Towards Better Generalization of Adaptive Gradient Methods

Yingxue Zhou[1], Belhal Karimi[2], Jinxing Yu[2], Zhiqiang Xu[2], Ping Li[2]

[1]University of Minnesota - Twin Cities, [2]Cognitive Computing Lab, Baidu Research

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):

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

★ 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*.

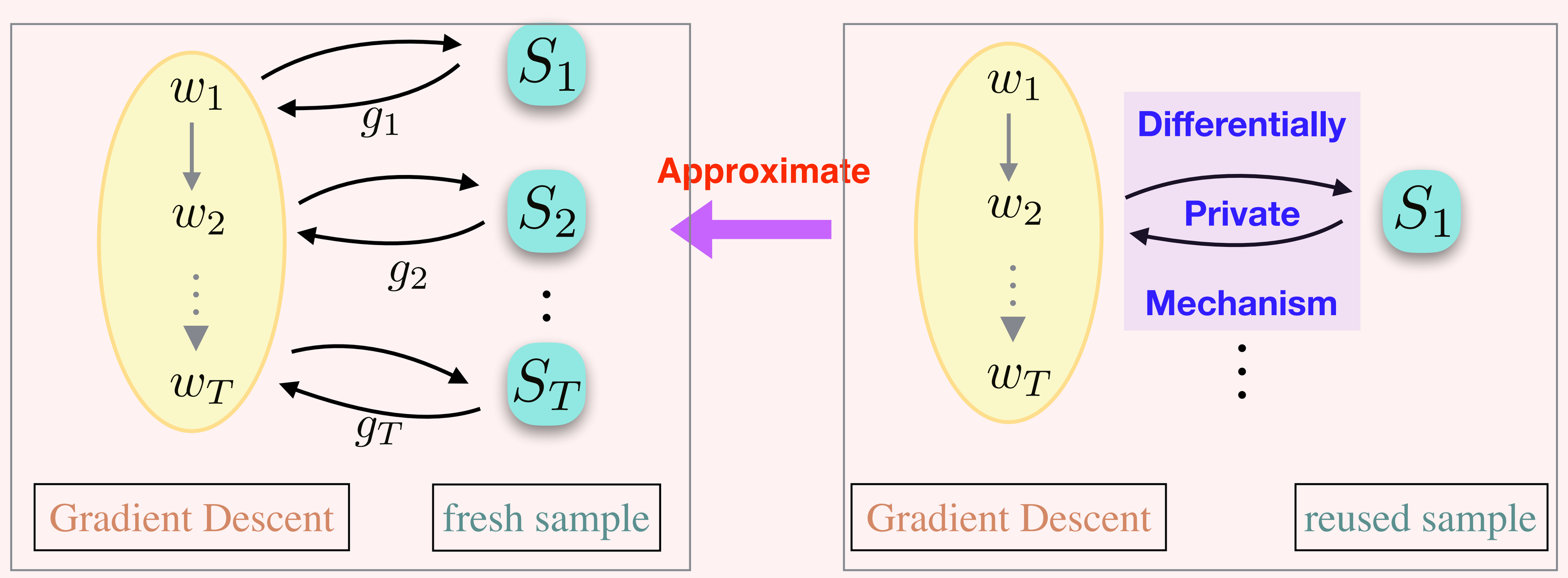
Our Idea

★ **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

★ Ideal case: unlimited training samples

★ Our method



★ **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.

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:
and for all pairs of adjacent datasets x, y that differ on a single data point:

$$\ln\left(\frac{P\{\mathcal{M}(x)\in\mathcal{S}\}}{P\{\mathcal{M}(y)\in\mathcal{S}\}}\right) \le \epsilon.$$

SAGD algorithm

★ 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** $t = 0, \ldots, T-1$ **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 Lap(σ) 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.
- Upper bound on T: $\sqrt{T\ln(1/\delta)}G_1/(n\sigma) \le \sigma/13$

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

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

★ Non-asymptotic convergence rate (population gradient):

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

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

★ SAGD with Sparse vector technique

Algorithm 2 SAGD with DPG-SPARSE

1: **Input**: Dataset S, certain loss $\ell(\cdot)$, initial point $\mathbf{w}_0$.
2: Set noise level σ, iteration number T, and stepsize η_t.
3: Split S randomly into S_1 and S_2.
4: **for** $t = 0, \ldots, T-1$ **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 $\gamma\sim$ Lap(2σ), $\tau\sim$ Lap(4σ).
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 Lap(σ), 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**: $\tilde{\mathbf{g}}_t$.

