2019 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)

Random Walk Gradient Descent for Decentralized Learning on Graphs

Ghadir Ayache
ECE Department, Rutgers University
New Brunswick, NJ
ghadir.ayache@rutgers.edu

Salim El Rouayheb
ECE Department, Rutgers University
New Brunswick, NJ
salim.elrouayheb@rutgers.edu

***Abstract*—We design a new variant of the stochastic gradient descent algorithm applied for learning a global model based on the data distributed over the nodes of a network. Motivated by settings such as in decentralized learning, we suppose that one special node in the network, which we call node 1, is interested in learning the global model. We seek a decentralized and distributed algorithm for many reasons including privacy and fault-tolerance. A natural candidate here is Gossip-style SGD. However, it suffers from slow convergence and high communication cost mainly because at the end all nodes, and not only the special node, will learn the model.**

We propose a distributed SGD algorithm using a weighted random walk to sample the nodes. The Markov chain is designed to have stationary probability distribution that is proportional to the smoothness bound L_i of the local loss function at node i. We study the convergence rate of this algorithm and prove that it depends on the smoothness average $\overline{L}$. This outperforms the case of uniform sampling algorithm obtained by a Metropolis-Hasting random walk (MHRW) which depends on the supremum of all L_is noted $\sup L$. We present numerical simulations that substantiate our theoretical findings and show that our algorithm outperforms random walk and gossip-style algorithms.

***Index Terms*—MCMC, SGD, decentralized learning.**

I. Introduction

We consider the setting in which data is distributed among nodes in a graph. One special node in the graph wants to learn a model by minimizing an averaged loss function on all the distributed data using an iterative Gradient Descent-based algorithm. Our main motivation is applications such as federated learning [1] where the nodes are users having their personal data on their phones and a model is to be learned on their collective data. However, for privacy concerns, we seek a decentralized solution in which nodes communicate with their neighbours without involving a centralized server. Our theoretical results can also be applied to any decentralized learning problem in networks such as social networks or Internet-Of-Things (IoT) applications.

Consensus or Gossip-like algorithms allow decentralized learning problem by making every node learn the model locally by only exchanging updates with neighbors [2, 3]. However, in our setting, not all the nodes are interested in learning the model and thus a gossip algorithm can suffer from slow convergence rate and incur high communication cost. Instead, we propose a decentralized algorithm based on a random walk on the graph. At each iteration, a node in the graph gets a token holding the last updated model and then updates it based on its data using the gradient descent update rule. In the next iteration, one of its neighbouring (adjacent) nodes is sampled and the token is passed to him. The main question we want to address is how to sample the graph nodes to ensure fast convergence of the algorithm.

We assume the local loss function at each node is convex and gradient Lipschitz continuous. We propose two algorithms for decentralized gradient descent based on Metropolis Hasting Random Walk (MHRW). The first based on uniform sampling and the second on weighted sampling. Our main contribution is proving refined convergence analysis of the decentralized algorithms using recent results on the convergence analysis of SGD [4, 5]. In particular we make the following contributions:

1) We propose a Uniform Random Walk Gradient Descent (RWGD) algorithm in which the nodes in the graph are sampled uniformly (as the number of iteration goes to infinity) based only on the nodes degrees. Our algorithm is similar to the one proposed in [5]. However, the novelty here is implementing the assumption above on the gradient Lipschitz loss function on the theoretical guarantees.
2) We propose a second algorithm which we call Weighted RWGD algorithm. The algorithm used MHRW to sample the nodes based on the distribution in [4] which takes into account the data at each node (Lipschitz constant). We also derive a theoretical analysis of the convergence rate of this algorithm and show that it outperforms uniform sampling.
3) We present numerical simulations for the Uniform RWGD, Weighted RWGD and the asynchronous Gossip GD that has equivalent number of communication rounds. First, we consider Erdos-Renyi graphs. Our simulations confirm our theoretical finding in showing that the Weighted RWGD outperforms the Uniform RWGD. To understand the effect of the graph structure on the convergence rate, we also compare our algorithms on two other types of graphs: Expander and Stochastic Block Model.

