Interaction-Based Distributed Learning in Cyber-Physical and Social Networks

Francesco Sasso*, Angelo Coluccia*, *Senior Member, IEEE* and Giuseppe Notarstefano†, *Member, IEEE*

***Abstract*—In this paper we consider a network scenario in which agents can evaluate each other according to a score graph that models some physical or social interaction. The goal is to design a distributed protocol, run by the agents, allowing them to learn their unknown state among a finite set of possible values. We propose a Bayesian framework in which scores and states are associated to probabilistic events with unknown parameters and hyperparameters respectively. We prove that each agent can learn its state by combining a local Bayesian classifier with a (centralized) Maximum Likelihood (ML) estimator of the parameter-hyperparameter. To overcome the intractability of the ML problem, we provide two relaxed probabilistic models that lead to distributed estimation schemes with affordable complexity. In order to highlight the appropriateness of the proposed relaxations, we demonstrate the distributed estimators on a machine-to-machine testing set-up for anomaly detection and on a social interaction set-up for user profiling.**

I. Introduction

A typical feature of cyber-physical and social networks is the mutual interaction among subsystems. In social networks individuals continuously interact by sharing contents and expressing opinions or ratings on different topics. Similarly, in industrial (control) networks, as power-networks, smart grids or automated factories, devices have the possibility to test each other to detect faults or malware attacks. In this paper we model a general network scenario in which nodes can give/receive a score to/from other "neighboring" nodes with the goal of deciding their own (or their neighbors') state. The state may indicate the level of (mis)trust or faultiness, the belonging to a class/community, or an influence level. Centralized solutions for node classification are computationally expensive in large-scale networks and do not preserve privacy. Thus, distributed solutions need to be investigated.

Literature review: In the past few years, a great interest has been devoted to distributed estimation schemes in which nodes aim at agreeing on a common parameter, e.g., by means of Maximum Likelihood (ML) approaches, [2]–[4]. In [5]–[7] a more general Bayesian framework is considered, in which nodes estimate local parameters, rather than reaching consensus on a common one. The estimation of the local parameters is performed by resorting to an Empirical Bayes approach in which the parameters of the prior distribution, called hyperparameters, are estimated through a distributed algorithm. The estimated hyperparameters are then combined with local measurements to obtain the Minimum Mean Square Error (MMSE) estimator of the local parameters. Consensus-based algorithms have been proposed in [4], [8], [9] for the simultaneous distributed estimation and classification of network nodes. A different classification set-up is considered in [10], where a group of individuals needs to decide on two alternative hypotheses; the global decision is, however, taken by a fusion center. The recent literature on distributed social learning, [11], focuses on non-Bayesian schemes in which each agent processes its own and its neighbors' beliefs [12], [13], see also [14] for a survey. Later work investigates the effect of network size/structure [15], [16] and faulty nodes [17] on the learning rules. Schemes for time-varying topologies are proposed in [18]. Differently from our set-up, all these references consider a scenario in which agents aim at learning a *common* unobservable state. A different batch of references proposes dynamic laws modeling interpersonal influences in groups of individuals, and investigates the asymptotic behavior of opinions [19]–[21]. The tutorial [22] reviews opinion formation in social networks and other applications by means of randomized distributed algorithms. Finally, the problem of self-rating in a social environment is addressed in [23] by introducing suitable distributed dynamics.

Statement of contributions: The contributions of this paper are as follows. First, we set up a learning problem in a network context in which nodes want to learn their own state based on observations coming from the interaction with other nodes. This general scenario captures a wide variety of social and machine-to-machine contexts, where information comes from interactions rather than from local measurements of the surrounding environment. For this set-up we devise a Bayesian probabilistic framework in which both the parameters of the observation model and the hyperparameters of the prior distribution are allowed to be unknown. In this sense, this framework can be seen as an Empirical Bayes approach with additional unknown parameters. Second, in order to solve this interaction-based problem, we propose a learning approach combining a local Bayesian classifier with a joint parameter-hyperparameter Maximum Likelihood estimation approach. For the local Bayesian classifier, we derive a closed form expression depending only on aggregated evaluations from the neighbors. This expression can be used to obtain both the Maximum A Posteriori (MAP) decision as well as a ranking of the alternatives with associated (probabilistic) trust. Since

*Francesco Sasso and Angelo Coluccia are with the Department of Engineering, Università del Salento, via Monteroni, 73100, Lecce, Italy, {name.lastname}@unisalento.it.

†Giuseppe Notarstefano is with the Department of Electrical, Electronic, and Information Engineering, University of Bologna, Viale del Risorgimento, 2, 40136, Bologna, Italy, giuseppe.notarstefano@unibo.it.

This result is part of a project that has received funding from the European Research Council (ERC) under the European Unions Horizon 2020 research and innovation programme (grant agreement No 638992 - OPT4SMART).

A preliminary version of this work appeared in ECC'18, [1].

the complexity of the ML estimator grows exponentially with the network size, we identify two reasonable relaxations that lead to modified likelihood functions exhibiting a separable structure. In particular, we propose a node-based relaxation, for which available distributed optimization algorithms can be used, and a full relaxation for which we design an ad-hoc distributed algorithm combining a local descent step with a diffusion step. We analyze the performance of the proposed distributed classifiers through Monte Carlo simulations on two interesting scenarios, namely on anomaly detection in cyber-physical networks and user profiling in social networks.

Organization: In Section II we set-up the Bayesian framework and introduce the two application scenarios. In Section III we derive the proposed distributed classification algorithms. Finally, in Section IV we report the numerical results.

II. Bayesian framework for interaction-based learning

In this section, we set up the interaction-based learning problem and introduce a general Bayesian probabilistic model.

A. Interaction network model

We consider a *network of agents* able to perform evaluations of other agents. The result of each evaluation is a score given by the evaluating agent to the evaluated one. Such an interaction is described by a *score graph*. Formally, let $\{1,\dots,N\}$ be the set of agent identifiers and $G_S = (\{1,\dots,N\}, E_S)$ a digraph such that $(i,j) \in E_S$ if agent i evaluates agent j. We denote by n the total number of edges in the graph, and assume that each node has at least one incoming edge in the score graph, that is there is at least one agent evaluating it.

Let $\mathcal{C}$ and $\mathcal{R}$ be the set of possible state and score values, respectively. Being finite sets, we can assume $\mathcal{C} = \{c_1,\dots,c_C\}$ and $\mathcal{R} = \{r_1,\dots,r_R\}$, where C and R are the cardinality of the two sets, respectively. Consistently, in the network we consider the following quantities:

- $x_i \in \mathcal{C}$, unobservable *state* (or community) of agent i;
- $y_{ij} \in \mathcal{R}$, *score* (or evaluation result) of the evaluation performed by agent i on agent j.

An example of score graph with associated state and score values is shown in Fig. 1.

