On-Line Learning Gossip Algorithm in Multi-Agent Systems with Local Decision Rules

Pascal Bianchi, Stéphan Clémençon, Gemma Morral
Institut Mines-Télécom
LTCI UMR 5141, Télécom ParisTech & CNRS
l37-39 rue Dareau, 75014 Paris, France
pascal.bianchi@telecom-paristech.fr
stephan.clemencon@telecom-paristech.fr
gemma.morral@telecom-paristech.fr

J. Jakubowicz
Institut Mines-Télécom
SAMOVAR UMR 5157, Télécom SudParis & CNRS
9 rue Charles Fourier, 91000 Evry, France
jeremie.jakubowicz@telecom-sudparis.com

***Abstract*—This paper is devoted to investigate binary classification in a distributed and on-line setting. In the Big Data era, datasets can be so large that it may be impossible to process them using a single processor. The framework considered accounts for situations where both the training and test phases have to be performed by taking advantage of a network architecture by the means of local computations and exchange of limited information between neighbor nodes. An online learning gossip algorithm (OLGA) is introduced, together with a variant which implements a node selection procedure. Beyond a discussion of the practical advantages of the algorithm we promote, the paper proposes an asymptotic analysis of the accuracy of the rules it produces, together with preliminary experimental results.**

***Keywords*-online statistical learning; distributed learning algorithm; gossip algorithm**

I. Introduction

In most analyses carried out in the field of statistical learning theory, the practical constraints related to the data acquisition/storage/access system and inherent to processing speed, memory and computing capacity are generally ignored or incorporated into the mathematical framework in a very stylized manner so far. With the advent of highly complex digital network infrastructures and the pressing necessity of sharing resources and distributing computing power (1; 2), this facet of the machine-learning environment is however becoming more and more essential from a technological perspective and is receiving now increasing attention, see (3; 4; 5; 6; 7; 8; 9; 10; 11; 12) for instance. Motivated by the recent developments in the architecture of data repositories and computer systems, it is the main purpose of this paper to investigate the binary classification problem, the "flagship" problem in statistical learning, in a *distributed* and *on-line* context, accounting for certain real-life situations, possibly more and more currently encountered in the near future.

Throughout the article, we consider the case where the training data are not stored in some central memory but split into distinct *clusters*, individually processed by independent *agents* (*e.g.* processors). To process Big Data, one generally distribute data subsamples over a network of processors communicating with each other. Precisely, it is assumed that the agents can exchange a limited amount of information per unit of time only, through a communication structure modeled by a graph of which they form the nodes. Hence, due to these capacity constraints, merging all training sets at any node is unfeasible and a distributed approach, limiting the network overhead, is required. Here, by "distributed", it is meant that both the learning and prediction stages are performed by the means of local computations of the agents and sparse communications between them: each agent simultaneously processes the data set it has been assigned to and shares some information with its neighbors in order to build a *local classifier*.

In (7; 8), a specific view to distributed learning has been developed, where the goal is to reach a *consensus* among local classifiers. In this setting, all agents originally dispose of the same collection of classification rules and a *local gradient descent* technique, jointly performed with a *gossip* step, is used to drive them to a consensus. At the end of the learning procedure, all agents use the consensus classifier to predict labels assigned to test data, with no need for further communications. The nature of the problem we investigate through this paper is very different, it is not of the type "distributed consensus". It should be noticed that, unlike most works on "distributed classification", agents are *not* assumed exchangeable in the framework we consider. First, we assume that the collection of classifiers may vary from an agent to another. This situation encompasses the case where each agent is an expert in the recognition based on a specific feature of the input observation for instance. Additionally, the issue at stake is not to seek to achieve a consensus between the agents but to learn how to aggregate efficiently the local decisions, typically through a a majority vote or a well-chosen weighted average. Hence, in the classification problem we consider, both learning and test phases require distributed computations, relying on the whole network of

agents. In addition, it is expected that, unlike consensus-based approaches that drive all nodes to a common classifier, our scheme should preserve and take full advantage of the peculiar skills of the local classifiers, being therefore closer to the spirit of *ensemble learning* algorithms.

The paper is organized as follows. Section II describes the specific framework of the learning problem considered. In section III, the principles of the algorithm promoted are described at length. The performance of the procedure proposed is analyzed in section IV, while section V focuses on a specific situation. Finally, numerical experiments are displayed in section VI, in order to provide some preliminary empirical evidence of the efficiency of the methods proposed in this paper. Section VII collects some concluding remarks and technical details are deferred to the Appendix section.

II. Background

We start off with setting out the notations and describing the key ingredients of the learning problem subsequently analyzed. Here and throughout, the indicator of any event $\mathcal{E}$ is denoted by $\mathbb{I}\{\mathcal{E}\}$.

A. Objective

Suppose we have a "black-box" system where Y is a binary output, taking its values in $\{-1,\ +1\}$ say, and X is an input random vector valued in a high-dimensional space X, modeling some (hopefully) useful observation for predicting Y. Based on training data, the goal is to build a prediction rule $\mathrm{sign}(h(X))$, where $h : \mathsf{X} \to \mathbb{R}$ is some measurable function, which minimizes the risk

$$R_\varphi(h) = \mathbb{E}\left[\varphi(-Yh(X))\right],$$

where expectation is taken over the unknown distribution of the pair of r.v.'s (X, Y) and $\varphi : \mathbb{R} \to [0,\ +\infty)$ denotes a cost function (*i.e.* a measurable function such that $\varphi(u) \geq \mathbb{I}\{u \geq 0\}$ for any $u \in \mathbb{R}$). For reasons which will appear obvious in the sequel (see Remark 3), we focus on the cost function $\varphi(u) = (u+1)^2/2$. Notice that, in this case, the optimal decision function is given by: $\forall x \in \mathsf{X}$, $h^*(x) = 2\mathbb{P}\{Y = +1 \mid X = x\} - 1$. The classification rule $H^*(x) = \mathrm{sign}(h^*(x))$ thus coincides with the naive Bayes classifier. For this specific choice, decision function candidates $h(x)$ will be assumed to be square integrable with respect to X's distribution. The learning environment under study is non standard. Here we consider a model of *distributed classification device* composed of a set $\mathcal{V}$ of $N \geq 1$ connected agents, which process independent databases: each agent $v \in \mathcal{V}$ disposes of a training dataset $\mathcal{D}_v = \{(X_{1,v}, Y_{1,v}),\ \ldots,\ (X_{n_v,v}, Y_{n_v,v})\}$ of size $n_v \geq 1$ and made of independent copies of the pair (X, Y). In addition, each agent $v \in \mathcal{V}$ must select a *weak classifier function* among a given parametric class possibly depending on v, namely $\{h_v(\cdot,\ \theta_v)\}_{\theta_v \in \mathbb{R}^{d_v}}$, where $d_v \geq 1$. We set $D = \sum_v d_v$. For any vector $\boldsymbol{\theta} = (\theta_1, \cdots, \theta_N) \in \boldsymbol{\Theta} = \mathbb{R}^{d_1} \times \cdots \times \mathbb{R}^{d_N}$, we define the *global (soft) classifier* as:

$$H(x, \boldsymbol{\theta}) = \sum_{v \in \mathcal{V}} h_v(x, \theta_v) \ \text{for} x \in \mathsf{X},$$

the label related to an observation X being estimated by $\mathrm{sign}(H(X, \boldsymbol{\theta}))$. To lighten notation, we set $R_\varphi(\boldsymbol{\theta}) = R_\varphi(H(\cdot, \boldsymbol{\theta}))$. This paper investigates the problem of finding a "global classification rule", as defined above, with minimum risk, *i.e.* the optimization problem

$$\min_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \ R_\varphi(\boldsymbol{\theta}), \tag{1}$$

while fulfilling some capacity constraints, which shall be described in the next subsection.