978-1-5386-5555-9/19/$31.00 ©2019 IEEE
DOI 10.1109/IPDPSW.2019.00157

II. Model

A. Network model and problem formulation

We consider a network of N interacting nodes represented by an undirected connected graph $G(V, E)$. $V = \{1, 2.., N\}$ is the set of all nodes and $E \subseteq V \times V$ is the set of undirected edges. We say that two nodes are adjacent or neighbors if they are connected by an edge. Let $\mathcal{N}(i)$ be the set of nodes that are neighbors to node i. We suppose that each node can only exchange information with its neighbors.

Each node i holds local private data represented by (x_i, y_i), where $x_i \in \mathbb{R}^d$ represents a d-dimensional data point. $y_i \in \mathbb{R}$ is the label assigned to that data point. We collect the data at all the nodes in one matrix $X \in \mathbb{R}^{N \times d}$, where x_i is the ith row of X. Similarly $y \in \mathbb{R}^N$ is the label vector and has y_i as its ith coordinate. We suppose that one special node of the network, called node 1, is interested in learning a global model $w^* \in \mathcal{W} \subset \mathbb{R}^d$ that minimizes an average loss function

$$f(w) = \frac{1}{N} \sum_{i=1}^{N} f_i(w), \tag{1}$$

subject to

$$w \in \mathcal{W}, \tag{2}$$

where $f_i(.)$ is the local loss function at node i and $\mathcal{W}$ is a convex compact set. So the goal is to find $w^* \in \mathcal{W}$ satisfying

$$w^* = \arg\min_{w \in \mathcal{W}} \frac{1}{N} \sum_{i=1}^{N} f_i(w). \tag{3}$$

B. Existing algorithms

The literature focuses mostly on two approaches to solve this problem: centralized and Gossip-like algorithms.

Centralized SGD: In the centralized approach, at each iteration, we sample a subset of nodes, then the learning goal could be achieved in two ways: either the sampled node sends its data to node 1 to perform the stochastic gradient descent update, or, in two rounds of communication, node 1 sends the global model to the sampled node, the last does the update on its local data and sends back the new global model to the special node [1, 6].

Gossip SGD: Another approach suggests to decentralize the work by running a Gossip-like algorithm where every node holds a local model and participate in the learning. In an asynchronous version of that algorithm, two adjacent nodes of the network wake up at a given iteration, update their local model based on their local training data, exchange their updates and do the averaging [2, 3, 7, 8].

Drawbacks of these approaches: The centralized approach seems to require a high number of communication rounds. For the case where we choose to send the data, the data needs to go through a direct path leading to node 1 at each iteration. Adding that node 1 needs to have enough computational power to take care of all the iterated updates. The route will be repeated twice in case where the updates are going to be performed locally to exchange the global model forth and back. Then, the communication cost is going to explode remarkably with the network size. Adding issues on privacy, where any node in the network is obliged to share its data or its gradient among other nodes that are not neighbors.

Gossip approach lets all nodes learn the global model including node 1 regardless they are interested in the type of application that they are learning for or not. Then, the gossip algorithm is shown to be slow because each of the local model is improved only when the holder gets the chance to wake up. While on the centralized version, the updates are made on a version of the model that was improved all over the time since starting iterating.

III. Algorithm

We propose to run a random walk over the users when at each round the token holding the last update of the model jumps to one of the neighbors and gets updated according to a gradient descent on this new node training data.

In the end of iterations, at time T, the averaged model $\bar{w}_T$ or the last model w_T will be transmitted to node 1 through a direct path. The MHRW algorithm guarantees a desired stationary distribution while running a Simple Random Walk (SRW) will end up with a stationary distribution proportional to the nodes degrees [9].

For the undirected graph G, the MHRW proposes jumps between nodes based on a SRW first, so the proposal probability for moving from a node v to a node u in V is

$$Q(v, u) = \frac{1}{deg(v)},$$

where $deg(v)$ is the degree of node v. To drive the non-uniform stationary of a SRW to a desired stationary distribution π, a proposed jump from node v to node u can be accepted according to the acceptance probability

$$a(v, u) = \min\left(1, \frac{\pi(u)}{\pi(v)} \frac{Q(u, v)}{Q(v, u)}\right),$$

or rejected with the probability $1 - a(v, u)$. Here, we end up with a transition matrix P of a time reversible Markov chain for the desired stationary distribution π, with

$$\begin{aligned} P(v, u) &= Q(v, u)\, a(v, u) \\ &= \min\left\{Q(v, u), Q(u, v) \frac{\pi(u)}{\pi(v)}\right\}. \end{aligned}$$

