Gossip Gradient Descent

Extended Abstract

Yang Liu
Harvard University
yangl@seas.harvard.edu

Ji Liu
Stony Brook University
ji.liu@stonybrook.edu

Tamer Başar
University of Illinois at
Urbana-Champaign
basar1@illinois.edu

ABSTRACT

We consider a problem of learning a linear regression model distributively with a network of N interconnected agents which receive private streaming data. Each agent can deploy an online learning algorithm, e.g. stochastic gradient descent, to learn adaptively the regression model using its receiving private data. The goal is to devise an algorithm for each agent, under the constraint that each of them can communicate only with its neighboring agents based on a communication graph, to enable each agent converge to the true model with a performance comparable to that of the traditional centralized solution. We propose an algorithm called *gossip gradient descent*, and establish $O\left(\sqrt{\frac{\log t}{(1-\lambda_2)Nt}}\right)$ convergence in expectation and mean square, where λ_2 is the second largest eigenvalue of the expected gossip matrix corresponding to the underlying communication graph. For the case when agents are privacy sensitive, we propose a differentially private variant of the algorithm, which achieves ϵ-differential privacy and $O\left(\sqrt{\frac{\log^2 t}{\epsilon(1-\lambda_2)Nt}}\right)$ convergence.

ACM Reference Format:
Yang Liu, Ji Liu, and Tamer Başar. 2018. Gossip Gradient Descent. In *Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), Stockholm, Sweden, July 10–15, 2018,* IFAAMAS, 3 pages.

1 INTRODUCTION

Distributed learning [1–3, 5, 10, 11, 13, 17] has become very popular for learning with large-scale training data, via leveraging distributed computing resources. Most existing distributed learning methods focus on improving computational efficiency without considering data collection and dissemination overhead. Nonetheless, coordinating the distributed learning sources requires substantial communications. This poses significant challenges when data is collected in a highly distributed manner and when the learned model needs to be re-distributed to the decentralized agents, as it necessitates a large amount of communication and effort in synchronization.

Within a learning context, the idea of distributed learning via cheap and asynchronous communication among computing resources has recently been implemented for learning different statistics [6, 9, 14]. Empirical evidence has also demonstrated the potential of asynchronous distributed learning [16]. In particular, gossip-type algorithms [4, 12] have been studied extensively owing to their simple, elegant and powerful solution structure. Existing results have established convergence of gossip algorithms devised for problems in which each agent acquires a static set of data and exchanges its local information with one of its neighbors at each time in an asynchronous manner. With above being said, most of the existing gossiping solutions focus on the case when local statistics/data of each agent remain unchanged during the message/information passing process. In contrast, it is often more likely that each agent observes a sequence of arriving data to update its local model or belief. For instance, for machine learning tasks that involve large-scale data collection, training and re-distribution (of the trained model), it is very likely that different agents or sensors collect their own data gradually over time. We aim to address the question of incorporating agents' local learning information into the distributed updating procedure in a communication-efficient way by integrating new measurements/samples into a single gossip process as they arrive.

2 PROBLEM FORMULATION

Consider a network of N agents. They communicate with each other over a communication graph $G = (V, E)$, where $V = \{1, 2, \ldots, N\}$, and E denotes the set of edges characterizing neighbor relationships in that agents i and j are called neighbors of each other whenever (i, j) is an edge in G. Each agent can communicate only with its neighbors. The communication graph G is a simple, undirected, connected graph. Suppose that there is a common discrete-time clock $t \in \{1, 2, \ldots\}$. Each agent observes a sequence of training data $\{x_i(t), y_i(t)\}_t$, where $x_i(t) \in \mathbb{R}^d$, $y_i(t) \in \mathbb{R}$. Each pair of the observations comes from a linear regression model $y_i(t) = (\theta^*)^\top x_i(t) + z_i(t)$ for some $\theta^* \in \mathbb{R}^d$, where $z_i(t)$ is a zero mean noise with bounded support, that is, $|z_i(t)| \le Z$ for some $Z > 0$. For analytical tractability, we assume that both sequences $\{x_i(t)\}_{i,t}$ and $\{z_i(t)\}_{i,t}$ are i.i.d., and that $\{x_i(t)\}_{i,t}$ are sampled uniformly from the unit ball (with dimension d), that is, $||x_i(t)||_2 \le 1$. The goal of the learning algorithm is to learn a consistent model $\theta' \to \theta^*$ in that both $||\mathbb{E}[\theta'] - \theta^*||_2 \to 0$ and $\mathbb{E}||\theta' - \theta^*||_2 \to 0$. We are interested in the setting where agents do not have the capability to broadcast their information to all other agents, and there does not exist a central authority to collect and aggregate all local information. Instead, agents can only communicate their local model information with their neighbors. The problem here is to design a communication-efficient algorithm to achieve comparable convergence results for this decentralized setting.