Remark 1. (MIXTURE OF EXPERTS.) *A typical example of the framework above stipulates that a fixed weak classifier $\overline{h}_v : \mathsf{X} \to \{-1, +1\}$ is assigned to each agent v. For any $(\theta_v, x) \in \mathbb{R} \times \mathsf{X}$, we set $h_v(x, \theta_v) = \theta_v \overline{h}_v(x)$ and the global classifier then reduces to a weighted sum of the local weak classifiers. In the learning phase, the issue is to determine the optimal weights using a distributed algorithm. In the test phase, it is compute a weighted sum of the local decisions, by using standard* average consensus *algorithms such as those studied in (13) for instance.*

Remark 2. (MAJORITY VOTE.) *Another useful example is given by the case where each θ_v corresponds to some local parameter of a local classifier $x \mapsto h_v(x, \theta_v) \in \mathbb{R}$. In this case, the global classifier output* $\mathrm{sign}(H(x, \boldsymbol{\theta}))$ *can be evaluated by a simple majority vote between agents, see (14).*

B. Distributed Learning

In order to give an insight into the approach we propose, we consider first the ideal case where a standard gradient descent for solving (1) could be applied. One would then generate in an iterative manner a sequence $\boldsymbol{\theta}^{(t)} = (\theta_1^{(t)}, \cdots, \theta_N^{(t)})$, $t \geq 1$, satisfying the following update equation for each $v \in \mathcal{V}$:

$$\theta_v^{(t+1)} = \theta_v^{(t)} + \gamma_t \, \mathbb{E}\left[Y \nabla_v h_v(X, \theta_v^{(t)}) \varphi'(-Y H(X, \boldsymbol{\theta}^{(t)}))\right], \tag{2}$$

where $\gamma_t > 0$ is a step size and ∇_v represents the gradient operator w.r.t. the argument θ_v. Naturally, as (X, Y)'s distribution is unknown, the expectation involved in (2) cannot be computed and must be replaced by a statistical version, in accordance with the *Empirical Risk Minimization* paradigm. It is assumed that each agent $v \in \mathcal{V}$ must rely on the local dataset $\mathcal{D}_v$ only to update its estimate, in a *one-pass* fashion: each observation $(X_{k,v}, Y_{k,v})$ must be used only once by agent v and is not stored into the agent's memory. This "on-line" framework is especially relevant in the context of large data sets, where it is generally hopeless to process the whole training sequence as a block. It shall

also be revealed useful in a distributed optimization setting, as will be discussed later on. The expectation in (2) can be then replaced by the following *unbiased* estimate

$$Y_{t+1,v}\nabla_v h_v(X_{t+1,v},\theta_v^{(t)})\varphi'(-Y_{t+1,v}H(X_{t+1,v},\boldsymbol{\theta}^{(t)})).$$

The second issue is related to the distributed setting. In the estimate of the gradient above, the evaluation of the quantity $H(X_{t+1,v},\boldsymbol{\theta}^{(t)})$ requires that: i) agent v sends the input $X_{t+1,v}$ to *all* the other nodes $w \neq v$, ii) each node w computes its local decision $h_w(X_{t+1,v},\theta_w^{(t)})$ and returns the result to node v. Needless to say, such a procedure can be revealed overwhelmingly complex when the number of nodes is significant and/or when the dimensionality of the input X is large. It is therefore crucial to reduce the amount of information exchanged in the network. To formalize this constraint, we define the *network throughput* τ as the average number of information bits successfully carried by the network during each unit of time. Formally, we require that the sum over all pairs of agents (v,w) of the number bits send by v to w does not exceed τ in expectation.

III. The Online Learning Gossip Algorithm (OLGA)

We now describe at length the general algorithm we propose, in order to solve the constrained optimization problem (1) in the general *on-line* and *distributed* framework described in section II. Suppose that, at step $t \geq 1$, for each agent $v \in \mathcal{V}$, the current parameter value is $\theta_{t,v}$. Set $\boldsymbol{\theta}_t = (\theta_{t,1},\cdots,\theta_{t,N})$. The update is performed as follows. Agent v observes the pair $(X_{t+1,v},Y_{t+1,v})$ and evaluates its local decision $h_v(X_{t+1,v},\theta_{t,v})$ using the former value of the parameter $\theta_{t,v}$. Next, it searches for an estimate of the global decision $H(X_{t+1,v},\theta_{t,v})$ as follows, by selecting some neighbors at random and sending its training input $X_{t+1,v}$ to the selected nodes. Let $\boldsymbol{\delta}_{t+1,v} = \{\delta_{t+1,v}^w\}_{w\in\mathcal{V},w\neq v}$ be a collection of $N-1$ independent Bernoulli r.v.'s $\mathcal{B}(p)$, with parameter $p \in (0,1]$, independent from $X_{t+1,v}$ given $\boldsymbol{\theta}_t$. Agent v sends the input $X_{t+1,v}$ to node w if and only if $\delta_{t+1,v}^w = 1$. An estimate of the global decision is then given by:

$$\hat{Y}_{t+1,v}^{(\mathcal{V})} = h_v(X_{t+1,v},\theta_{t,v}) + p^{-1}\sum_{w\in\mathcal{V}\setminus\{v\}} \delta_{t+1,v}^w\, h_w(X_{t+1,v},\theta_{t,w}) \ , \quad (3)$$

where the superscript emphasizes the fact that the estimate is computed by means of communications in the whole network $\mathcal{V}$. It is worth noticing that (3) is an unbiased estimate of the global decision in the sense that $\mathbb{E}\left[\hat{Y}_{t+1,v}^{(\mathcal{V})}|X_{t+1,v},\boldsymbol{\theta}_t\right] = H(X_{t+1,v},\theta_{t,v})$. If B represent the number of bits required to represent an arbitrary input $x \in \mathsf{X}$, then each link $v \to w$ carries in average pB bits per unit of time. In order not to exceed the network throughput τ, one must pick the sampling parameter p so that $p \leq \tau/BN(N-1)$. Finally, agent v performs a local gradient descent as follows:

$$\theta_{t+1,v} = \theta_{t,v} + \gamma_t\, Y_{t+1,v}\nabla_v h_v(X_{t+1,v},\theta_{t,v})\varphi'(-Y_{t+1,v}\hat{Y}_{t+1,v}^{(\mathcal{V})}) \ . \quad (4)$$

As mentioned above, we shall pay a particular attention to the case $\varphi(x) = \frac{1}{2}(x+1)^2$. In that case, the update equation (4) boils down to:

$$\theta_{t+1,v} = \theta_{t,v} + \gamma_t\,\nabla_v h_v(X_{t+1,v},\theta_{t,v})(Y_{t+1,v} - \hat{Y}_{t+1,v}^{(\mathcal{V})}) \ . \quad (5)$$

Remark 3. (On the cost function.) *The quadratic nature of the cost functional is essential in the subsequent analysis. It guarantees that the OLGA output remains unbiased at each iteration, in spite of its on-line nature and the randomness incurred by the gossip phase.*

The algorithm is summarized in Table 1 below.

Algorithm 1 OLGA

Initialize: Set arbitrary initial values $\theta_{0,v}$ for each node $v \in \mathcal{V}$.