At any moment, the active node needs just to know its neighbors, choose one of them uniformly and accept the transition or reject it based on the ratio of its degree to the degree of the chosen node. [9, 10]. Then the new active node i applies

$$w_{t+1} = \mathbf{\Pi}_{\mathcal{W}}(w_t - \gamma_t \nabla f_i(w_t)), \tag{4}$$

where $\mathbf{\Pi}_{\mathcal{W}}$ is the projection operator on $\mathcal{W}$.

The MHRW in graph-based learning problem is discussed for unbiased gradient estimate where the targeted stationary distribution over the nodes training data is uniform [5, 11]. That is Algorithm 1.

Algorithm 1 Uniform Random Walk GD

Initialization: Initial node v_0, Initial model w_0
for $t = 0$ **to** T **do**
  Choose node u uniformly at random from $\mathcal{N}(v_t)$.
  Generate $p \sim U(0, 1)$ where U is the uniform distribution.
  if $p \leq \min\left\{1, \frac{deg(v_t)}{deg(u)}\right\}$ **then**
    $v_{t+1} \leftarrow u$
  else
    $v_{t+1} \leftarrow v_t$
  end if
  $w_{t+1} = \mathbf{\Pi}_{\mathcal{W}}\left(w_t - \gamma_t \nabla f_{v_{t+1}}(w_t)\right)$
end for
Return: w_T and $\bar{w}_T = \frac{\sum_{i=1}^{T}(\gamma_i w_i)}{\sum_{j=1}^{T}\gamma_j}$. {returned to node 1}

Algorithm 2 Weighted Random Walk GD

Initialization: Initial node v_0, Initial model w_0
for $t = 0$ **to** T **do**
  Choose node u uniformly at random from $\mathcal{N}(v_t)$.
  Generate $p \sim U(0, 1)$ where U is the uniform distribution.
  if $p \leq \min\left\{1, \frac{L_u}{L_{v_t}}\frac{deg(v(t))}{deg(u)}\right\}$ **then**
    $v_{t+1} \leftarrow u$
  else
    $v_{t+1} \leftarrow v_t$
  end if
  $w_{t+1} = \mathbf{\Pi}_{\mathcal{W}}\left(w_t - \gamma_t \nabla f_{v_{t+1}}(w_t)\right)$
end for
Return: w_T and $\bar{w}_T = \frac{\sum_{i=1}^{T}(\gamma_i w_i)}{\sum_{j=1}^{T}\gamma_j}$. {returned to node 1}

We propose to reweight the sampling, and let it be proportional to the gradient Lipschitz constant of the node local loss function. We take advantage of the Metropolis algorithm, and redesign the random walk as in Algorithm 2 in order to get that sampling. We use $\sup L$ to denote the supremum of all Lipschitz constants.

Remark 1. *We interchangeably index the local function with he node index or the node name pointing the same local loss on the local data of the node.*

Remark 2. *For the convergence analysis, we are able to get theoretical guarantee on the convergence rate of $f(\bar{w}_T)$. For $f(w_T)$, we are able to prove the convergence. Our simulations show that $f(w_T)$ converges to its optimal value $f(w^*)$ at least as fast as $f(\bar{w}_T)$.*

IV. Convergence Analysis

In this section, we present the results on the convergence of algorithms 1 and 2 in theorems 1 and 2 respectively. For completeness, we start with theorem 1 that analyses Algorithm 1 based on uniform sampling. Theorem 1 can be seen as a modified version of the general result of [5] in which we incorporate the assumption on the Lipschitzness of the local loss function. Our main contribution is in Theorem 2 that analysis the convergence of Algorithm 2 based on weighted sampling that depends on the local Lipschitz constants. Comparing the convergence rates in theorems 1 and 2 we can see the speed up in convergence brought up by our proposed weighted sampling.