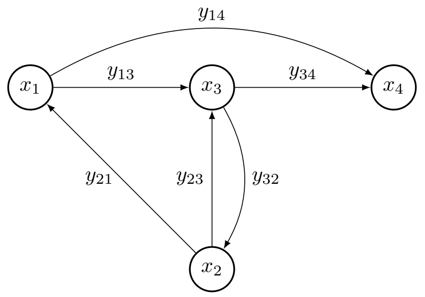

Fig. 1. Example of a score graph G_S.

Besides the evaluation capability, the agents have also *communication* and *computation* functionalities. That is, agents communicate according to a time-dependent directed *communication graph* $t \mapsto G_{\texttt{cmm}}(t) = (\{1,\dots,N\}, E_{\texttt{cmm}}(t))$, where the edge set $E_{\texttt{cmm}}(t)$ describes the communication among agents: $(i,j) \in E_{\texttt{cmm}}(t)$ if agent i communicates to j at time $t \in \mathbb{Z}_{\geq 0}$. We introduce the notation $N^I_{\texttt{cmm},i}(t)$ and $N^O_{\texttt{cmm},i}(t)$ for the in- and out-neighborhoods of node i at time t in the communication graph. We will require these neighborhoods to include the node i itself; formally, we have

$$N^I_{\texttt{cmm},i}(t) = \{j : (j,i) \in E_{\texttt{cmm}}(t)\} \cup \{i\},$$
$$N^O_{\texttt{cmm},i}(t) = \{j : (i,j) \in E_{\texttt{cmm}}(t)\} \cup \{i\}$$

We assume the following on the communication graph:

Assumption II.1. *There exists an integer $Q \geq 1$ such that the graph $\bigcup_{\tau=tQ}^{(t+1)Q-1} G_{\texttt{cmm}}(\tau)$ is strongly connected $\forall\, t \geq 0$.*

We point out that the (time-dependent) communication graph, modeling the distributed computation, is not necessarily related to the (fixed) score graph. We just assume that, when the distributed algorithm starts, each node i knows the scores received by in-neighbors in the score graph. This could be obtained by assuming that if the distributed algorithm starts at some time t_0, then for some $\bar{t} > 0$, $G_S \subseteq \bigcup_{\tau=t_0-\bar{t}}^{t_0} G_{\texttt{cmm}}(\tau)$.

B. Bayesian probabilistic model

We consider the score $y_{ij}, (i,j) \in E_S$, as the (observed) realization of a random variable denoted by Y_{ij}; likewise, each state value $x_i, i \in \{1,\dots,N\}$, is the (unobserved) realization of a random variable X_i. In order to highlight the conditional dependencies among the random variables involved in the score graph, we resort to the tool of graphical models, in particular Bayesian networks [24]. Specifically, we introduce the *Score Bayesian Network* with $N+n$ nodes X_i, $i = 1,\dots,N$, and Y_{ij}, $(i,j) \in E_S$ and $2n$ (conditional dependency) arrows defined as follows. For each $(i,j) \in E_S$, we have $X_i \to Y_{ij} \leftarrow X_j$ indicating that Y_{ij} conditionally depends on X_i and X_j. In Fig. 2 we represent the Score Bayesian Network related to the score graph in Fig. 1.

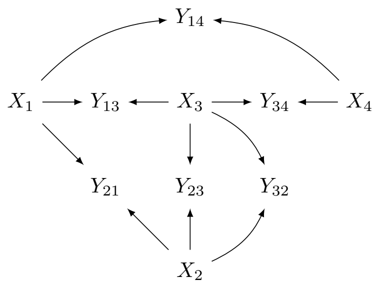

Fig. 2. The score Bayesian network related to the score graph in Fig. 1.

Denoting by $\boldsymbol{Y}_{E_S}$ the vector of all random variables $Y_{ij}, (i,j) \in E_S$, the joint distribution factorizes as

$$\mathbb{P}(\boldsymbol{Y}_{E_S}, X_1,\dots,X_N) = \Big(\prod_{(i,j)\in E_S} \mathbb{P}(Y_{ij}|X_i,X_j)\Big)\Big(\prod_{i=1}^{N} \mathbb{P}(X_i)\Big).$$

We assume Y_{ij}, $(i,j) \in E_S$ are ruled by a conditional probability distribution $\mathbb{P}(Y_{ij}|X_i,X_j;\boldsymbol{\theta})$, depending on a *parameter* vector $\boldsymbol{\theta}$ whose components take values in a given set Θ. For notational purposes, we define the tensor

$$p_{h|\ell,m}(\boldsymbol{\theta}) := \mathbb{P}(Y_{ij} = r_h | X_i = c_\ell, X_j = c_m; \boldsymbol{\theta}), \qquad (1)$$

where $r_h \in \mathcal{R}$ and $c_\ell, c_m \in \mathcal{C}$. From the definition of probability distribution, we have the constraint $\boldsymbol{\theta} \in \mathcal{S}_\Theta$ with

$$\mathcal{S}_\Theta := \Big\{ \boldsymbol{\theta} \in \Theta : p_{h|\ell,m}(\boldsymbol{\theta}) \in [0,1], \sum_{h=1}^{R} p_{h|\ell,m}(\boldsymbol{\theta}) = 1 \Big\}.$$

To clarify the notation, an example realization of $p_{h|\ell,m}(\boldsymbol{\theta})$ for a given $\boldsymbol{\theta} \in \mathcal{S}_\Theta$, is depicted in Fig. 3.

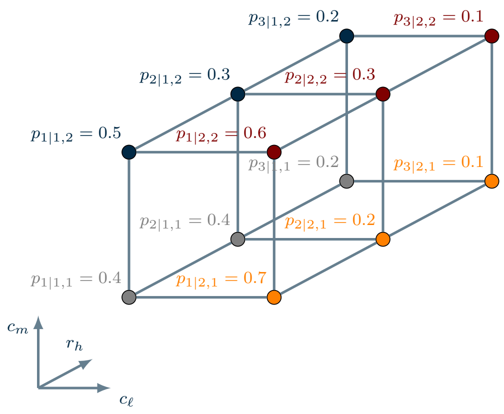

Fig. 3. Example of a tensor $\{p_{h|\ell,m}(\boldsymbol{\theta})\}_{\ell,m=1,2|h=1,2,3}$ for fixed $\boldsymbol{\theta} \in \mathcal{S}_{\boldsymbol{\theta}}$.