Update: At each time $t = 1,2,\cdots$ **do**

For each $v \in \mathcal{V}$ **do**

Neighbors selection: Draw independent Bernoulli r.v.'s $\delta_{t+1,v}^w \sim \mathcal{B}(p)$ for any $w \neq v$

Gossip step:

Transmit $X_{t+1,v}$ to all w such that $\delta_{t+1,v}^w = 1$ and obtain $h_w(X_{t+1,v},\theta_{t,w})$ in return

Local descent: Update the parameter value $\theta_{t+1,v}$ using (4)

As the algorithm is *single-pass*, the number of iterations is necessary smaller than the size of the full data sample, $n = \sum_{v\in\mathcal{V}} n_v$. Hence, in the asymptotic framework that stipulates $t \to +\infty$, it is implicitly assumed that $n \to +\infty$.

IV. Performance Analysis

In this section, we investigate the asymptotic behavior of the predictor output by OLGA as $t \to \infty$. First, we establish its almost-sure convergence to the set of minimizers of $R_\varphi(\boldsymbol{\theta})$. Next, we provide a Central Limit Theorem which characterizes the fluctuations of the excess of risk as $t \to \infty$. This result determines the convergence rate of the algorithm and explicitly characterizes the impact of the "sparsifying" parameter p on the performance of the algorithm. Finally, using the results of (15), we provide a uniform bound on the error probability of the proposed classifier.

The following assumption is rather standard in stochastic approximation.

Assumption 1. *The step size γ_t decays to 0 as $t \to \infty$, so that: $\sum_{t\geq 1}\gamma_t = \infty$ and $\sum_{t\geq 1}\gamma_t^2 < \infty$.*

Additionally, some classical regularity conditions on the weak classifier functions h_v are required.

Assumption 2. *The conditions below hold true for any $v \in \{1, \cdots, N\}$ and any compact set $\mathcal{K} \subset \mathbb{R}^{d_v}$.*

(a) For any $x \in \mathsf{X}$, the function $\theta_v \mapsto h_v(x, \theta_v)$ is continuously differentiable.

(b) For any $\theta_v \in \mathbb{R}^{d_v}$, $\mathbb{E}\left[h_v(X, \theta_v)^2\right] < \infty$.

(c) We have:

$$\begin{aligned} \mathbb{E}\left[\sup_{\theta_v \in \mathcal{K}} \|\nabla_v h_v(X, \theta_v)\|^2\right] &< \infty, \\ \mathbb{E}\left[\sup_{\theta_v \in \mathcal{K}} \left\|\nabla_v h_v^2(X, \theta_v)\right\|\right] &< \infty. \end{aligned}$$

(d) The mappings $\theta_v \mapsto h_v(X, \theta_v)$ and $\theta_v \mapsto \nabla_v h_v(X, \theta_v)$ on $L^2(\mathbb{P})$ are both continuous.

(e) We have:

$$\begin{aligned} \sup_{\theta_v \in \mathcal{K}} \mathbb{E}\left[\|\nabla_v h_v(X, \theta_v)\|^4\right] &< \infty, \\ \sup_{\theta_v \in \mathcal{K}} \mathbb{E}\left[h_v^4(X, \theta_v)\right] &< \infty. \end{aligned}$$

(f) The set of stationary points

$$\mathcal{L} = \{\boldsymbol{\theta} : \nabla R_\varphi(\boldsymbol{\theta}) = 0\}$$

is finite.

Assumption 2 is clearly satisfied in the example described in Remark 1, *i.e.* when $h_v(x, \theta_v) = \theta_v \overline{h}_v(x)$ for some fixed local weak classifier $\overline{h}_v$ such that the fourth moment of $\overline{h}_v(X)$ is finite.

Recall that the algorithm is said to be *stable* if there exists a compact set $\mathcal{K} \subset \mathbb{R}^{d_v}$ such that the sequence $(\theta_{t,v})_{t \geq 1}$ remains in $\mathcal{K}$ for any $v \in \mathcal{V}$, with probability one: $\mathbb{P}\left\{\exists K > 0,\ \sup_{t \geq 1} \|\boldsymbol{\theta}_t\| < K\right\} = 1$. The next result reveals that, provided that it is stable, the algorithm produces a consistent decision rule as the number of iterations grows to infinity.

Theorem 1. (CONSISTENCY) *Assume that the algorithm is stable. Under Assumptions 1 and 2, the sequence $(\boldsymbol{\theta}_t)_{t \geq 1}$ almost-surely converges to the set of stationary points $\mathcal{L}$ of R_φ.*

The stability condition may not be easy to check in practice. There are several ways to guarantee stability. A possible approach is to confine the sequence to a predetermined bounded set. This can be achieved by introducing a projection step at each iteration of the stochastic gradient algorithm. Each time an estimate $\theta_{t,v}$ falls outside some convex compact set $\mathcal{K}_v$, agent v brings the estimate back into $\mathcal{K}_v$ by replacing $\theta_{t,v}$ with the nearest point in $\mathcal{K}_v$. In that case, differential inclusion arguments may show that the conclusions of Theorem 1 remain true: $\boldsymbol{\theta}_t$ converges to the set of Karush-Kuhn Tucker points of the functional $R_\varphi(\boldsymbol{\theta})$ over the set $\prod_v \mathcal{K}_v$. Refer to (16) or (17) for further details on projected stochastic approximation algorithms. Alternatively, one can stipulate additional assumptions for the weak classifier functions, see for instance (18). The following result focuses on the situation described in Remark 1.

Theorem 2. (CONSISTENCY (BIS)) *Suppose that, for all $v \in \mathcal{V}$, $h_v(x, \theta_v) = \theta_v \overline{h}_v(x)$ for some given function $\overline{h}_v$ such that $\mathbb{E}[(\overline{h}_v(X))^4] < \infty$. Then, OLGA is stable and the sequence $(\boldsymbol{\theta}_t)_{t \geq 1}$ almost-surely converges to the set of minimizers of R_φ as $t \to +\infty$.*

In the sequel, notation ∇^2 (resp. ∇_v^2) denotes the Hessian operator w.r.t. $\boldsymbol{\theta}$ (resp. θ_v). We also use notation ∇_v^1 for ∇_v, and ∇_v^0 stands for the identity *i.e.*, $\nabla_v^0 f(\theta_v) = f(\theta_v)$. Superscript ^{T} represents transposition. Let $\boldsymbol{\theta}^\star = (\theta_1^\star, \cdots, \theta_N^\star)$ be an arbitrary point. We make the following assumption.

Assumption 3. *Suppose that $\boldsymbol{\theta}^\star \in \mathcal{L}$ and that the following conditions hold true for any $v \in \mathcal{V}$.*

(a) There exists a neighborhood $\mathcal{N}_v$ of $\theta_v^\star$ such that for any $x \in \mathsf{X}$, function $\theta_v \mapsto h_v(x, \theta_v)$ is twice continuously differentiable on $\mathcal{N}_v$.

(b) We have: $\mathbb{E}\left[\left\|\nabla_v^2 h_v(X, \theta_v^\star)\right\|^2\right] < \infty$ where $\|\,.\,\|$ represents any matrix norm.

(c) There exists a square-integrable random variable $C(X)$ s.t. for any $i \in \{0, 1, 2\}$ and $\theta_v \in \mathcal{N}_v$,

$$\left\|\nabla_v^i h_v(X, \theta_v) - \nabla_v^i h_v(X, \theta_v^\star)\right\| \leq C(X) \left\|\theta_v - \theta_v^\star\right\| \ .$$

(d) The matrix

$$\begin{aligned} Q^\star = \mathbb{E}\left[\nabla H(X, \boldsymbol{\theta}^\star) \nabla^T H(X, \boldsymbol{\theta}^\star)\right] \\ + \mathbb{E}\left[(H(X, \boldsymbol{\theta}^\star) - Y) \nabla^2 H(X, \boldsymbol{\theta}^\star)\right] \end{aligned}$$

is a Hurwitz matrix, i.e. the largest real part of its eigenvalues is $-L$ with $L > 0$.