When training data arrives in an online fashion, a common solution is to do online gradient descent as given in Algorithm 1 (see details in [15], with some prior $\tilde{\theta}_i^{\text{online}}(0)$), where $\nabla\tilde{\theta}_i(t)$ is the gradient update at time t, and η_t is the step size at time t. This

Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), M. Dastani, G. Sukthankar, E. André, S. Koenig (eds.), July 10–15, 2018, Stockholm, Sweden.

online updating procedure is often referred to as online *Stochastic Gradient Descent* (SGD).

Algorithm 1 (Online Stochastic Gradient Descent for Regularized Linear Regression)

$$\tilde{\theta}_i^{\text{online}}(t+1) = \tilde{\theta}_i^{\text{online}}(t) + \nabla\tilde{\theta}_i(t), \text{ where}$$

$$\nabla\tilde{\theta}_i(t) = -\eta_t \nabla_{\tilde{\theta}_i^{\text{online}}(t)} \left[(\theta^\top x_i(t) - y_i(t))^2 + \gamma ||\theta||_2^2 \right],$$

3 GOSSIP GRADIENT DESCENT

We extend the classical gradient descent algorithm to a gossiping setting, which we name as *gossip gradient descent* (GGD). Each agent i maintains $\vartheta_i(t)$, $\tilde{\theta}_i^{\text{online}}(t)$, and $\nabla\tilde{\theta}_i(t)$ where (1) $\vartheta_i(t)$ is agent i's estimation of θ^* using both its locally observed information and information received from its neighbors at time t, (2) $\tilde{\theta}_i^{\text{online}}(t)$ is agent i's local model (trained only using its local information), and (3) $\nabla\tilde{\theta}_i(t)$ is agent i's local gradient as defined in the online gradient descent algorithm. Following the basic idea of gossiping, at every step t, we activate exactly two agents (i, j) on one edge from E on G uniformly and randomly, and they will exchange and update information. Define $W(t) = I_N - \frac{1}{2}(e_i - e_j)(e_i - e_j)^\top$, where e_i is the unit vector with the ith entry equal to 1. Note that

$$W := \mathbb{E}[W(t)] = \frac{1}{|E|} \sum_{(i,j)\in E} \left(I_N - \frac{1}{2}(e_i - e_j)(e_i - e_j)^\top \right) , \quad (1)$$

which is a symmetric matrix. At initial time $t = 1$, each agent i sets its local estimate as $\vartheta_i(1) = \tilde{\theta}_i^{\text{online}}(1)$, using its local observed sample $(x_i(1), y_i(1))$. Starting from $t = 2$, exactly one edge is activated, and only the two agents connected by this edge exchange information. When agents i and j are selected to exchange a message, they update their local beliefs as follows:

$$\vartheta_i(t) = \frac{\vartheta_i(t-1) + \vartheta_j(t-1)}{2} + \nabla\tilde{\theta}_i(t) , \quad (2)$$

$$\vartheta_j(t) = \frac{\vartheta_i(t-1) + \vartheta_j(t-1)}{2} + \nabla\tilde{\theta}_i(t) . \quad (3)$$

For every other agent $k \notin \{i, j\}$,

$$\vartheta_k(t) = \vartheta_k(t-1) + \nabla\tilde{\theta}_k(t) . \quad (4)$$

This algorithm is summarized below as Algorithm 2. Let λ_2 be the second largest eigenvalue of W defined in (1). We have the following convergence results for GGD.