Assumption 1. *The local loss function f_i for each node $i \in V$ is a convex function and has a L_i-Lipschitz continuous gradient; that is,*

$$\|\nabla f_i(w_1) - \nabla f_i(w_2)\|_2 \leq L_i \|w_1 - w_2\|_2.$$

Condition 1. *We use a diminishing step size that satisfies*

$$\sum_{t=1}^{\infty} \gamma_t = +\infty,$$

and

$$\sum_{t=1}^{\infty} \ln t.\gamma_t^2 < +\infty.$$

In particular, we can take $\gamma_t = \frac{1}{t^q}$ for $0.5 < q < 1$ as an example that satisfies Condition 1.

Assumption 2. *The Markov chain defined on V with a transition matrix P induced by Algorithm 1 is irreducible and aperiodic and has a stationary distribution π^*.*

We denote by λ_i, for $i = 1, 2, ..N$, the eigenvalues of the reversible transition matrix P with $\lambda_1 = 1$. As we define $\lambda_{max} = \max\{|\lambda_2|, |\lambda_N|\}$ and $\lambda^* = \frac{\lambda_{max}+1}{2}$.

After stating the assumptions on the Markov chain and the local loss functions, we are at the point to state the following theorem.

Theorem 1 (Convergence rate of Algorithm 1)**.** *Under assumptions 1 and 2, running Algorithm 1, with a step size $\gamma_t = \frac{1}{t^q}$ when $0.5 < q < 1$, we get*

$$\mathbb{E}\left(f(\bar{w}_t) - f(w^*)\right) \leq \frac{1}{t^{1-q}}\left(c + (\sup L)^2 c' + \frac{c''}{\ln \frac{1}{\lambda^*}}\right),$$

where, c is a constant the depends on is a constant the depends on $\max_i \|\nabla f_i(w^*)\|_2^2$ noted ϵ, c' depends on the radius R of the bounded set $\mathcal{W}$, and c'' is a constant that depends on the transition matrix P.

Now, we consider sampling the data according to a weighted probability distribution proportional to a weight noted by k. So

$$\pi_k^*(i) \propto k(i)\,\pi^*(i).$$

Similarly to the work in [4], we choose

$$k(i) = \frac{L_i}{\overline{L}},$$

where $\overline{L} = \frac{\sum_{i=1}^{N} L_i}{N}$.

Assumption 3. *The Markov chain defined on V with a transition matrix P_k induced by Algorithm 2 is irreducible and aperiodic and has a stationary distribution π_k^*.*

Theorem 2 (Convergence rate of Algorithm 2)**.** *Under assumptions 1 and 3, running Algorithm 2 with a step size $\gamma_t = \frac{1}{t^q}$ when $0.5 < q < 1$, we get*

$$\mathbb{E}\left(f\left(\bar{w}_t\right) - f\left(w^*\right)\right) \leq \frac{1}{t^{1-q}}\left(d + \left(\overline{L}\right)^2 d' + \frac{d''}{\ln \frac{1}{\lambda_k^*}}\right),$$

where, d is a constant the depends on $\frac{\overline{L}}{\inf L}\epsilon$ for $\inf L$ the infimum over all Lipschitz constants L_i, d' depends on the radius R of the bounded set $\mathcal{W}$, and d'' is a constant that depends on the transition matrix P_k.

Lemma 3 (Convergence of algorithms 1 and 2)**.** *Under assumptions 1, 2 and 3 and with a step size satisfying Condition 1, both algorithms 1 and 2 give the following convergence result*

$$\lim_{t\to\infty} \mathbb{E}\left(f\left(w_t\right) - f\left(w^*\right)\right) = 0.$$

A proof for this lemma follows directly from the work [5].

V. Technical proofs

In order to to get to the proof of theorems 1 and 2 , we need to state some technical lemma that helps for later.

Lemma 4 (Lipschitzness)**.** *If f_i is a convex function on an open subset $\Omega \subseteq \mathbb{R}$, then for a closed bounded subset $\mathcal{W} \subset \Omega$, there exists a constant $D_i \geq 0$, such that, for any $w_1,\ w_2 \in \mathcal{W}$,*

$$\left|f_i\left(w_1\right) - f_i\left(w_2\right)\right| \leq D_i \left\|w_1 - w_2\right\|_2.$$

We define $D = \sup_{i\in V} D_i$. Therefore,

$$\left|f_i\left(w_1\right) - f_i\left(w_2\right)\right| \leq D \left\|w_1 - w_2\right\|_2.$$

A proof for Lemma 4 can be found in [12].