We model X_i, $i = 1, \ldots, N$, as identically distributed random variables ruled by a probability distribution $\mathbb{P}(X_i; \boldsymbol{\gamma})$, depending on a *hyperparameter* vector $\boldsymbol{\gamma}$ whose components take values in a given set Γ. We introduce the notation

$$p_\ell(\boldsymbol{\gamma}) := \mathbb{P}(X_i = c_\ell; \boldsymbol{\gamma}) \qquad (2)$$

and, analogously to $\boldsymbol{\theta}$, we have the constraint $\boldsymbol{\gamma} \in \mathcal{S}_\Gamma$ with

$$\mathcal{S}_\Gamma := \Big\{ \boldsymbol{\gamma} \in \Gamma : p_\ell(\boldsymbol{\gamma}) \in [0,1], \sum_{\ell=1}^{C} p_\ell(\boldsymbol{\gamma}) = 1 \Big\}.$$

We suppose each node knows $p_{h|\ell,m}$, p_ℓ and the scores received from its in-neighbors and given to its out-neighbors in G_S. Also, for the sake of analysis, we assume $p_{h|\ell,m}$ and p_ℓ to be continuous functions.

Notice that the least structured case for the model above is given by the *categorical model* in which the vector of parameters and the vector of hyperparameters are given by the corresponding probability masses. That is, $\boldsymbol{\theta}$ and $\boldsymbol{\gamma}$ have respectively $R+2C$ and C components. We point out that the categorical model, being so unstructured, is the most flexible one. Clearly, this flexibility is paid by a much higher number of parameters, which quickly degenerates in over-fitting. Therefore, in practical applications one usually exploits domain-specific knowledge to identify a suitable parametrization in terms of the most relevant parameters and hyperparameters. Some examples are discussed in the next subsection, while the problem of jointly estimating the *parameter-hyperparameter* $(\boldsymbol{\theta}, \boldsymbol{\gamma})$ will be addressed in the next section as a building block of the (distributed) learning scheme.

C. Examples of application scenarios

1) Binary-state learning for anomaly detection: We consider a network in which each node i tests neighboring nodes j with a binary outcome indicating if the tested node is deemed faulty (i.e., its state is $x_j = 1$) or not (i.e., $x_j = 0$). Since each node performing the evaluation can be itself faulty, its outcome may be not reliable; also, no node knows whether it itself is faulty or not. We consider a probabilistic extension of the well-known Preparata's model [25]. Specifically, we assume that the evaluation outcome is determined as follows: if node i is working properly, then it will return the true status of the evaluated node j (i.e., $y_{ij} = 1$ if node j is faulty and $y_{ij} = 0$ if it is working properly); conversely, if node i is faulty, the outcome is uniformly random. Formally:

$$\begin{aligned} p_{h|\ell,m} &= (1 - c_\ell)\Big[(1 - c_m)(1 - r_h) + c_m r_h\Big] + \frac{1}{2}c_\ell, \\ p_\ell(\gamma) &= \gamma^{c_\ell}(1-\gamma)^{1-c_\ell}, \qquad \gamma \in [0,1], \end{aligned} \qquad (3)$$

with $R = 2$ $(r_1 = 0, r_2 = 1)$ and $C = 2$ $(c_1 = 0, c_2 = 1)$. In this first scenario, we assume for simplicity that the distribution of evaluation results is known, so the only unknown is the hyperparameter γ, i.e., the *a priori* probability that a node is faulty. We refer to this model as *anomaly detection model*.

2) Social ranking: Another relevant scenario is user profiling in social networks, wherein people tend to aggregate (tacitly or explicitly) into groups based on some affinity.

For example, consider an online forum on a dedicated subject, wherein each member can express her/his preferences by assigning to posts of other members/colleagues a score from 1 to R indicating an increasing level of appreciation for that post. In order to model the distribution of scores, we consider distance-based ranking distributions, [26], [27], in which (ranking) probabilities decrease as the distance from a reference (ranking) probability increases. To fit our needs, we propose for the distribution of scores the following slight variant of the so-called Mallow's ϕ-model (see [28]):

$$p_{h|\ell,m}(\theta) = \frac{1}{\psi_{\ell,m}(\theta)} e^{-\left(\frac{(r_R - r_h)/r_R - d(c_\ell, c_m)/c_C}{\theta}\right)^2}, \qquad (4)$$

where $r_h = h$ $(h = 1, \ldots, R)$, $c_\ell = \ell$ $(\ell = 1, \ldots, C)$, $\theta \in \mathbb{R}_{>0}$ is a dispersion parameter, $\psi_{\ell,m}(\theta)$ is a normalizing constant, and d is a semi-distance, i.e., $d \geq 0$ and $d(c_\ell, c_m) = 0$ iff $c_\ell = c_m$. Informally, the "farther" a given community c_ℓ is from another community c_m, the higher will be the distance $d(c_\ell, c_m)$, thus the lower the score.

In many cases the resulting subgroups reflect some hierarchy in the population. Basic examples could be forums or working teams. Thus, we consider a scenario in which each person belongs to a community reflecting some degree of expertise about a given topic or field. In particular, we have C ordered communities, with the ℓth community given by $c_\ell = \ell$. That is, for example, a person in the community c_1 is an *newbie*, while a person in c_C is a *master*. Since climbing in the hierarchy is typically the result of several promotion events, a natural probabilistic model for the communities is a binomial distribution $\mathcal{B}(C-1,\gamma)$, where $\gamma \in [0,1]$ represents the probability of being promoted, i.e.,

$$p_\ell(\gamma) = \binom{C-1}{c_\ell - 1}\gamma^{c_\ell - 1}(1-\gamma)^{C-1-(c_\ell-1)}.$$

We will refer to this second set-up as *social ranking model*.

III. Interaction-based distributed learning

In this section we describe the proposed distributed learning scheme. Without loss of generality, we focus on a set-up in which a node wants to self-classify. The same scheme also applies to a scenario in which a node wants to classify its neighbors, provided it knows their given and received scores. Notice that, in many actual contexts, as, e.g., social network platforms, this information is readily available.

The section is structured as follows. First, we derive a local Bayesian classifier provided that an estimation of parameter-hyperparameter $(\boldsymbol{\theta},\boldsymbol{\gamma})$ is available. Then, based on a combination of plain ML and Empirical Bayes estimation approaches, we derive a joint parameter-hyperparameter estimator. Finally, we propose two suitable relaxations of the Score Bayesian Network which lead to distributed estimators, based on proper distributed optimization algorithms.

A. Bayesian classifiers (given parameter-hyperparameter)

Each node can self-classify (i.e., learn its own state) if an estimate $(\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\gamma}})$ of parameter-hyperparameter $(\boldsymbol{\theta},\boldsymbol{\gamma})$ is available. Before discussing in details how this estimate can be obtained in a distributed way, we develop a *decentralized* MAP self-classifier that uses only single-hop information, i.e., the scores it gives to and receives from neighbors.

Formally, let $\boldsymbol{y}_{N_i}$ be the vector of (observed) scores that agent i obtains by in-neighbors and provides to out-neighbors, i.e., the stack vector of y_{ji} with $(j,i) \in E_S$ and y_{ij} with $(i,j) \in E_S$. Consistently, let $\boldsymbol{Y}_{N_i}$ be the corresponding random vector; then for each agent $i = 1,\dots,N$, we define

$$u_i(c_\ell) := \mathbb{P}(X_i = c_\ell | \boldsymbol{Y}_{N_i} = \boldsymbol{y}_{N_i}; \hat{\boldsymbol{\gamma}}, \hat{\boldsymbol{\theta}}), \quad \ell = 1,\dots,C.$$

The *soft classifier* of i is the probability vector $\boldsymbol{u}_i := (u_i(c_1),\dots,u_i(c_C))$ (with nonnegative components that sum to 1). Fig. 4 reports a pie-chart representation of an example.