Algorithm 2 (Gossip Gradient Descent)

```
Initialization: $t = 1$, $\vartheta_i(1) = \tilde{\theta}_i^{\text{online}}(1), \forall i$.
while No termination do
    1. At time $t$, select one edge from $E$ uniformly; denote the edge as $e(t) = (i, j)$.
    2. Agents compute local model and gradient $(\tilde{\theta}_i^{\text{online}}(t), \nabla\tilde{\theta}_i(t)), \forall i$ via standard online SGD.
    3. For $i, j$, update $\vartheta_i(t), \vartheta_j(t)$ via Eqn. (2) and (3).
    4. For $k \notin \{i, j\}$, update $\vartheta_k(t)$ via Eqn. (4).
    5. $t := t + 1$.
end while
```

Theorem 3.1. *With GGD, at any time t, $\forall i$,*

$$\left\| \mathbb{E}[\vartheta_i(t)] - \theta^* \right\|_2 \le O\left(\sqrt{\frac{\log t}{(1-\lambda_2)Nt}} \right) ,$$

$$\mathbb{E}\left\| \vartheta_i(t) - \theta^* \right\|_2 \le O\left(\sqrt{\frac{\log t}{(1-\lambda_2)Nt}} \right) .$$

4 PRIVATE GOSSIP GRADIENT DESCENT

We extend the algorithm proposed earlier to a privacy preserving one. Suppose that all $x_i(t)$'s (the feature variables) are publicly observable, but the local responses, $y_i(t)$'s, are private. We follow the basic idea of adding Laplacian noise to the information that contains each agent's private data before exchanging them [8]. This is summarized in Algorithm 3.

Algorithm 3 (Private Gossip Gradient Descent)

```
Add individual Laplacian noise $v$ to each $y_i(n)$:
    $\tilde{y}_i(n) = y_i(n) + v, \ \Pr(v) \propto \exp(-\epsilon|v|)$.
Train $(\tilde{\theta}_i^{\text{online}}(t), \nabla\tilde{\theta}_i(t))$ using $\{\tilde{y}_i(n)\}_n$ instead of $\{y_i(n)\}_n$.
Everything else follows (Gossip Gradient Descent).
```

To quantify privacy preserving level, we will adopt differential privacy as our solution concept [7]. Denote by $\mathcal{D}$ the set of data $\{(x_i(t), y_i(t))\}_{i,t}$ when the algorithm terminates. We have mechanism (algorithm) $\mathcal{M} : (\mathbb{R}^d \times \mathbb{R})^{|\mathcal{D}|} \to \mathbb{R}$ is ϵ-differentially private if for any $i \in d(t)$, t, any two distinct $y_i(t), y_i'(t)$, and for every subset of possible outputs $S \subseteq \mathcal{O}$ ($\mathcal{O}$ denotes the outcome space),

$$\Pr[\mathcal{M}(\mathcal{D}) \in S] \le \exp(\epsilon) \Pr\left[\mathcal{M}\left(\{\mathcal{D} \backslash \{y_i(t)\}, y_i'(t)\} \right) \in S \right] .$$

Note that the noise can be added in different ways. Adding noise directly to each $\nabla\tilde{\theta}_i(t)$ will not preserve enough privacy: each data point $y_i(n)$ appears in infinitely many $\nabla\tilde{\theta}_i(t)$, $t \ge n$. If the noise level in $\nabla\tilde{\theta}_i(t)$ is able to preserve ϵ-DP for each $y_i(n)$, and we can prove that the sensitivity of $\nabla\tilde{\theta}_i(t)$ of $y_i(n)$ is $O(\frac{1}{t})$ w.h.p. (since $\eta_t = \Theta(1/t)$). Then, summing over all t (by composition theory of differential privacy [8]), we have the preserved differential privacy level at step t at best as $\sum_{\tau=1}^{t} \frac{1}{t} \cdot \epsilon \propto \epsilon \log t$, which will diverge for any positive constant ϵ. The above observation motivates us to add noise to raw training samples directly.