(e) There exists $b > 4$ such that for any $i \in \{0, 1\}$,

$$\sup_{\theta_v \in \mathcal{N}_v} \mathbb{E}\left[\left\|\nabla_v^i h_v(X, \theta_v)\right\|^b\right] < \infty.$$

(f) The mapping $\boldsymbol{\theta} \mapsto \Gamma_v(\boldsymbol{\theta})$ is continuous at point $\boldsymbol{\theta}^\star$, where $\Gamma_v(\boldsymbol{\theta})$ is defined by:

$$\begin{aligned} \mathbb{E}\left[(H(X, \boldsymbol{\theta}) - Y)^2 \nabla_v h_v(X, \theta_v) \nabla_v^T h_v(X, \theta_v)\right] \\ + \frac{1-p}{p} \times \\ \sum_{w \in \mathcal{V} \setminus \{v\}} \mathbb{E}\left[h_w(X, \theta_w)^2 \nabla_v h_v(X, \theta_v) \nabla_v^T h_v(X, \theta_v)\right] \ . \end{aligned} \tag{6}$$

(g) The block-diagonal matrix $\Gamma^\star = \mathrm{diag}(\Gamma_v(\boldsymbol{\theta}^\star))_{v \in \mathcal{V}}$ is positive definite.

Observe that the mapping (6) is well-defined in a neighborhood of $\boldsymbol{\theta}^\star$, by virtue of Assumption 3(e).

Theorem 3. (A CONDITIONAL CLT) *Suppose that Assumptions 2 and 3 hold true and that $\gamma_t = \gamma_0 t^{-\alpha}$ for some*

constants $\gamma_0 > 0$ *and* $\alpha \in (1/2, 1]$. *When* $\alpha = 1$, *take* $\gamma_0 > (2L)^{-1}$ *and* $\eta = 1/(2\gamma_0)$. *Otherwise, set* $\eta = 0$. *Conditioned upon the event* $\{\lim_{t\to\infty} \boldsymbol{\theta}_t = \boldsymbol{\theta}^\star\}$, *the sequence* $\sqrt{\gamma_t}(\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star)$ *converges in distribution to a centered Gaussian distribution* $\mathcal{N}(0, \Sigma)$ *whose covariance matrix* Σ *is the unique solution to the Lyapunov equation:* $(Q^\star - \eta I)\Sigma + \Sigma(Q^\star - \eta I)^T = \Gamma^\star$.

Theorem 3 still holds true under milder assumptions on the step size, see (19) for more general conditions. The effect of the Bernoulli sampling parameter p on the asymptotic behavior of the estimation error deserves some attention. The case $p = 1$ corresponds to a centralized setting where all nodes communicate without restriction at any time. The matrix $\Gamma_v(\boldsymbol{\theta}^\star)$ then boils down to the first term in (6) solely. This gives the insight that the second term of (6) corresponds to the *additional noise covariance* induced by the distributed setting, as opposed to a centralized situation. In effect, when p becomes close to zero *i.e.* when communications become rare, the second term of (6) becomes significant and produces a dramatic increase of the asymptotic covariance matrix Σ. In that sense, Theorem 3 quantifies the unavoidable tradeoff between *throughput* and *accuracy*.

Corollary 1. (ERROR RATE) *Let* U *be a* $D \times 1$ *vector of independent centered Gaussian r.v.'s with unit variance. Under Theorem 3's assumptions and conditioned upon the event* $\{\lim_{t\to\infty} \boldsymbol{\theta}_t = \boldsymbol{\theta}^\star\}$, $\gamma_t(R_\varphi(\boldsymbol{\theta}_t) - R_\varphi(\boldsymbol{\theta}^\star))$ *converges in distribution to the noncentral* χ^2 *r.v.* $\frac{1}{2}U^T\Sigma^{1/2}Q^\star\Sigma^{1/2}U$.

Remark 4. (ON THE COST FUNCTION (BIS).) *We recall that the excess of probability of error of a classifier* $\mathrm{sign}(H(x))$ *is bounded by* $(R_\varphi(H) - R_\varphi^*)^{1/2}$, *see (15). However, the damage to the rate of the excess of risk caused by the use of a quadratic convex surrogate for the cost is somehow compensated by the (possibly parametric) rate stated in Corollary 1.*

V. DISTRIBUTED SELECTION

This section investigates more specifically the situation where for any $v \in \mathcal{V}$, $h_v(x, \theta_v) = \alpha_v \, \overline{h}_v(x, \beta_v)$, the local parameter θ_v being of the form $\theta_v = (\alpha_v, \beta_v)$ with $\alpha_v \in \mathbb{R}$, $\beta_v \in \mathbb{R}^{d_v - 1}$ and $\overline{h}_v : \mathsf{X} \times \mathbb{R}^{d_v-1} \to \mathbb{R}$ being a local weak classifier function. For any agent v, the aim is to jointly determine the value of β_v parametrizing the local classifier and the *weight* α_v of agent v in the sum:

$$H(x, \boldsymbol{\theta}) = \sum_{v\in\mathcal{V}} \alpha_v \, \overline{h}_v(x, \beta_v) \ , \tag{7}$$

for $\boldsymbol{\theta} = (\theta_1, \cdots, \theta_N)$. In this scenario, it is natural to include a nonnegativity constraint on the weights: $\alpha_v \geq 0$ for any $v \in \mathcal{V}$. Clearly, the vector $\boldsymbol{\theta}$ can be achieved by using a distributed algorithm as proposed in Section III. However, when the number N of nodes is very large, the implementation of OLGA can be difficult or even unfeasible. Indeed, in the learning phase, a significant amount of information should be exchanged by all nodes and, in the test phase, all N nodes are involved in the decision process. It is therefore desirable to restrict the number of nodes in order to simplify the optimization stage and the prediction process both at the same time. This remark is also motivated by the fact that in practice, different nodes might generate quite similar outputs. For such nodes, it is useless to duplicate the information in the sum (7). In this section, we propose an online method to jointly *i)* learn the parameters $\boldsymbol{\theta}$ and *ii)* withdraw the nodes which are not essential for classification. Note that the withdrawal of a node v can be achieved by setting α_v to zero in (7). Based on this remark, we propose to include a ℓ^1-penalization term to the initial cost function. For some fixed constant $\lambda > 0$, this yields the following optimization problem:

$$\min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} R_\varphi(\boldsymbol{\theta}) + \lambda \sum_{v\in\mathcal{V}} |\alpha_v| \ , \tag{8}$$