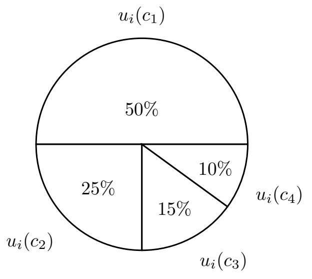

Fig. 4. Example of outcome of the soft classifier of an agent i, for $C = 4$: $\boldsymbol{u}_i = (0.5, 0.25, 0.15, 0.1)$.

From the soft classifier we can define the classical *Maximum A-Posteriori probability (MAP) classifier* as the argument corresponding to the maximum component of u_i, i.e.,

$$\hat{x}_i := \operatorname*{argmax}_{c_\ell \in \mathcal{C}} u_i(c_\ell).$$

The main result here is to show how to efficiently compute the soft and MAP classifiers. First, we define

$$\begin{aligned} N_i^{\leftrightarrow} &:= \{j : (j,i) \in E_S, (i,j) \in E_S\},\\ N_i^{\leftarrow} &:= \{j : (j,i) \in E_S, (i,j) \notin E_S\},\\ N_i^{\rightarrow} &:= \{j : (i,j) \in E_S, (j,i) \notin E_S\}, \end{aligned}$$

and for each $h,k = 1,\dots,R$ we introduce the quantities:

$$\begin{aligned} n_i^{\leftrightarrow}(h,k) &:= |\{j \in N_i^{\leftrightarrow} : y_{ij} = r_h, y_{ji} = r_k\}|,\\ n_i^{\leftarrow}(h) &:= |\{j \in N_i^{\leftarrow} : y_{ji} = r_h\}|,\\ n_i^{\rightarrow}(h) &:= |\{j \in N_i^{\rightarrow} : y_{ij} = r_h\}|. \end{aligned}$$

Theorem III.1. *Let $i \in \{1,\dots,N\}$ be an agent of the score graph. Then, the components of the vector u_i are given by*

$$u_i(c_\ell) = \frac{v_i(c_\ell)}{Z_i}$$

where $Z_i = \sum_{\ell=1}^{C} v_i(c_\ell)$ is a normalizing constant, and $v_i(c_\ell) = p_\ell(\hat{\boldsymbol{\gamma}})\pi_i^{\leftrightarrow}(c_\ell)\pi_i^{\leftarrow}(c_\ell)\pi_i^{\rightarrow}(c_\ell)$ with

$$\pi_i^{\leftrightarrow}(c_\ell) = \prod_{h,k=1}^{R}\Big(\sum_{m=1}^{C} p_{k|m,\ell}(\hat{\boldsymbol{\theta}})p_{h|\ell,m}(\hat{\boldsymbol{\theta}})p_m(\hat{\boldsymbol{\gamma}})\Big)^{n_i^{\leftrightarrow}(h,k)},$$

$$\pi_i^{\leftarrow}(c_\ell) = \prod_{h=1}^{R}\Big(\sum_{m=1}^{C} p_{h|m,\ell}(\hat{\boldsymbol{\theta}})p_m(\hat{\boldsymbol{\gamma}})\Big)^{n_i^{\leftarrow}(h)},$$

$$\pi_i^{\rightarrow}(c_\ell) = \prod_{h=1}^{R}\Big(\sum_{m=1}^{C} p_{h|\ell,m}(\hat{\boldsymbol{\theta}})p_m(\hat{\boldsymbol{\gamma}})\Big)^{n_i^{\rightarrow}(h)}.$$

The proof is given in Appendix B.

Corollary III.2. *If the score graph is undirected, then we have*

$$v_i(c_\ell) = p_\ell(\hat{\boldsymbol{\gamma}})\pi_i^{\leftrightarrow}(c_\ell).$$

B. Joint Parameter-Hyperparameter ML estimation (JPH-ML)

Classification requires that at each node an estimate $(\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\gamma}})$ of parameter-hyperparameter $(\boldsymbol{\theta},\boldsymbol{\gamma})$ is available. In this regard, a few remarks about $\boldsymbol{\theta}$ and $\boldsymbol{\gamma}$ are now in order.

Depending on both the application and the network context, these parameters may be known, or (partially) unknown to the nodes. If both of them are known, we are in a pure Bayesian set-up in which, as just shown, each node can independently self-classify with no need of cooperation. The case of unknown $\boldsymbol{\theta}$ (and known $\boldsymbol{\gamma}$) falls into a Maximum Likelihood framework, while the case of unknown $\boldsymbol{\gamma}$ (and known $\boldsymbol{\theta}$) can be addressed by an *Empirical Bayes* approach. In this paper we consider a general scenario in which both of them may be unknown. Our goal is then to compute, in a distributed way, an estimate of *parameter-hyperparameter* $(\boldsymbol{\theta},\boldsymbol{\gamma})$ and use it for classification at each node.

In the following we show how to compute such an estimator in a distributed way by following a mixed Empirical Bayes and Maximum Likelihood approach. We define the *Joint Parameter-Hyperparameter Maximum Likelihood (JPH-ML) estimator* as

$$(\hat{\boldsymbol{\theta}}_{\text{ML}},\hat{\boldsymbol{\gamma}}_{\text{ML}}) := \operatorname*{argmax}_{(\boldsymbol{\theta},\boldsymbol{\gamma})\in\mathcal{S}_\Theta\times\mathcal{S}_\Gamma} L(\boldsymbol{y}_{E_S};\boldsymbol{\theta},\boldsymbol{\gamma}) \qquad (5)$$

where $\boldsymbol{y}_{E_S}$ is the vector of all scores $y_{ji}, (j,i) \in E_S$, and

$$L(\boldsymbol{y}_{E_S}; \boldsymbol{\theta}, \boldsymbol{\gamma}) = \mathbb{P}(\boldsymbol{Y}_{E_S} = \boldsymbol{y}_{E_S}\,;\, \boldsymbol{\theta}, \boldsymbol{\gamma}) \tag{6}$$

is the *likelihood function*.