- 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., $\|\hat{\mathbf{g}}_{S_1,t} - \hat{\mathbf{g}}_{S_2,t}\| + \gamma > \tau$ is true, over T iterations in SAGD.
- Imply an improved upper bound on T: $\sqrt{C_s\ln(1/\delta)}G_1/(n\sigma) \le \sigma/13$

Experimental Results

Model: 2-layer LSTM and 3-layer LSTM
Datasets: Penn Treebank

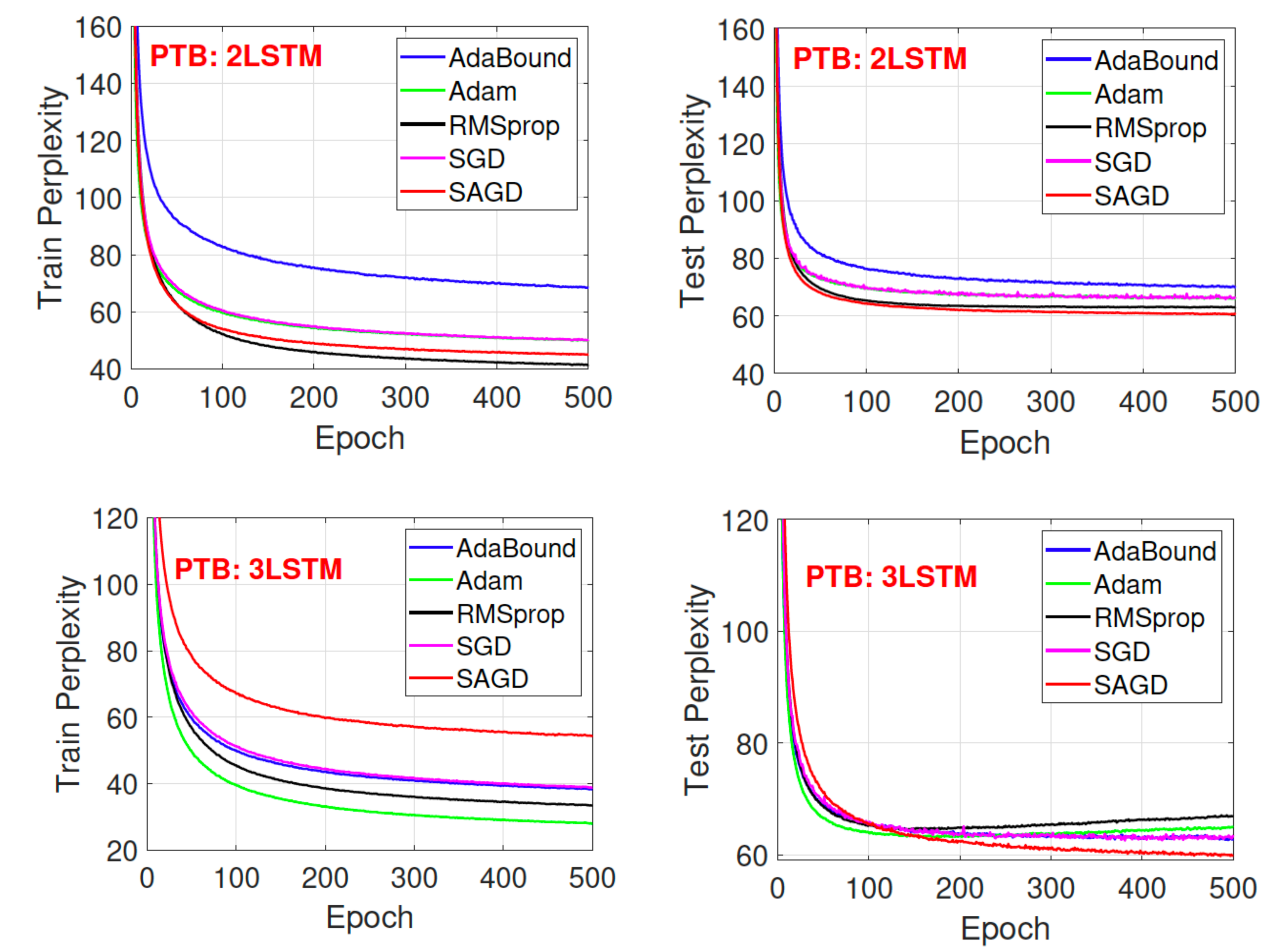


Figure 1: Train (upper panels) and test (bottom panels) perplexity of 2-layer LSTM (2LSTM) and 3-layer LSTM (3LSTM). Even though some baseline optimizers achieve better training performance than SAGD, the latter performs the best in terms of test perplexity among all methods.

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

	RMSprop	Adam	AdaBound	SGD	SAGD
2-layer LSTM	62.87 ± 0.05	66.02 ± 0.05	65.82 ± 0.08	65.96 ± 0.23	**60.66 ± 0.05**
3-layer LSTM	63.97 ± 018	63.23 ± 004	62.33 ± 0.07	62.51 ± 0.11	**59.43 ± 0.24**

References

Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Roth. Generalization in adaptive data analysis and holdout reuse. In Advances in Neural Information Processing Systems (NIPS) , pages 2350–2358, Montreal, Quebec, Canada, 2015.

Moritz Hardt, Ben Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. In Proceedings of the 33nd International Conference on MachineLearning (ICML), pages 1225–1234, New York City, NY, 2016.

Saeed Ghadimi and Guanghui Lan. Stochastic first- and zeroth-order methods for nonconvex stochastic programming. SIAM J. Optim. , 23(4):2341–2368, 2013.Ashia C. Wilson, Rebecca Roelofs, Mitchell Stern, Nati Srebro, and Benjamin Recht.

Ashia C. Wilson, Rebecca Roelofs, Mitchell Stern, Nati Srebro, and Benjamin Recht. The marginal value of adaptive gradient methods in machine learning. In Advances in Neural Information Processing Systems (NIPS), pages 4148–4158, Long Beach, CA, 2017.

Acknowledgements

We thank the anonymous Referees and Area Chair for their constructive comments. The work is supported by Baidu Research.