-1$ **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 $\mathrm{Lap}(\sigma)$ 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**

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

SAGD with Laplace mechanism

Lemma 1. *Let $\mathcal{A}$ be an (ϵ, δ)-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*

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

- If the privacy cost ϵ is bounded by the estimation error, the differential privacy mechanism enables the reused training sample set to maintain statistical guarantees as if they were fresh samples

- SAGD with DPG-LAP (Alg. 1) is $\left(\frac{\sqrt{T\ln(1/\delta)}G_1}{n\sigma}, \delta\right)$ -differentially private.
- Upper bound on T: $\sqrt{T\ln(1/\delta)}G_1/(n\sigma) \leq \sigma/13$

SAGD with Laplace mechanism

★ High-probability bound: noisy gradient approximates population gradient.

$$\mathbb{P}\left\{\|\tilde{\mathrm{g}}_t - \mathrm{g}_t\| \geq \sqrt{d}\sigma(G+\mu)\right\} \leq d\beta + d\exp(-\mu), \ \forall t \in [T], \ \beta > 0 \ \text{ and } \mu > 0.$$

★ Non-asymptotic convergence rate (population gradient):

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

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

- Given n samples, previous approaches can achieve $\mathcal{O}(1/\sqrt{n})$
- This paper can achieve $\mathcal{O}(1/n^{2/3})$

SAGD with Laplace mechanism

★ High-probability bound: noisy gradient approximates population gradient.

$$\mathbb{P}\left\{\|\tilde{\mathrm{g}}_t - \mathrm{g}_t\| \geq \sqrt{d}\sigma(G+\mu)\right\} \leq d\beta + d\exp(-\mu),\ \forall t \in [T],\ \beta > 0 \text{ and } \mu > 0.$$

★ Non-asymptotic convergence rate (population gradient):

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

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

- SAGD with DPG-LAP (Alg. 1) is $\left(\frac{\sqrt{T\ln(1/\delta)}G_1}{n\sigma}, \delta\right)$ -differentially private.
- Upper bound on T: $\sqrt{T\ln(1/\delta)}G_1/(n\sigma) \leq \sigma/13$

SAGD with Sparse vector technique

★ SAGD with Sparse vector technique

Algorithm 2 SAGD with DPG-SPARSE

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

SAGD with Sparse vector technique

- SAGD with DPG-SPARSE is $\left(\frac{\sqrt{C_s \ln(2/\delta)}2G_1}{n\sigma}, \delta\right)$ -differentially private.
- C_s - the number of times the validation fails, i.e., $\|\hat{\mathrm{g}}_{S_1,t} - \hat{\mathrm{g}}_{S_2,t}\| + \gamma > \tau$ is true, over T iterations in SAGD.
- Imply an improved upper bound on T: $\sqrt{C_s \ln(1/\delta)}G_1/(n\sigma) \leq \sigma/13$

if $C_s = \mathcal{O}(\sqrt{T})$, the upper bound of T can be improved from $T \leq \mathcal{O}(n^2)$ to $T \leq \mathcal{O}(n^4)$

Mini-batch SAGD algorithm

Algorithm 3 Mini-Batch SAGD

1: **Input**: Dataset S, certain loss $\ell(\cdot)$, initial point $\mathbf{w}_0$.
2: Set noise level σ, epoch number T, batch size m, and stepsize η_t.
3: Split S into $B = n/m$ batches: $\{s_1, ..., s_B\}$.
4: **for** $epoch = 1, ..., T$ **do**
5: **for** $k = 1, ..., B$ **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**

★Non-asymptotic convergence rate (population gradient):

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

- When $m = \sqrt{n}$, mini-batch SAGD achieves $\mathcal{O}(1/\sqrt{n})$ convergence rate
- When $m = n$, recover the full batch convergence rate