Notice that, while $\boldsymbol{\theta}$ is directly linked to the observables $\boldsymbol{y}_{E_S}$, the hyperparameter $\boldsymbol{\gamma}$ is related to the unobservable states. While one could readily obtain the likelihood function for the sole estimation of $\boldsymbol{\theta}$ from the distribution of scores, the presence of $\boldsymbol{\gamma}$ requires to marginalize out all unobservable state (random) variables. By the law of total probability

$$\begin{aligned} &L(\boldsymbol{y}_{E_S}; \boldsymbol{\theta}, \boldsymbol{\gamma}) = \\ &\sum_{\ell_1=1}^{C} \cdots \sum_{\ell_N=1}^{C} \mathbb{P}(\boldsymbol{Y}_{E_S} = \boldsymbol{y}_{E_S}, X_1 = c_{\ell_1}, \ldots, X_N = c_{\ell_N}). \end{aligned} \tag{7}$$

Denoting by N_i^I the set of in-neighbors of agent i in the score graph (we are assuming that it is non-empty), the probability in (7) can be written as the product of the conditional probability of scores, i.e.,

$$\begin{aligned} &\mathbb{P}(\boldsymbol{Y}_{E_S} = \boldsymbol{y}_{E_S} \mid X_1 = c_{\ell_1}, \ldots, X_N = c_{\ell_N}) = \\ &\qquad \prod_{i=1}^{N} \prod_{j \in N_i^I} \mathbb{P}(Y_{ji} = y_{ji} | X_j = c_{\ell_j}, X_i = c_{\ell_i}) \end{aligned}$$

multiplied by the prior probability of states, i.e.,

$$\mathbb{P}(X_1 = c_{\ell_1}, \ldots, X_N = c_{\ell_N}) = \prod_{i=1}^{N} \mathbb{P}(X_i = c_{\ell_i}).$$

Thus, the likelihood function turns out to be

$$L(\boldsymbol{y}_{E_S}; \boldsymbol{\theta}, \boldsymbol{\gamma}) = \sum_{\ell_1=1}^{C} \cdots \sum_{\ell_N=1}^{C} \prod_{i=1}^{N} g_i(\boldsymbol{\theta}, \boldsymbol{\gamma}; \ell_1, \ldots, \ell_N),$$

where

$$g_i(\boldsymbol{\theta}, \boldsymbol{\gamma}; \ell_1, \ldots, \ell_N) = p_{\ell_i}(\boldsymbol{\gamma}) \prod_{j \in N_i^I} p_{h_{ji}|\ell_j, \ell_i}(\boldsymbol{\theta}),$$

with h_{ji} the index of the score element $r_{h_{ji}} \in \mathcal{R} = \{r_1, \ldots, r_R\}$ associated to the score y_{ji}, i.e., $y_{ji} = r_{h_{ji}}$.

Equations above clearly show that data from all nodes are coupled in the likelihood function in a nontrivial way. As typical in distributed computation approaches, we may try to manipulate the problem formulation in order to obtain a "separable" structure by introducing copies of the decision variables together with consistency constraints. Specifically, we can introduce a collection of C^N additional variables, one for each $g_i(\boldsymbol{\theta}, \boldsymbol{\gamma}; \ell_1, \ldots, \ell_N)$, and denote them $\rho_{[\ell_1 \cdots \ell_N]}$, $\ell_1, \ldots, \ell_N \in \{1, \ldots, C\}$. Problem (5) is, thus, equivalent to

$$\begin{aligned} \max_{\substack{\rho_{[\ell_1 \cdots \ell_N]} \in \mathbb{R}, \\ (\boldsymbol{\theta}, \boldsymbol{\gamma}) \in \mathcal{S}_\Theta \times \mathcal{S}_\Gamma}} \quad & \sum_{\ell_1=1}^{C} \cdots \sum_{\ell_N=1}^{C} \rho_{[\ell_1 \cdots \ell_N]}, \\ \text{subj. to} \quad & \rho_{[\ell_1 \cdots \ell_N]} \leq \prod_{i=1}^{N} g_i(\boldsymbol{\theta}, \boldsymbol{\gamma}; \ell_1, \ldots, \ell_N), \end{aligned}$$

which in turn is equivalent to

$$\begin{aligned} \max_{\substack{\rho_{[\ell_1 \cdots \ell_N]} \in \mathbb{R}_{>0}, \\ (\boldsymbol{\theta}, \boldsymbol{\gamma}) \in \mathcal{S}_\Theta \times \mathcal{S}_\Gamma}} \quad & \sum_{\ell_1=1}^{C} \cdots \sum_{\ell_N=1}^{C} \rho_{[\ell_1 \cdots \ell_N]}, \\ \text{subj. to} \quad & \log(\rho_{[\ell_1 \cdots \ell_N]}) \leq \sum_{i=1}^{N} \log(g_i(\boldsymbol{\theta}, \boldsymbol{\gamma}; \ell_1, \ldots, \ell_N)). \end{aligned} \tag{8}$$

Notice that the constraints are always well-defined because g_i is a positive function for each $i = 1, \ldots, N$. In (8) we recognize a separable structure in the cost and the constraints, which can lead to distributed solutions. However this optimization problem involves C^N constraints per node and $C^N + d_{\boldsymbol{\theta}} + d_{\boldsymbol{\gamma}}$ decision variables, with $d_{\boldsymbol{\theta}}$ and $d_{\boldsymbol{\gamma}}$ the number of components of $\boldsymbol{\theta}$ and $\boldsymbol{\gamma}$. Thus, in order to solve it in a distributed way, one should address the challenging task of designing distributed optimization algorithms capable to handle a number of decision variables and constraints which is exponential in the number of nodes. In view of these considerations, one might argue whether following such a route is really necessary for the nodes to obtain a satisfactory solution for the classification problem. In the rest of paper we propose a different methodological approach based on suitable relaxations of the likelihood function and show, through numerical examples, that such solutions provide a satisfactory classification outcome. In particular, we propose a relaxation approach for the likelihood function which leads to separable and computationally tractable optimization problems.

Remark III.3 (Alternative Estimation Methods)**.** *Before proceeding, one might argue whether other alternative estimation approaches may be more convenient; in particular, it is known that the popular Expectation Maximization (EM) estimation approach may offer valuable solutions when ML cannot be obtained or it is computationally expensive. However, it is a simple matter to show that, for the problem at hand, the maximization involved in the EM procedure requires to evaluate the sum of C^N terms, thus the same complexity order as the ML approach.*

C. Distributed JPH Node-based Relaxed estimation (JPH-NR)

We introduce, instead of $L(\boldsymbol{y}_{E_S}; \boldsymbol{\theta}, \boldsymbol{\gamma})$, a *Node-based Relaxed (NR) likelihood* $L_{NR}(\boldsymbol{y}_{E_S}; \boldsymbol{\theta}, \boldsymbol{\gamma})$. Let $\boldsymbol{y}_{N_i^I}$ be the vector of (observed) scores that agent i obtains by in-neighbors and $\boldsymbol{Y}_{N_i^I}$ the corresponding random vector. Then,

$$L_{NR}(\boldsymbol{y}_{E_S}; \boldsymbol{\theta}, \boldsymbol{\gamma}) := \prod_{i=1}^{N} \mathbb{P}(\boldsymbol{Y}_{N_i^I} = \boldsymbol{y}_{N_i^I}\,;\, \boldsymbol{\theta}, \boldsymbol{\gamma}). \tag{9}$$

This relaxation can be interpreted as follows. We suppose each node has a virtual state, independent of its true state, every time it evaluates another node. Thus, in the Score Bayesian Network, besides the state variables X_i, $i = 1, \ldots, N$, there will be additional variables $X_i^{\to j}$ for each j with $(i,j) \in E_S$. To clarify this model, Figs. 5-6 depict the node-based relaxed graph and the corresponding graphical model for the same example given in Figs. 1-2.