Corollary 1. *For a $D-$Lipschitz convex function f_i, we get*

$$\left\|\nabla f_i\left(x\right)\right\|_2 \leq D.$$

Proof. Given $y = x + \nabla f_i\left(x\right)$

$$\begin{aligned} D\left\|\nabla f_i\left(x\right)\right\|_2 &= D\left\|x - y\right\|_2 \\ &\geq \left|f_i\left(y\right) - f_i\left(x\right)\right| \\ &\geq \left|\left\langle \nabla f_i\left(x\right), \nabla f_i\left(y\right)\right\rangle\right| \\ &= \left\|\nabla f_i\left(x\right)\right\|_2^2. \end{aligned}$$

■

Next, we state our proof of Theorem 1 which is adapted from the proof of [5, Theorem 1] to take into consideration the gradient Lipschitz constant of the local loss function f_i. The importance of this theorem is that it will allow us later to compare the speed up in the convergence rate obtained by the weighted sampling in our main algorithm which is Algorithm 2.

Proof of Theorem 1

$$\begin{aligned} \left\|w_{t+1} - w^*\right\|_2^2 &= \left\|\mathbf{\Pi}_{\mathcal{W}}\left(w_t - \gamma_t \nabla f_{v_t}\left(w_t\right)\right) - \mathbf{\Pi}_{\mathcal{W}}\left(w^*\right)\right\|_2^2 \\ &\overset{(a)}{\leq} \left\|w_t - \gamma_t \nabla f_{v_t}\left(w_t\right) - w^*\right\|_2^2 \\ &= \left\|w_t - w^*\right\|_2^2 - 2\gamma_t \left\langle w_t - w^*, \nabla f_{v_t}\left(w_t\right)\right\rangle \\ &\quad + \gamma_t^2 \left\|\nabla f_{v_t}\left(w_t\right)\right\|_2^2 \\ &= \left\|w_t - w^*\right\|_2^2 - 2\gamma_t \left\langle w_t - w^*, \nabla f_{v_t}\left(w_t\right)\right\rangle \\ &\quad + \gamma_t^2 \left\|\nabla f_{v_t}\left(w_t\right) - \nabla f_{v_t}\left(w^*\right) + \nabla f_{v_t}\left(w^*\right)\right\|_2^2 \\ &\overset{(b)}{\leq} \left\|w_t - w^*\right\|_2^2 - 2\gamma_t \left\langle w_t - w^*, \nabla f_{v_t}\left(w_t\right)\right\rangle \\ &\quad + 2\gamma_t^2 \left\|\nabla f_{v_t}\left(w_t\right) - \nabla f_{v_t}\left(w^*\right)\right\|_2^2 \\ &\quad + 2\gamma_t^2 \left\|\nabla f_{v_t}\left(w^*\right)\right\|_2^2. \end{aligned} \tag{5}$$

(a) follows from $\mathcal{W}$ being a convex closed set, so one can apply nonexpansivity theorem[13], (b) follows from Jensen's inequality applied to the squared norm.

Now, we define a bound on the residual for any $i \in V$,

$$\left\|\nabla f_i\left(w^*\right)\right\|_2 \leq \epsilon < D.$$

Using Lemma 4 and the convexity of the functions f_i, we get

$$\begin{aligned} \left\|w_{t+1} - w^*\right\|_2^2 &\overset{(a)}{\leq} \left\|w_t - w^*\right\|_2^2 - 2\gamma_t \left\langle w_t - w^*, \nabla f_{v_t}\left(w_t\right)\right\rangle \\ &\quad + 2\gamma_t^2 L_{v_t}^2 \left\|w_t - w^*\right\|_2^2 + 2\gamma_t^2 \epsilon^2 \\ &\overset{(b)}{\leq} \left\|w_t - w^*\right\|_2^2 - 2\gamma_t \left\langle w_t - w^*, \nabla f_{v_t}\left(w_t\right)\right\rangle \\ &\quad + 2\gamma_t^2 \left(\sup L\right)^2 \left\|w_t - w^*\right\|_2^2 + 2\gamma_t^2 \epsilon^2. \end{aligned} \tag{6}$$

(a) follows from the Lipschitzness Lemma, (b) follows by bounding by the $\sup L$.