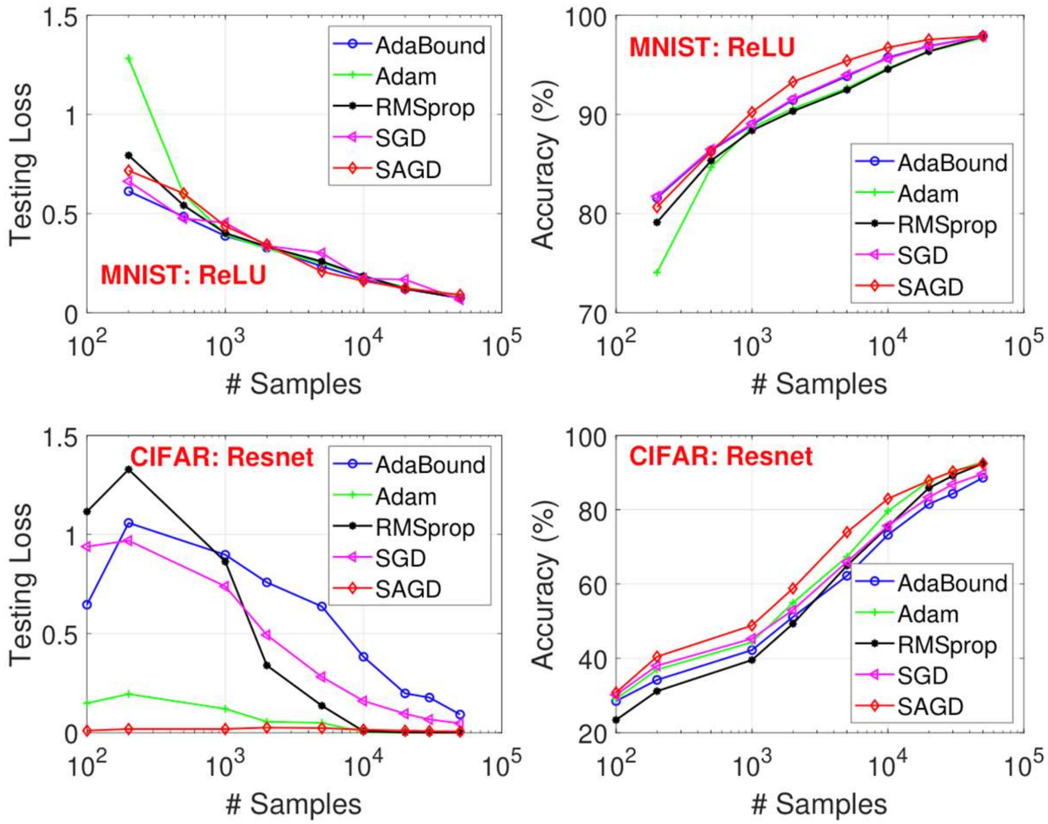
Experiments Settings

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

For each task, 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, …} are constructed.
- For CIFAR10, training sets of size n $\in$ {200, 500, 1000, …} are constructed.

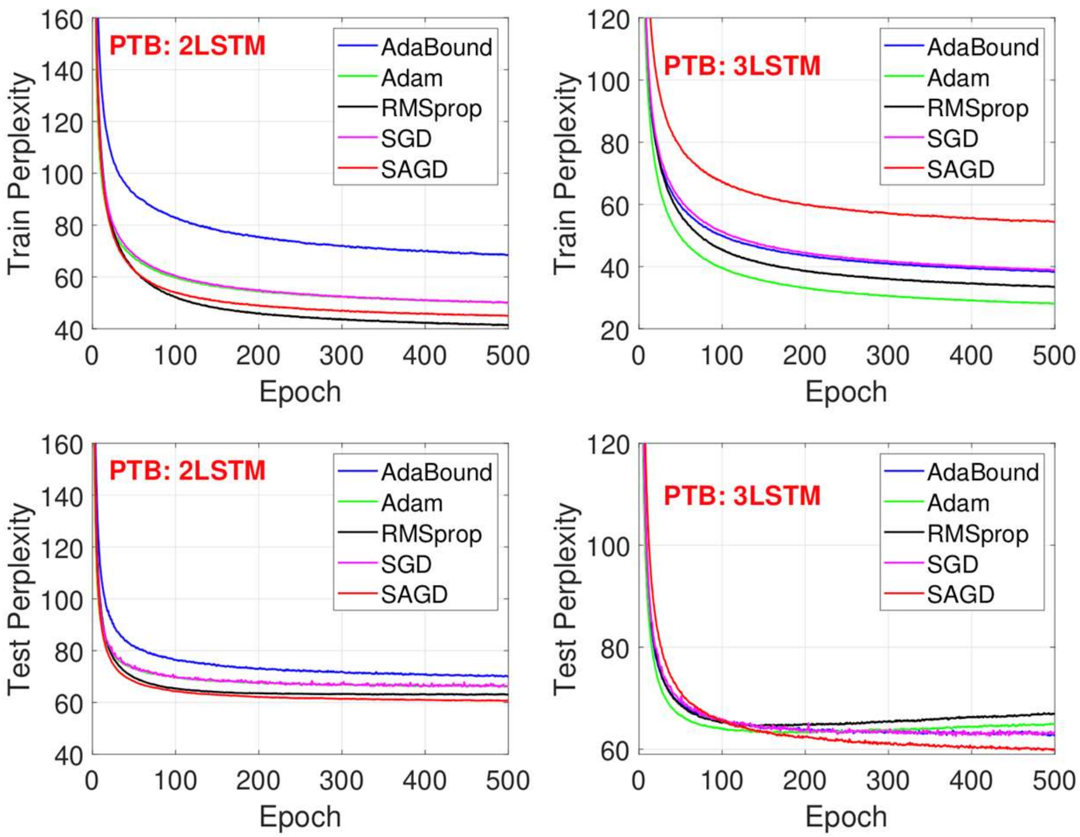
Experiments Results



- Test loss and test accuracy of ReLU neural network on MNIST
- Test loss and accuracy of ResNet-18 on CIFAR10

- SAGD obtains the best test accuracy among all the methods

Experiments Results



- Train and test perplexity of 2-layer LSTM (2LSTM), 3- layer LSTM (3LSTM)
- Even though some baseline optimizers achieve better training performance than SAGD, the latter performs the best in terms of test perplexity among all methods.

Conclusions

- Focus on the generalization ability of adaptive gradient methods
- Propose SAGD algorithms, which boost the generalization performance in both theory and practice through a novel use of differential privacy
- Experimental studies highlight that the proposed algorithms are competitive and often better than baseline algorithms for training deep neural networks