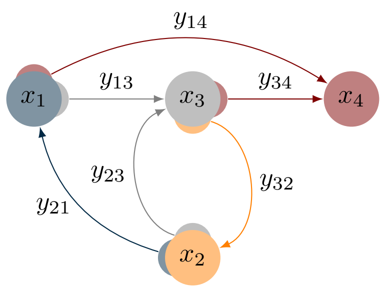

Fig. 5. Node-based relaxation of the score graph in Fig. 1, with virtual nodes indicating the virtual states of each node.

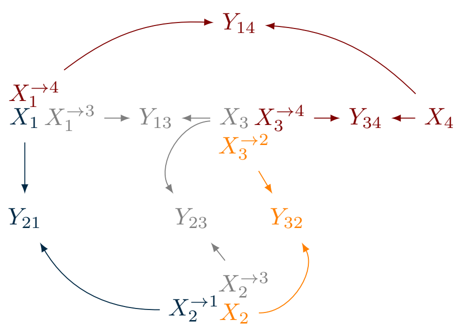

Fig. 6. Node-based relaxation of the score Bayesian network of Fig. 5.

Since $\boldsymbol{Y}_{N_i^I}$, $i = 1, \dots, N$, are not independent, then clearly $L \neq L_{NR}$. However, as it will be apparent from the numerical performance assessment, reported in the Section IV, this choice yields reasonably small estimation errors.

Using this virtual independence between $\boldsymbol{Y}_{N_i^I}$, with $i = 1, \dots, N$, we define the *JPH-NR estimator* as

$$(\hat{\boldsymbol{\theta}}_{\text{NR}}, \hat{\gamma}_{\text{NR}}) := \underset{(\boldsymbol{\theta},\boldsymbol{\gamma})\in\mathcal{S}_\Theta\times\mathcal{S}_\Gamma}{\operatorname{argmax}} \; L_{NR}(\boldsymbol{y}_{E_S}; \boldsymbol{\theta}, \boldsymbol{\gamma}). \tag{10}$$

The next result characterizes the structure of JPH-NR (10).

Proposition III.4. *The JPH-NR estimator based on the node-based relaxation of the score Bayesian network is given by*

$$(\hat{\boldsymbol{\theta}}_{\text{NR}}, \hat{\boldsymbol{\gamma}}_{\text{NR}}) = \underset{(\boldsymbol{\theta},\boldsymbol{\gamma})\in\mathcal{S}_\Theta\times\mathcal{S}_\Gamma}{\operatorname{argmax}} \sum_{i=1}^{N} g(\boldsymbol{\theta}, \boldsymbol{\gamma}; \boldsymbol{n}_i) \tag{11}$$

with $\boldsymbol{n}_i = [n_i^{(1)} \cdots n_i^{(R)}]^\top$, $n_i^{(h)} := |\{j \in N_i^I : y_{ji} = r_h\}|$, *and*

$$g(\boldsymbol{\theta}, \boldsymbol{\gamma}; \boldsymbol{n}_i) = \log \Big(\sum_{\ell=1}^{C} p_\ell(\boldsymbol{\gamma}) \prod_{h=1}^{R} \Big(\sum_{m=1}^{C} p_{h\,|\,m,\ell}(\boldsymbol{\theta}) p_m(\boldsymbol{\gamma}) \Big)^{n_i^{(h)}} \Big). \tag{12}$$

The proof is given in Appendix C.

Proposition III.4 ensures that the JPH-NR estimator can be computed by solving an optimization problem that has a separable cost (i.e., the sum of N local costs).

Available distributed optimization algorithms for asynchronous networks can be adopted to this aim, e.g. [29], [30], [31]. This means that the distributed JPH-NR estimation algorithm inherits the convergence rate of the chosen distributed optimization algorithm.

D. Distributed JPH Fully-Relaxed estimation (JPH-FR)

Although JPH-NR estimator is a viable solution for which we will report simulation results later in the paper, we consider a stronger relaxation, which gives rise to a more convenient distributed algorithm consisting of a linear (consensus-like) averaging process and a purely local optimization step.

Thus, we introduce the *Fully Relaxed (FR) likelihood*:

$$L_{FR}(\boldsymbol{y}_{E_S}; \boldsymbol{\theta}, \boldsymbol{\gamma}) := \prod_{(i,j)\in E_S} \mathbb{P}(Y_{ij} = y_{ij} \,;\, \boldsymbol{\theta}, \boldsymbol{\gamma}) \tag{13}$$

where all dependencies among the variables $Y_{ij}, (i,j) \in E_S$ are neglected. Accordingly, the *JPH-FR estimator* is given by

$$(\hat{\boldsymbol{\theta}}_{\text{FR}}, \hat{\boldsymbol{\gamma}}_{\text{FR}}) := \underset{(\boldsymbol{\theta},\boldsymbol{\gamma})\in\mathcal{S}_\Theta\times\mathcal{S}_\Gamma}{\operatorname{argmax}} \; L_{FR}(\boldsymbol{y}_{E_S}; \boldsymbol{\theta}, \boldsymbol{\gamma}). \tag{14}$$

The following proposition exposes the structure of (14).

Proposition III.5. *The JPH-FR estimator based on the full relaxation of the score Bayesian network is given by*

$$(\hat{\boldsymbol{\theta}}_{\text{FR}}, \hat{\boldsymbol{\gamma}}_{\text{FR}}) = \underset{(\boldsymbol{\theta},\boldsymbol{\gamma})\in\mathcal{S}_\Theta\times\mathcal{S}_\Gamma}{\operatorname{argmin}} \; \boldsymbol{\phi}^\top \boldsymbol{g}(\boldsymbol{\theta}, \boldsymbol{\gamma}) \tag{15}$$

where $\boldsymbol{g} := (g_{(1)}, \dots, g_{(R)})$, $\quad \boldsymbol{\phi} := (\phi^{(1)}, \dots, \phi^{(R)})$, *and*

$$g_{(h)}(\boldsymbol{\theta}, \boldsymbol{\gamma}) := -\log \Big(\sum_{\ell,m=1}^{C} p_{h|\ell,m}(\boldsymbol{\theta}) p_\ell(\boldsymbol{\gamma}) p_m(\boldsymbol{\gamma}) \Big),$$

$$\phi^{(h)} := \sum_{i=1}^{N} \frac{n_i^{(h)}}{n}, \qquad h = 1, \dots, R$$

The proof is given in Appendix D.