For the next we use the convexity of f_i,

$$\begin{aligned} \left\|w_{t+1} - w^*\right\|_2^2 \leq \left\|w_t - w^*\right\|_2^2 &- 2\gamma_t \left(f_{v_t}\left(w_t\right) - f_{v_t}\left(w^*\right)\right) \\ &+ 2\gamma_t^2 \left(\sup L\right)^2 \left\|w_t - w^*\right\|_2^2 + 2\gamma_t^2 \epsilon^2. \end{aligned} \tag{7}$$

By re-arranging (7), we come to

$$\begin{aligned} \gamma_t \left(f_{v_t}\left(w_t\right) - f_{v_t}\left(w^*\right)\right) \leq \frac{1}{2}\left(\left\|w_t - w^*\right\|_2^2 - \left\|w_{t+1} - w^*\right\|_2^2\right) \\ + \gamma_t^2 \left(\sup L\right)^2 \left\|w_t - w^*\right\|_2^2 + \gamma_t^2 \epsilon^2. \end{aligned} \tag{8}$$

Now summing (8) over t and using Condition 1 and the boundness of $\mathcal{W}$,

$$\begin{aligned} &\sum_t \gamma_t \left(f_{v_t}\left(w_t\right) - f_{v_t}\left(w^*\right)\right) \\ &\quad \leq \frac{1}{2}\left\|w_0 - w^*\right\|_2^2 + \left(\sup L\right)^2 \sum_t \gamma_t^2 \left\|w_t - w^*\right\|_2^2 \\ &\qquad + \sum_t \gamma_t^2 \epsilon^2 < \infty. \end{aligned} \tag{9}$$

Claim 1. Following the same analysis on [5], and using (9), for an integer $\mathcal{T}_t = O\left(\frac{\ln t}{\ln\left(\frac{1}{\lambda^*}\right)}\right)$, we get,

$$\sum_t \gamma_t \mathbb{E}\left(f\left(w_{t-\tau_t}\right) - f\left(w^*\right)\right) \leq C + \left(\sup L\right)^2 C' + \frac{C''}{\ln \frac{1}{\lambda^*}}, \tag{10}$$

and

$$\sum_t \gamma_t \mathbb{E}\left(f\left(w_t\right) - f\left(w_{t-\tau_t}\right)\right) \leq D' + \frac{D''}{\ln \frac{1}{\lambda^*}}. \tag{11}$$

Combining (10) and (11), and applying Jensen's inequality gives the rate of convergence of Algorithm 1.

$$\left(\sum_t \gamma_t\right) \mathbb{E}\left(f\left(\bar{w}_t\right) - f\left(w*\right)\right) \leq \sum_t \gamma_t \mathbb{E}\left(f\left(w_t\right) - f\left(w*\right)\right) \leq c + \left(\sup L\right)^2 c' + \frac{c''}{\ln \frac{1}{\lambda^*}}.$$

For $\gamma_t = \frac{1}{t^q}$ when $0.5 < q < 1$,

$$\mathbb{E}\left(f\left(\bar{w}_t\right) - f\left(w*\right)\right) \leq \frac{1}{\left(\sum_{j=1}^{t} \gamma_j\right)} \left(c + \left(\sup L\right)^2 c' + \frac{c''}{\ln \frac{1}{\lambda^*}}\right) \leq \frac{1}{t^{1-q}} \left(c + \left(\sup L\right)^2 c' + \frac{c''}{\ln \frac{1}{\lambda^*}}.\right). \tag{12}$$

Proof of Theorem 2

To prove Theorem 2, we scale the sampling by the Lipschitz constant L_i of each local loss function f_i. The new local loss in the case of the weighted loss is re-expressed as $f_i^k = \frac{f_i}{k(i)}$ where $k\left(i\right)$ is the normalized weight taken as $k\left(i\right) = \frac{L_i}{\bar{L}}$. So $\sup_i L_i^k = \sup_i \frac{L_i \bar{L}}{L_i} = \bar{L}$. One way to show the outperformance of the Weighted RW SGD, is to consider the case where $\epsilon \approx 0$.