The "lasso" penalization term $\sum_v |\alpha_v|$ is introduced in order that the minimizers exhibit a certain level of sparsity, *i.e.* are such that a significant number of coefficients α_v are exactly equal to zero. Here, λ is a tuning parameter which allows to set the tradeoff between the minimization of the cost and the sparsity of the minimizers. The following modifications should be brought to OLGA in order to produce an efficient distributed on-line algorithm for selecting the relevant experts. At each iteration t, we assume that certain nodes have been definitively declared as idle, and we denote by $\mathcal{S}_t \subset \mathcal{V}$ the remaining subset of active nodes. Following in the footsteps of the approach described in section III, a given active node $v \in \mathcal{S}_t$ observing a pair of the training sample $(X_{t+1,v}, Y_{t+1,v})$ can obtain a noisy estimation of the output classifier by *i)* drawing independent Bernoulli distributed r.v.'s $\{\delta_{t+1,v}^{(w)}\}_{w\in\mathcal{S}_t\setminus\{v\}}$ with parameter $p_t \in (0, 1]$, *ii)* computing the sum:

$$\hat{Y}_{t+1,v}^{(\mathcal{S}_t)} = h_v(X_{t+1,v}, \theta_{t,v}) + p_t^{-1} \sum_{w\in\mathcal{S}_t\setminus\{v\}} \delta_{t+1,v}^w \, h_w(X_{t+1,v}, \theta_{t,w}) \ .$$

The Bernoulli parameter p_t should be chosen in a way that the network throughput does not exceed τ. Thus, if $|\mathcal{S}_t|$ denotes the cardinal of the set $\mathcal{S}_t$ and B the number of bits required to represent an arbitrary input $x \in \mathsf{X}$, the Bernoulli parameter should be such that:

$$p_t \leq \frac{\tau}{B\,|\mathcal{S}_t|(|\mathcal{S}_t| - 1)} \ . \tag{9}$$

Next, agent v updates its local parameters $\theta_{t,v} = (\alpha_{t,v}, \beta_{t,v})$ using a stochastic gradient descent. Unlike the algorithm of section III, the update should include the ℓ^1-penalization term in (8) and keep the nonnegativity constraints satisfied.

We thus propose the following update equations:

$$\alpha_{t+1,v} = [\alpha_{t,v} + \gamma_t\, h_v(X_{t+1,v}, \beta_{t,v})(Y_{t+1,v} - \hat{Y}^{(\mathcal{S}_t)}_{t+1,v}) - \lambda \text{sign}(\alpha_{t,v})\gamma_t]_+, \quad (10)$$

$$\beta_{t+1,v} = \beta_{t,v} + \gamma_t\, \alpha_{t,v} \nabla_{\beta_v} h_v(X_{t+1,v}, \beta_{t,v})(Y_{t+1,v} - \hat{Y}^{(\mathcal{S}_t)}_{t+1,v}), \quad (11)$$

with the notation $[u]_+ = \max(u, 0)$ and where ∇_{β_v} represents the gradient w.r.t. β_v. Finally, we need a criterion to decide whether agent v should declare itself as idle at step $t+1$ or should be kept active. Here, we propose to declare a node as idle at iteration $t+1$ if the current value $\alpha_{t+1,v}$ of the parameter α_v is zero for the M-th time, where M is an integer fixed in advance. Formally, a node v declares itself as idle if the sequence $(\alpha_{k,v})_{1 \le k \le t+1}$ contains at least M zeros.

VI. Numerical Results

The algorithms proposed have been tested on toy examples based on simulated data and on public datasets. Due to space limitations, only a few experiments are reported below: OLGA with experts selection is evaluated on a toy example since its usefulness can be simply illustrated and tested OLGA on real data.

Simulation data. We placed ourselves in the mixture of experts case, using randomly placed affine experts as weak classifiers namely $h_v(x_1, x_2; \theta_v) = \theta_v \text{sign}(\cos a_v x_1 + \sin a_v x_2 - \rho_v)$, where a_v and ρ_v are considered fixed for each agent. We then ran OLGA with experts selection and kept v for which $\theta_v \neq 0$. Notice below how the algorithm selects mostly affine separators relevant to the distribution (X, Y).

Real data. In this section, we compare the performances of GentleBoost (20) and OLGA on some benchmark datasets for binary classification: `banana` and `twonorm`. Detailed information about these datasets can be found in (21), see also (22) for a distributed boosting. We split each data sample into a *training* set and a *test* set using a $80\% - 20\%$ rule. For both GentleBoost and OLGA, the classifier is of the form $H(x) = \sum_{1 \le m \le M} \alpha_m h_m(x, \beta_m)$, based on linear combinations of *weak classifiers* $h(x, \beta)$, where β are the target parameters for the algorithm. For GentleBoost, the (α_m, β_m)'s are estimated using a *stagewise block* procedure. This means that $\alpha_1 h_1(\beta_1, \cdot)$ is added first, then $\alpha_2 h(\beta_2, \cdot)$, *etc.* and, for each $\alpha_m h_m(\beta_m, \cdot)$ to be added, a pass over the whole block of training data is required. For OLGA, the algorithm is *online* and *distributed*; meaning that each data is processed only *once* and then *forgotten*. In addition, each $\alpha_m h(\beta_m, \cdot)$ is computed *simultaneously* by separate agents forming a network. For GentleBoost, the form of the weak classifier is arbitrary, but a widespread choice is to use *stumps*, *i.e.* rules of the form $\mathbb{I}\{x^{(j)} \ge \beta\}$, where $x^{(j)}$ denote the x's j-th component. For OLGA we used "smooth stumps", of the form $F\left(\sigma(x^{(j)} - \beta)\right)$ where $F(x) = 1 - 2/(1 + \exp(-x))$. The smoothness is required by the gradient algorithm approach. In the case of OLGA, weak classifiers are in one-to-one correspondence with the agents: $V = \{1, \ldots, M\}$. Each agent v starts by uniformly randomly selecting an axis $j(v) \in \{1, \ldots, d\}$, independently from all other agents, and applies next the algorithm described in section V, using $\theta_{t,w} = (\alpha_{t,w}, \beta_{t,w}, \sigma_{t,w})$ and $h_w(x, \theta_{t,w}) = F\left(\sigma_{t,w}(x^{j(v)} - \beta_{t,w})\right)$. On both examples, one can see that OLGA does not outperform GentleBoost and has a more erratic error curve, due do its stochastic nature. However, it should be emphasized that: 1) both algorithms lead to comparable results and 2) OLGA is *online* and *distributed*, thus far less demanding in storage and power capability, which are crucial properties in a wide variety of applications.

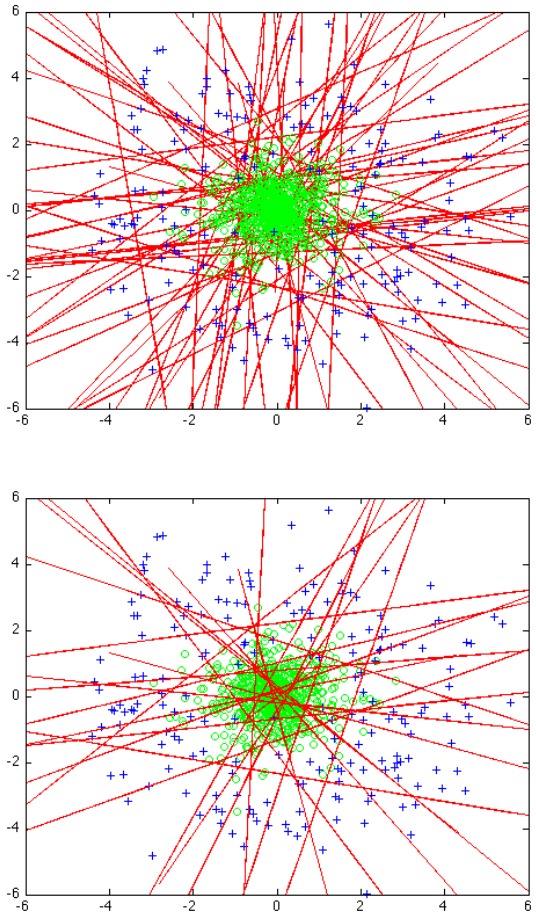

Figure 1. Left: Data a_v, ρ_v are represented by lines in red and sampling points (X, Y) by "+" in blue when $Y = -1$ and by "o" in green when $Y = 1$. Right: Only v having a non-zero weight $\theta_v \neq 0$ at the end of the iterations are represented.