In order to solve the optimization problem (15) in a distributed way, each agent in the network needs to know the vector $\boldsymbol{\phi}$. A naive approach is to first run a consensus algorithm to obtain approximated local "copies" of $\boldsymbol{\phi}$; then, (15) can be solved by applying a standard (centralized) optimization method, e.g. the projected gradient method. However, in this approach one needs to wait for the consensus algorithm to converge (up to the required accuracy), then to start another iterative (local) procedure to finally obtain the solution. We propose here a different approach, where only a single iterative (distributed) procedure is run. The idea is to combine one step of consensus with one step of gradient in order to build a sequence that converges to an optimal solution.

Let n_i be the number of incoming edges of agent i into the score graph G_S, i.e., $n_i = |N_i^I|$. For each $t \in \mathbb{Z}_{\geq 0}$, agent i stores in memory two local states $\boldsymbol{\xi}_i(t) = (\xi_i^{(1)}(t), \dots, \xi_i^{(R)}(t))$ and $\eta_i(t)$, an estimate $\boldsymbol{\phi}_i(t)$ of $\boldsymbol{\phi}$, and an estimate $(\hat{\boldsymbol{\theta}}_i(t), \hat{\boldsymbol{\gamma}}_i(t))$ of $(\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\gamma}})$.

By following the push-sum consensus algorithm to compute averages in directed graphs [32], we provide the following distributed algorithm to compute $\boldsymbol{\phi}$. By denoting $d_j(t) =$

$|N^{O}_{\mathrm{cmm},i}(t)|$ the out-degree of node j at time t in the communication graph $G_{\mathrm{cmm}}(t)$, node i implements

$$\begin{aligned}\xi_i^{(h)}(t+1) &= \sum_{j\in N_i^{\mathrm{cmm},I}(t)} \frac{\xi_j^{(h)}(t)}{d_j(t)}\\ \eta_i(t+1) &= \sum_{j\in N_i^{\mathrm{cmm},I}(t)} \frac{\eta_j(t)}{d_j(t)} \\ \phi_i^{(h)}(t+1) &= \frac{\xi_i^{(h)}(t+1)}{\eta_i(t+1)}\end{aligned} \tag{16}$$

with $\xi_i^{(h)}(0) = n_i^{(h)}$ and $\eta_i(0) = n_i$.

Then, each node can use its current estimate $\phi_i^{(h)}(t)$ to implement a gradient step on the estimated cost function $\boldsymbol{\phi}_i(t)^\top \boldsymbol{g}$. That is, let $(\hat{\boldsymbol{\theta}}_{i,0}, \hat{\boldsymbol{\gamma}}_{i,0}) \in \mathcal{S}_{\boldsymbol{\theta}} \times \mathcal{S}_{\boldsymbol{\gamma}}$ be a starting point for the distributed estimation algorithm, $\alpha > 0$ a suitable step-size, then $(\hat{\boldsymbol{\theta}}_i(0), \hat{\boldsymbol{\gamma}}_i(0)) = (\hat{\boldsymbol{\theta}}_{i,0}, \hat{\boldsymbol{\gamma}}_{i,0})$ and

$$\begin{aligned}&(\hat{\boldsymbol{\theta}}_i(t+1), \hat{\boldsymbol{\gamma}}_i(t+1)) = \\ &\qquad \Big[(\hat{\boldsymbol{\theta}}_i(t), \hat{\boldsymbol{\gamma}}_i(t)) - \alpha \boldsymbol{\phi}_i(t)^\top \nabla \boldsymbol{g}(\hat{\boldsymbol{\theta}}_i(t), \hat{\boldsymbol{\gamma}}_i(t))\Big]^+,\end{aligned} \tag{17}$$

with $[\cdot]^+$ the (Euclidean) projection operator onto the feasible set $\mathcal{S}_\Theta \times \mathcal{S}_\Gamma$.

The following technical assumption ensures uniqueness of the projection.

Assumption III.6. *The given sets Θ and Γ are both subsets of finite-dimensional real-vector spaces, and the product set $\mathcal{S}_\Theta \times \mathcal{S}_\Gamma$ is compact and convex.*

The convergence properties of the distributed algorithm defined by (16)-(17) are given in the following theorem.

Theorem III.7. *Let Assumptions II.1 and III.6 hold. Suppose that $\boldsymbol{g}$ is differentiable, and that $\nabla \boldsymbol{g}$ is bounded and Lipschitz continuous, with constant $L > 0$, over the feasible set $\mathcal{S}_\Theta \times \mathcal{S}_\Gamma$. Let $0 < \alpha < \frac{2}{L}$. Then, any limit point of the sequence $\{(\hat{\boldsymbol{\theta}}_i(t), \hat{\boldsymbol{\gamma}}_i(t))\}_{t\in\mathbb{N}}$ generated by* (16)-(17) *is a stationary point of the objective function $\boldsymbol{\phi}^\top \boldsymbol{g}$ over the feasible set $\mathcal{S}_\Theta \times \mathcal{S}_\Gamma$.*

The proof is given in Appendix E.

Remark III.8. *We observe that in the distributed JPH-FR estimation algorithm, defined by equations* (16) *and* (17), *the local copies $\phi_i^{(h)}(t)$ exhibit exponential convergence to the average consensus value, as recalled in Lemma A.5. Thus, the distributed JPH-FR estimation algorithm turns out to be a projected subgradient algorithm with an error converging to zero exponentially fast.*

IV. Application of the framework

In this section we provide numerical results for two meaningful case studies, using the anomaly detection and the social ranking models described in Section II-C. Beforehand, we analyze a special binary/binary case for which the JPH-FR estimator can be computed in closed form.

A. Distributed learning for anomaly detection: binary scores and binary states

In this section we discuss the anomaly detection model described in Section II-C, reported here for the sake of clarity:

$$\begin{aligned}p_{h|\ell,m} &= (1-c_\ell)\Big[(1-c_m)(1-r_h) + c_m r_h\Big] + \frac{1}{2}c_\ell,\\ p_\ell(\gamma) &= \gamma^{c_\ell}(1-\gamma)^{1-c_\ell}, \qquad \gamma \in [0,1],\end{aligned}$$

with $R = 2$ $(r_1 = 0, r_2 = 1)$ and $C = 2$ $(c_1 = 0, c_2 = 1)$.

Using Lemma A.4, the fully relaxed likelihood is obtained:

$$L_{FR}(\gamma) = \Big[\frac{1}{2}\gamma + \gamma(1-\gamma)\Big]^{n^{(2)}} \Big[\frac{1}{2}\gamma + (1-\gamma)^2\Big]^{n^{(1)}}.$$

By studying the roots of its derivative, one can show, see [6], that for this model a closed form for $\hat{\gamma}_{\mathrm{FR}}$ (depending on $\boldsymbol{\phi}$) can be found. Hence, only a (distributed) consensus algorithm is needed to compute $\boldsymbol{\phi}$.