VI. Numerical Results

In this section, we report our numerical results applying both algorithms 1 and 2 to a decentralized least squares problem and compare it to the asynchronous Gossip SGD for a same number of communication rounds.

We have

$$f\left(w\right) = \frac{1}{2} \sum_{i=1}^{N} \left(x_i^T w - y_i\right)^2.$$

So the local loss function for node i is $f_i = \frac{N}{2}\left(x_i^T w - y_i\right)$. Then, for such local loss function the Lipschitz constant is $L_i = N \left\|x_i\right\|_2^2$. We construct a network of N nodes where the edges are constructed according to 3 different graph structures ensuring connectivity showing for a network of 20 nodes the effect of the structure involved through λ^*. In the Erdos-Renyi graph, the graph is uniform in average, this should help to emphasize the effect of the weighted sampling over the uniform sampling regardless the structure. On this graph, we compare the value of the averaged loss f on different outputs: w_t and $\bar{w}_t$. We find out the $f\left(w_t\right)$ converges faster to the optimal value f^* so we decide to continue the simulations on w_t. The

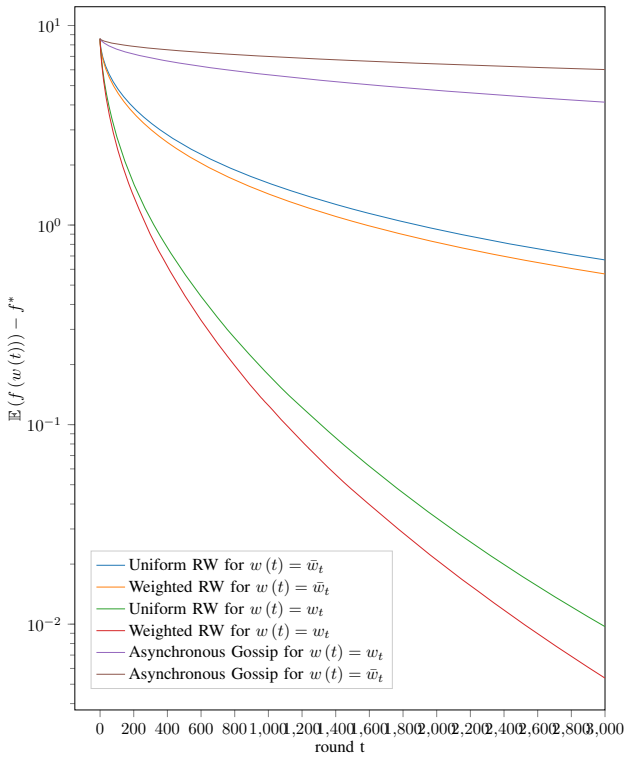

Fig. 1. Comparison of the Uniform RW SGD, Weighted RW SGD and the Gossip SGD on a chordal cycle graph of 20 nodes and 60 edges.

Stochastic Block Model graph with 4 communities is a good example for learning over training data distributed over social network users with friendship connections.

For the simulations, we choose $d = 3$ as features dimension. We sample the entries of the global model w^* independently from the Gaussian distribution $\mathcal{N}\left(0, 1\right)$. For the data, we sample x_i from $\mathcal{N}\left(0, I_3\right)$.Then, the label is $y_i = x_i^T w^*$.

References

[1] H. B. McMahan, E. Moore, D. Ramage, and B. A. y Arcas, "Federated learning of deep networks using model averaging," *CoRR*, vol. abs/1602.05629, 2016.

[2] K. Yuan, Q. Ling, and W. Yin, "On the convergence of decentralized gradient descent," *SIAM Journal on Optimization*, vol. 26, pp. 1835–1854, 2016.

[3] J. C. Duchi, A. Agarwal, and M. J. Wainwright, "Dual averaging for distributed optimization: Convergence analysis and network scaling," *IEEE Transactions on Automatic Control*, vol. 57, pp. 592–606, 2012.

[4] D. Needell, R. Ward, and N. Srebro, "Stochastic gradient descent, weighted sampling, and the randomized kaczmarz algorithm," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 1017–1025.

[5] T. Sun, Y. Sun, and W. Yin, "On Markov Chain Gradient Descent," *arXiv e-prints*, Sep. 2018.