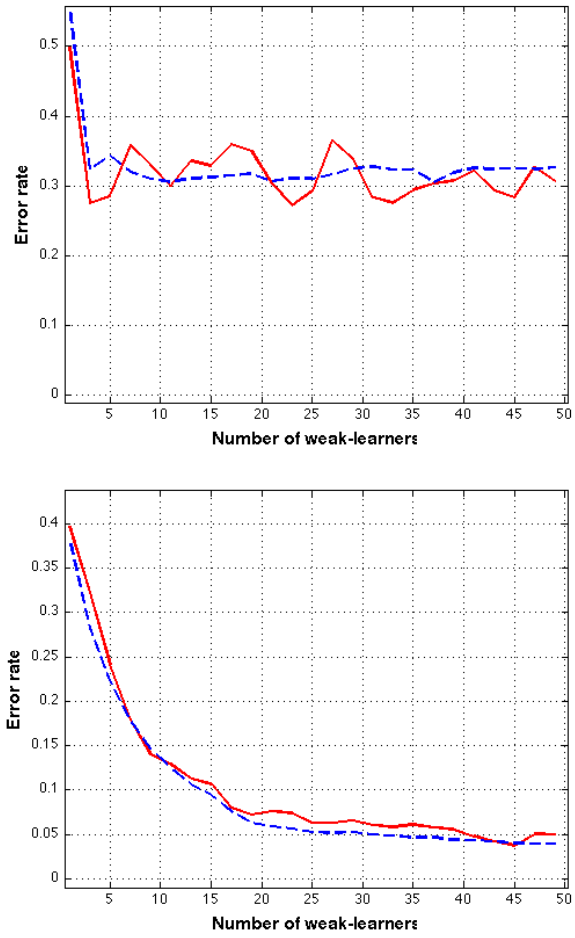

Figure 2. Comparison between GentleBoost and OLGA on datasets `banana` (left) and `twonorm` (right).

VII. Conclusion

In this paper, two variants of an online learning gossip algorithm (OLGA) for binary classification, very different in nature from "distributed consensus" approaches, are proposed and investigated. The main strength of OLGA lies in its ability to process data "on the fly" and then forget about it forever. Being distributed, datasets can be split and partially processed by several agents, while the network is able to benefit from the whole dataset. On real datasets tested in this paper, OLGA performs nearly as well as GentleBoost (20), a block centralized robust version of boosting that needs to store the whole dataset in order to process it. OLGA is well suited to underlying *complete* graphs, while subsampling edges to perform *sparsification* (23). Even this assumption seems realistic for IP networks, one might want to alleviate it and introduce hierarchies. Sophisticated versions of OLGA should be thus designed and analyzed in the near future.

Appendix - Technical details

Proof of Theorem 1 (sketch of)

By Assumption 2(a), $R_\varphi(\boldsymbol{\theta})$ is finite for any $\boldsymbol{\theta}$. Let us write its derivative. For any $\boldsymbol{\theta}$, any $v \in \mathcal{V}$ and any $(x,y) \in \mathsf{X} \times \{\pm 1\}$,

$$\begin{aligned}\nabla_v \varphi(-yH(x,\boldsymbol{\theta})) &= (H(x,\boldsymbol{\theta}) - y)\,\nabla_v h_v(x,\theta_v) \\ &= \left(\sum_{w \neq v} h_w(x,\theta_w) - y\right) \nabla_v h_v(x,\theta_v) \\ &+ \frac{1}{2}\nabla_v h_v^2(x,\theta_v)\ .\end{aligned}$$

In particular, for any fixed value of $\boldsymbol{\theta}$ and any $\theta_v' \in B(\theta_v,1) := \{\tilde{\theta} : \|\tilde{\theta} - \theta_v\| \leq 1\}$, we obtain that

$$\begin{aligned}\|\nabla_v \varphi(-yH(x,\boldsymbol{\theta}'))\| \leq \left(\sup_{\tilde{\theta} \in B(\theta_v,1)} \|\nabla_v h_v(x,\tilde{\theta})\|\right) \times \\ (1 + \sum_{w \neq v} |h_w(x,\theta_w))|) + \sup_{\tilde{\theta} \in B(\theta_v,1)} \|\nabla_v h_v^2(x,\tilde{\theta})\|,\end{aligned}$$

where we set $\boldsymbol{\theta}' = (\theta_1 \cdots \theta_{v-1}, \theta_v', \theta_{v+1} \cdots \theta_N)$. Thus, $\nabla_v \varphi(-yH(x,\boldsymbol{\theta}'))$ is bounded with a r.v. which does not depend on θ_v' and which can be proved to be integrable by straightforward application of Cauchy-Schwartz' inequality along with Assumptions 2(b,c). Using Lebesgue's dominated convergence theorem, R_φ is differentiable w.r.t. θ_v and its gradient coincides with

$$\nabla_v R_\varphi(\boldsymbol{\theta}) = \mathbb{E}\big[\,(H(X,\boldsymbol{\theta}) - Y)\,\nabla_v h_v(X,\theta_v)\big]$$

The next step is to prove that $\nabla_v R_\varphi$ is continuous, and thus that $R_\varphi(\boldsymbol{\theta})$ is continuously differentiable w.r.t. $\boldsymbol{\theta}$. This is a direct consequence of Assumption 2(d): the proof is left to the reader.

We are now in position to prove Theorem 1. Let us write our algorithm under the form $\theta_{t+1,v} = \theta_{t,v} + \gamma_t Z_{t+1,v}$ where we set:

$$Z_{t+1,v} = \nabla_v h_v(X_{t+1,v},\theta_{t,v})(Y_{t+1,v} - \hat{Y}_{t+1,v}^{(\mathcal{V})})\ . \qquad (12)$$

Let $(\mathcal{F}_t : t \geq 0)$ represent the natural filtration $\mathcal{F}_t = \sigma(\mathcal{F}_{t-1}, X_{t,1}, \cdots, X_{t,N}, Y_{t,1}, \cdots, Y_{t,N})$. From the previous statement, it follows that $\mathbb{E}(Z_{t+1,v}\,|\mathcal{F}_t) = \nabla_v R_\varphi(\boldsymbol{\theta}_t)$. Using Minkowski's inequality followed by Cauchy-Schwartz' inequality, we obtain:

$$\begin{aligned}\mathbb{E}\left[\|Z_{t+1,v}\|^2\,|\mathcal{F}_t\right]^{\frac{1}{2}} &\leq \mathbb{E}\left[\|\nabla_v h_v(X,\theta_v\|^2\right]^{\frac{1}{2}} + \\ &\sum_w \left(\mathbb{E}\left[\|h_w(X,\theta_w)\,\nabla_v h_v(X,\theta_v)\|^2\right]\right)^{\frac{1}{2}} \\ &\leq \mathbb{E}\left[\|\nabla_v h_v(X,\theta_v\|^4\right]^{\frac{1}{4}} \times \\ &\left(1 + \sum_w \mathbb{E}\left[h_w(X,\theta_w)^4\right]^{\frac{1}{4}}\right)\ .\end{aligned}$$

Therefore, Assumption 2(d) implies that for any compact set $\mathcal{K} \subset \boldsymbol{\Theta}$,

$$\sup_{\boldsymbol{\theta} \in \mathcal{K}} \mathbb{E}\left[\|Z_{t+1,v}\|^2 \,|\mathcal{F}_t\right] < \infty \ .$$

The proof is concluded by direct application of (18).