Following classical differential privacy results [8], PGGD achieves ϵ-DP for each agent's data point. Since we have added noise to each individual data point, it is to be expected that this additional noise will slow down the convergence of our algorithm.

Theorem 4.1. *With PGGD, at any time t, $\forall i$,*

$$\left\| \mathbb{E}[\vartheta_i(t)] - \theta^* \right\|_2 \le O\left(\sqrt{\frac{\log^2 t}{\epsilon(1-\lambda_2)Nt}} \right) ,$$

$$\mathbb{E}\left\| \vartheta_i(t) - \theta^* \right\|_2 \le O\left(\sqrt{\frac{\log^2 t}{\epsilon(1-\lambda_2)Nt}} \right) .$$

REFERENCES

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, and M. Isard. 2016. TensorFlow: A system for Large-scale Machine Learning. In *12th OSDI*.

[2] R. Bekkerman, M. Bilenko, and J. Langford. 2011. *Scaling Up Machine Learning: Parallel and Distributed Approaches*. Cambridge University Press.

[3] L. Bottou. 2010. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*. 177–186.

[4] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. 2006. Randomized Gossip Algorithms. *IEEE Transactions on Information Theory* 52, 6 (2006), 2508–2530.

[5] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. 2011. Distributed optimization and statistical learning via the Alternating Direction Method of Multipliers. *Foundations and Trends® in Machine Learning* 3, 1 (2011), 1–122.

[6] I. Colin, A. Bellet, J. Salmon, and S. Clémençon. 2015. Extending Gossip Algorithms to Distributed Estimation of U-Statistics. In *Advances in Neural Information Processing Systems*. 271–279.

[7] C. Dwork. 2006. Differential privacy. In *Proceedings of the 33rd International Conference on Automata, Languages and Programming*. 1–12.

[8] C. Dwork and A. Roth. 2014. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science* 9, 3-4 (2014), 211–407.

[9] S. B. Korada, A. Montanari, and S. Oh. 2011. Gossip PCA. In *Proceedings of the ACM SIGMETRICS*. ACM, 209–220.

[10] M. Li, D. G. Andersen, J. W. Park, A. J. Smola, A. Ahmed, V. Josifovski, J. Long, E. J. Shekita, and B.-Y. Su. 2014. Scaling distributed machine learning with the parameter server. In *11th OSDI*.

[11] Roger Mailler and Victor Lesser. 2004. Solving distributed constraint optimization problems using cooperative mediation. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 1*. IEEE Computer Society, 438–445.

[12] Elth Ogston and Frances Brazier. 2011. Agentscope: Multi-agent systems development in focus. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems, 389–396.

[13] Liviu Panait and Sean Luke. 2005. Cooperative multi-agent learning: The state of the art. *Autonomous agents and multi-agent systems* 11, 3 (2005), 387–434.

[14] K. Pelckmans and J. A. K. Suykens. 2009. Gossip algorithms for computing U-statistics. In *Proceedings of the First IFAC Workshop on Estimation and Control of Networked System*. 48–53.

[15] A. Rakhlin, O. Shamir, and K. Sridharan. 2012. Making gradient descent optimal for strongly convex stochastic optimization. In *Proceedings of the 29th International Conference on Machine Learning*. 449–456.

[16] P. Smyth, M. Welling, and A. U. Asuncion. 2009. Asynchronous distributed learning of topic models. In *Advances in Neural Information Processing Systems*. 81–88.

[17] L. Xiao, S. Boyd, and S.-J. Kim. 2007. Distributed average consensus with least-mean-square deviation. *J. Parallel and Distrib. Comput.* 67, 1 (2007), 33–46.