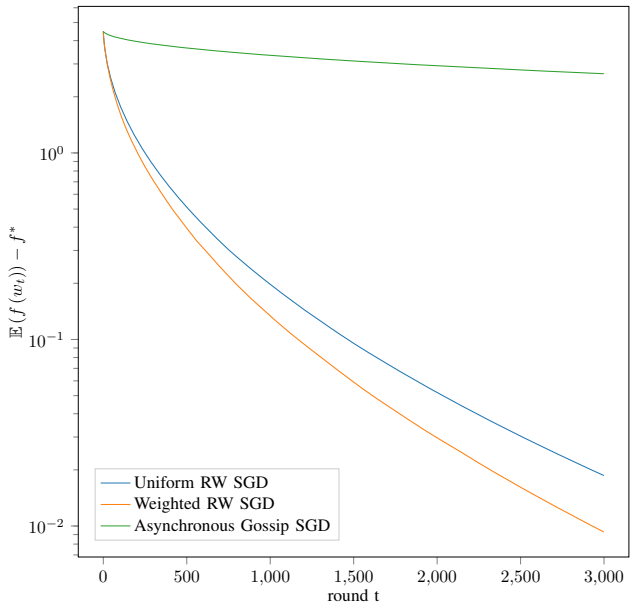

Fig. 2. Comparison of the Uniform RW SGD, Weighted RW SGD and the Gossip SGD on a chordal cycle graph of 20 nodes and 60 edges.

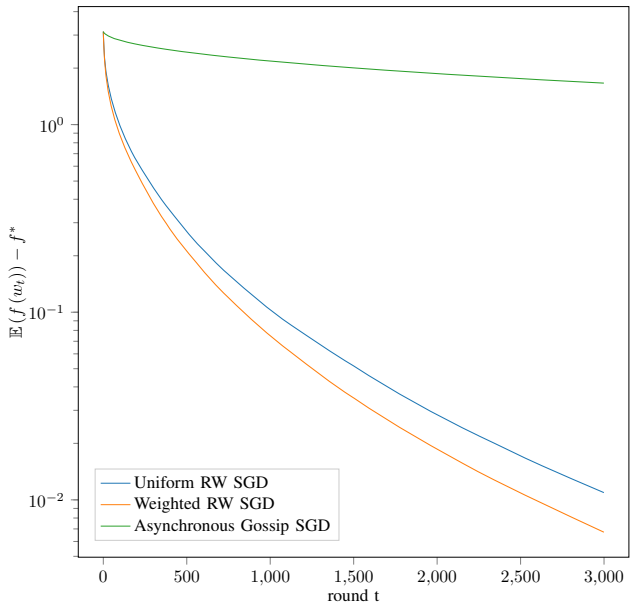

Fig. 3. Comparison of the Uniform RW SGD, Weighted RW SGD and the Gossip SGD on a stochastic block model graph of 4 clusters with 5 nodes in each.

[6] M. Li, D. G. Andersen, J. W. Park, A. J. Smola, A. Ahmed, V. Josifovski, J. Long, E. J. Shekita, and B.-Y. Su, "Scaling distributed machine learning with the parameter server," in *Proceedings of the 11th USENIX Conference on Operating Systems Design and Implementation*, ser. OSDI'14. Berkeley, CA, USA: USENIX Association, 2014, pp. 583–598.

[7] A. Nedic and A. E. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, pp. 48–61, 2009.

[8] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, "Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent," in *NIPS*, 2017.

[9] C.-H. Lee, X. Xu, and D. Y. Eun, "Beyond random walk and metropolis-hastings samplers: why you should not backtrack for unbiased graph sampling," in *SIGMETRICS*, 2012.

[10] D. Levin and Y. Peres, *Markov Chains and Mixing Times: Second Edition*, ser. MBK. American Mathematical Society, 2017.

[11] S. S. Ram, A. Nedic, and V. V. Veeravalli, "Incremental stochastic subgradient algorithms for convex optimization," *SIAM Journal on Optimization*, vol. 20, pp. 691–717, 2009.

[12] "Every convex function is locally lipschitz," *The American Mathematical Monthly*, vol. 79, no. 10, pp. 1121–1124, 1972. [Online]. Available: http://www.jstor.org/stable/2317434

[13] M. Dattorro and J. Dattorro, "Convex optimization euclidean distance geometry," 2005.