Proof of Theorem 2 (sketch of)

The proof relies on the fact R_φ is a Lyapunov function R_φ for the mean field of our algorithm, and that it is well-behaved as $\|\boldsymbol{\theta}\| \to \infty$. More precisely, we prove that ∇R_φ is Lipschitz-continuous and satisfies $\|\nabla R_\varphi\|^2 \leq \mathcal{C}\,(1 + R_\varphi)$ for some constant $\mathcal{C} > 0$. Using these conditions along with adequate estimates of the conditional moments of the noise sequence ξ_t, standard stochastic approximation results imply that sequence $\boldsymbol{\theta}_t$ remains in a compact set (see for instance (18)). Moreover, R_φ is convex under the assumptions of Theorem 2. Thus the stationary points coincide with the global minimizers.

Proof of Theorem 3 (sketch of)

Define $\xi_{t+1} = Z_{t+1} + \nabla R_\varphi(\boldsymbol{\theta}_t)$ where Z_{t+1} is the vector whose vth block-component coincides with $Z_{t+1,v}$ defined by (12). As already seen in the proof of Theorem 1, the sequence ξ_t is a martingale increment sequence adapted to the natural filtration, meaning that $\mathbb{E}[\xi_{t+1}\,|\mathcal{F}_t] = 0$ for any t. **Algorithm 1** writes $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \gamma_t \nabla R_\varphi(\boldsymbol{\theta}_t) + \gamma_t \xi_{t+1}$. Function $-\nabla R_\varphi$ is the so-called *mean field* of the algorithm, whereas ξ_t plays the role of a *noise* sequence.

Theorem 3 is a consequence of (19, Theorem 1). We only need to show that the assumptions of (19) are satisfied. To that end, we prove the following two technical lemmas. The first Lemma provides some conditions on the mean field of the algorithm. Due to space limitations, its proof is omitted.

Lemma 1. *Set $\boldsymbol{\theta}^\star \in \mathcal{L}$. Under Assumptions 2b−c and 3a−c, function R_φ is twice continuously differentiable on $\mathcal{N} := \prod_v \mathcal{N}_v$ and satisfies:*

$$\nabla^2 R_\varphi(\boldsymbol{\theta}) = \mathbb{E}\left[\nabla H(X,\boldsymbol{\theta}) \nabla^T H(X,\boldsymbol{\theta})\right] + \mathbb{E}\left[(H(X,\boldsymbol{\theta}) - Y) \nabla^2 H(X,\boldsymbol{\theta})\right] \ .$$

Moreover, $\nabla R_\varphi(\boldsymbol{\theta}) = Q^\star(\boldsymbol{\theta} - \boldsymbol{\theta}^\star) + \mathcal{O}(\|\boldsymbol{\theta} - \boldsymbol{\theta}^\star\|^2)$.

The second Lemma yields the required conditions on the probabilistic behavior of noise sequence.

Lemma 2. *Set $\boldsymbol{\theta}^\star \in \mathcal{L}$. Let Assumptions 2(a-d) and 3(e-f) hold true. Then, we have:*

$$\sup_{t \geq 0} \mathbb{E}(\|\xi_{t+1}\|^{b/2} \,|\mathcal{F}_t) \mathbb{I}_{\boldsymbol{\theta}_t \in \mathcal{N}} < \infty.$$

Moreover, almost surely on the event $\{\boldsymbol{\theta}_t \to \boldsymbol{\theta}^\star\}$, $\mathbb{E}(\xi_{t+1}\xi_{t+1}^T \,|\mathcal{F}_t) \to \Gamma^\star$ as $t \to \infty$.

Theorem 3 directly follows from Lemmas 1 and 2 by straightforward application of (19).

PROOF OF LEMMA 2.

Since ∇R_φ is continuous, it is bounded in a neighborhood of $\boldsymbol{\theta}^\star$. Therefore, it is quite immediate to see that the statement $\sup_{t \geq 0} \mathbb{E}[\|\xi_{t+1}\|^{b/2} \,|\mathcal{F}_t] \mathbb{I}\{\boldsymbol{\theta}_t \in \mathcal{N}\} < \infty$ is in fact equivalent to $\sup_{t \geq 0} \mathbb{E}[\|Z_{t+1,v}\|^{b/2} \,|\mathcal{F}_t] \mathbb{I}\{\boldsymbol{\theta}_t \in \mathcal{N}\} < \infty$ for any $v \in \mathcal{V}$. Recalling the definition (12) of $Z_{t+1,v}$, we obtain using Cauchy-Schwartz' inequality:

$$\begin{aligned}\mathbb{E}[\|Z_{t+1,v}\|^{b/2} \,|\mathcal{F}_t] &\leq \mathbb{E}\left[\|\nabla_v h_v(X_{t+1,v}, \theta_{t,v})\|^b \,|\mathcal{F}_t\right]^{\frac{1}{2}} \\ &\quad \times \mathbb{E}[|\hat{Y}_{t+1,v}^{(\mathcal{V})} - Y_{t+1,v}|^b \,|\mathcal{F}_t]^{\frac{1}{2}} \\ &\leq \mathcal{C}\ \mathbb{E}\left[\|\nabla_v h_v(X, \theta_{t,v})\|^b\right]^{\frac{1}{2}} (1 + \mathbb{E}[|\hat{Y}_{t+1,v}^{(\mathcal{V})}|^b \,|\mathcal{F}_t])^{\frac{1}{2}}\end{aligned}$$

for some constant $\mathcal{C} > 0$ which depends on b. Assumption 3(e) ensures that the first factor in the righthand side of the above inequality is bounded uniformly in t when multiplied by the indicator of event $\{\boldsymbol{\theta}_t \in \mathcal{N}\}$. The remaining task is thus to estimate $\mathbb{E}[|\hat{Y}_{t+1,v}^{(\mathcal{V})}|^b \,|\mathcal{F}_t]$. Recalling (3), one can prove after some algebra that:

$$\mathbb{E}[|\hat{Y}_{t+1,v}^{(\mathcal{V})}|^b \,|\mathcal{F}_t]^{1/b} \leq \mathcal{C}' \times \sum_{w \in \mathcal{V}} \left(\sup_{\theta_w \in \mathcal{N}_w} \mathbb{E}\left[\|h_w(X, \theta_w)\|^b\right]\right)^{1/b} .$$

for some constant $\mathcal{C}' > 0$. Using again Assumption 3(e), we obtain that the righthand side is bounded. Putting all pieces together, this proves $\sup_{t \geq 0} \mathbb{E}[\|\xi_{t+1}\|^{b/2} \,|\mathcal{F}_t] \mathbb{I}\{\boldsymbol{\theta}_t \in \mathcal{N}\} < \infty$. Consider the second statement of Lemma 2. For any $v \neq w$, $Z_{t+1,v}$ and $Z_{t+1,w}$ are independent conditionally to $\mathcal{F}_t$. Thus, it is sufficient to study the conditional covariance of $\xi_{t+1,v}$ for a given v. The latter covariance matrix coincides with $U_v(\boldsymbol{\theta}_t) - \nabla_v R_\varphi(\boldsymbol{\theta}_t) \nabla_v R_\varphi(\boldsymbol{\theta}_t)^T$ where $U_v(\boldsymbol{\theta}_t) = \mathbb{E}[Z_{t+1,v} Z_{t+1,v}^T \,|\mathcal{F}_t]$. Upon noting that $\nabla_v R_\varphi$ is continuous and $\nabla_v R_\varphi(\boldsymbol{\theta}^\star) = 0$, it is therefore sufficient to show that $\boldsymbol{\theta} \mapsto U_v(\boldsymbol{\theta})$ is continuous and that $U_v(\boldsymbol{\theta}^\star) = \Gamma_v(\boldsymbol{\theta}^\star)$, in order to complete the proof of Lemma 2. Note that $U_v(\boldsymbol{\theta}_t)$ coincides with the conditional expectation of $(\hat{Y}_{t+1,v}^{(\mathcal{V})} - Y_{t+1,v})^2 \nabla_v h_v(X_{t+1,v}, \theta_{t,v}) \nabla_v h_v(X_{t+1,v}, \theta_{t,v})^T$, given $\mathcal{F}_t$. After some tedious but straightforward derivations, one is able to show that $U_v(\boldsymbol{\theta}) = \Gamma_v(\boldsymbol{\theta})$. By Assumption 3(f), the proof of Lemma 2 is complete.

Proof of Corollary 1

We use a second-order Taylor-Lagrange expansion of R_φ at $\boldsymbol{\theta}^\star$. As $\nabla_v R_\varphi(\boldsymbol{\theta}^\star) = 0$, $R_\varphi(\boldsymbol{\theta}_t) - R_\varphi(\boldsymbol{\theta}^\star)$ is equal to $\frac{1}{2}(\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star)^T \nabla^2 R_\varphi(\boldsymbol{\theta}^\star)(\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star)$ up to a negligible term. Upon noting that $\nabla^2 R_\varphi(\boldsymbol{\theta}^\star) = Q^\star$, the result follows from Theorem 3.

Pseudo-code - Penalized OLGA

The method proposed in Section V is summarized by **Algorithm 2** below.

Algorithm 2 OLGA

Initialize: Set $\mathcal{S} = \mathcal{V}$. For each node $v \in \mathcal{V}$, set initial values $\beta_{0,v}$ and $\alpha_{0,v} = 1$. Set $\texttt{counter}_v = 0$.

Update: At each time $t = 1, 2, \cdots$ **do**

For each $v \in \mathcal{S}$ **do**

Neighbors selection: Set $p_t = \tau / B|\mathcal{S}|(|\mathcal{S}| - 1)$
Draw independent Bernoulli r.v.'s

$$\delta^w_{t+1,v} \sim \mathcal{B}(p_t)$$

for any $w \in \mathcal{S}$, $w \neq v$

Gossip step: Transmit $X_{t+1,v}$ to all nodes w such that $\delta^w_{t+1,v} = 1$, and obtain $h_w(X_{t+1,v}, \theta_{t,w})$ in return

Local descent: Update parameters $\alpha_{t+1,v}, \beta_{t+1,v}$ using (10)-(11)

if $\alpha_{t+1,v} = 0$ **then** $\texttt{counter}_v \leftarrow \texttt{counter}_v + 1$
if $\texttt{counter}_v = M$ **then** $\mathcal{S} \leftarrow \mathcal{S} \setminus \{v\}$

REFERENCES

[1] J. Tsitsiklis, "Problems in Decentralized Decision Making and Computation," Ph.D. dissertation, Massachusetts Institute of Technology, 1984.

[2] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, 1997.

[3] L. Huang, X. Nguyen, M. Garofalakis, M. Jordan, A. Joseph, and N. Taft, "In-network pca and anomaly detection," *Advances in Neural Information Processing Systems*, vol. 19, p. 617, 2007.

[4] S. Korada, A. Montanari, and S. Oh, "Gossip pca," *ACM SIGMETRICS Performance Evaluation Review*, vol. 39, no. 1, pp. 169–180, 2011.

[5] S. Lee and A. Nedic, "Drsvm: Distributed random projection algorithms for svms," submitted.

[6] A. Navia-Vazquez, D. Gutierrez-Gonzalez, E. Parrado-Hernandez, and J. Navarro-Abellan, "Distributed support vector machines," *Neural Networks, IEEE Transactions on*, vol. 17, no. 4, pp. 1091–1097, 2006.

[7] P. Forero, A. Cano, and G. Giannakis, "Consensus-based distributed support vector machines," *The Journal of Machine Learning Research*, vol. 11, pp. 1663–1707, 2010.

[8] G. Mateos, J. Bazerque, and G. Giannakis, "Distributed sparse linear regression," *Signal Processing, IEEE Transactions on*, vol. 58, no. 10, pp. 5262–5276, 2010.

[9] Y. Low, J. Gonzalez, A. Kyrola, D. Bickson, C. Guestrin, and J. M. Hellerstein, "Distributed GraphLab: A Framework for Machine Learning and Data Mining in the Cloud," *PVLDB*, 2012.

[10] O. Dekel, R. Gilad-Bachrach, O. Shamir, and L. Xiao, "Optimal distributed online prediction," in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 2011, pp. 713–720.

[11] J. Duchi, J. Agarwal, and M. Wainwright, "Dual Averaging for Distributed Optimization: Convergence Analysis and Network Scaling," *Automatic control, IEEE Transactions on*, vol. 99, no. 10, pp. 1–40, 2010.

[12] ——, "Distributed Dual Averaging in Networks," in *Advances in Neural Information Systems*, 2010.

[13] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized Gossip Algorithms," *IEEE Transactions on Inform. Theory*, vol. 52, no. 6, pp. 2508–2530, 2006.

[14] F. Bnzit, "Distributed Average Consensus for Wireless Sensor Networks," Ph.D. dissertation, École Polytechnique Fdrale de Lausanne, 2009.

[15] P. Bartlett, M. Jordan, and J. McAuliffe, "Convexity, classification, and risk bounds," *Journal of the American Statistical Association*, vol. 101, no. 473, pp. 138–156, 2006.

[16] M. Benaim, J. Hofbauer, and S. Sorin, "Stochastic Approximations and Differential Inclusions," *SIAM Journal on Control and Optimization*, vol. 44, no. 1, pp. 328–348, 2005.

[17] H. Kushner and D. Clark, *Stochastic Approximation Methods for Constrained and Unconstrained Systems*. Springer-Verlag, 1978.

[18] B. Delyon, "Stochastic Approximation with Decreasing Gain: Convergence and Asymptotic Theory," *Unpublished Lecture Notes, http://perso.univ-rennes1.fr/bernard.delyon/as_cours.ps*, 2000.

[19] M. Pelletier, "Weak convergence rates for stochastic approximation with application to multiple targets and simulated annealing," *Annals of Applied Probability*, vol. 8, no. 1, pp. 10–44, 1998.

[20] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors)," *The annals of statistics*, vol. 28, no. 2, pp. 337–407, 2000.

[21] G. Rätsch, T. Onoda, and K. Müller, "Soft margins for adaboost," *Machine learning*, vol. 42, no. 3, pp. 287–320, 2001.

[22] J. Ye, J. Chow, J. Chen, and Z. Zheng, "Stochastic gradient boosted distributed decision trees," in *Proceeding of the 18th ACM conference on Information and knowledge management*, 2009, pp. 2061–2064.

[23] D. Achlioptas and F. McSherry, "Fast computation of low rank matrix approximations," in *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, 2001, pp. 611–618.

[24] H. Robbins and S. Monro, "A Stochastic Approximation Method," *Ann. of Mathem. Statist.*, vol. 22, no. 3, pp. 400–407, 1951.

[25] H. Kushner and G. Yin, *Stochastic Approximation and Recursive Algorithms and Applications*. Springer, 2003.