Besides this special (binary-binary) case, in general it is not possible to find a closed form, hence the proposed distributed estimation schemes are needed in practice.

B. Distributed learning for anomaly detection: R-ary scores and binary states

We consider an extension of the previous scenario by allowing scores to assume multiple values. We basically relax the fact that a normally working node gives the exact state of the tested node in a deterministic way. We assume that the R possible scores are given according to some probability, depending, e.g., on the reliability of the test or expressing the level of trust about the tested node. For the sake of clarity we just consider a linear trend. Formally, we let

$$\begin{aligned}&R \geq 2, \qquad r_h = h-1, \quad h = 1,\dots,R,\\ &C = 2, \qquad c_1 = 0, \quad c_2 = 1,\end{aligned}$$

and consider the following probabilistic model:

$$\begin{aligned}p_{h|\ell,m} &= \frac{2}{R}(1-c_\ell)\Big[(1-c_m)\Big(1-\frac{r_h}{r_R}\Big) + c_m\frac{r_h}{r_R}\Big] + \frac{c_\ell}{R},\\ p_\ell(\gamma) &= \gamma^{c_\ell}(1-\gamma)^{1-c_\ell}, \qquad \gamma \in [0,1].\end{aligned}$$

Notice that this model, hereafter referred to as the *reliability model*, boils down to the Preparata's model [25] for the case of binary score ($R = 2$, see Sec. II-C1).

We have performed Monte Carlo simulations, with 5000 trials for each point, for a score graph with $N = 300$ agents according to the probabilistic model above, for $R = 5$ and $\gamma = 0.3$. As for the score graph, we considered a sequence of scenarios by starting from a directed cyclic configuration and then, progressively, adding edges up to 10^4 edges.

We considered the reliability model to compute both the JPH-NR and the JPH-FR estimator of the hyperparameter, which were then used to perform the classification. Results of these simulations are shown in Fig. 7, where the relative Root Mean Square Error (RMSE) of the estimates of γ is reported. We observe that in this case both the JPH-NR and JPH-FR estimators attain a very low relative RMSE, with misclassification rates very close to an "oracle" classifier that knows the true value of γ. Clearly, this is a favorable situation. In the following we will stress the estimation set-up to highlight differences among the algorithms.

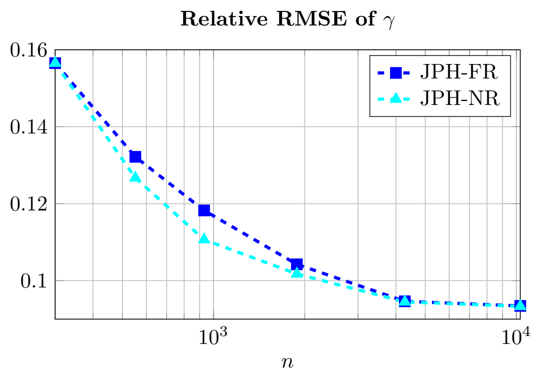

Fig. 7. Relative RMSE of the estimates of γ as a function of the number of edges n from N (cycle graph) to 10^4, with $N = 300$, $\gamma = 0.3$ and $R = 5$.

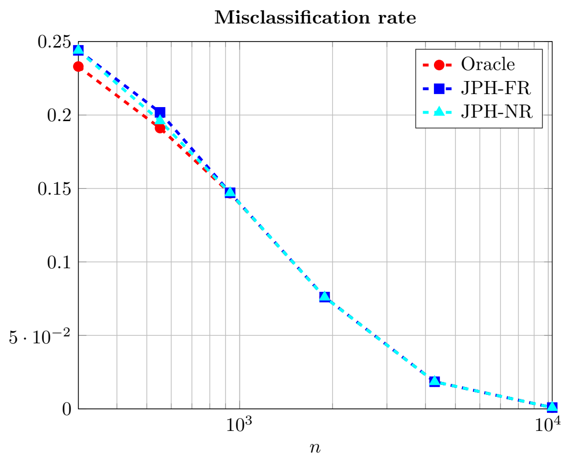

Fig. 8. Misclassification rate as a function of the number of edges n from N (cycle graph) to 10^4, with $N = 300$, $\gamma = 0.3$ and $R = 5$.

C. Distributed learning for social ranking

In this Section we show results for the social ranking model described in Section II-C, reported here for the sake of clarity:

$$p_{h|\ell,m}(\theta) = \frac{1}{\psi_{\ell,m}(\theta)} e^{-\left(\frac{(r_R - r_h)/r_R - d(c_\ell, c_m)/c_C}{\theta}\right)^2},$$

$$p_\ell(\gamma) = \binom{C-1}{c_\ell - 1} \gamma^{c_\ell - 1} (1-\gamma)^{C-1-(c_\ell - 1)},$$

where $r_h = h$, $c_\ell = \ell$, while $\theta \in \mathbb{R}_{\geq 0}$, $\gamma \in [0, 1]$. We use as semi-distance $d(c_\ell, c_m) = |c_\ell - c_m| = |\ell - m|$.

As in the previous fault detection scenario, there exist configurations of parameters and network in which the two estimators exhibit almost the same performance. For instance for $C = 3$ communities and $R = 3$ possible scores, with parameters $\theta = 0.1$ and $\gamma = 0.3$, and a sequence of score graphs with $N = 300$ nodes and edges added as before, we have obtained a misclassification error for the two estimators whose deviation from the oracle is at most 1.4%.

Next, we consider two scenarios that aim at stressing the differences among the estimators. In particular, we consider a set-up with few nodes in which correlations should play a stronger role. Then, we consider a more challenging classification problem with a higher number of classes and a lower number of available score values. Specifically, in the first set-up we have reduced the network size to $N = 30$ and kept the same values for the other parameters. In the second set-up we have kept the same number of agents ($N = 300$) and parameters θ and γ, and changed the number of communities and scores to $C = 5$ and $R = 2$. As before, we have increased the number of edges in the score graph and run Monte Carlo simulations with 5000 trials for each configuration of the score graph. The misclassification rates for the classifiers based on the JPH-NR and JPH-FR estimators, and for the oracle are reported in Fig. 9 and Fig. 10, respectively for the first and second scenario. As appearing from these pictures, the error curves of the proposed classifiers deviate from the oracle when the number of edges is low. While a gap remains for the JPH-FR curve, when the number of edges increases the JPH-NR curve approaches the oracle in both the analyzed scenarios. In general, it worth noting that the misclassification error decreases for all the classifiers when increasing the number of edges. In particular, in the second scenario, even with a moderate size of the network ($N = 300$) and a relatively sparse graph (at most $n = 10^4$, which is roughly 10% of the edges of the complete graph), about 85% of the nodes correctly classify their own state (among 5 possibilities) when the JPH-NR estimator is used.

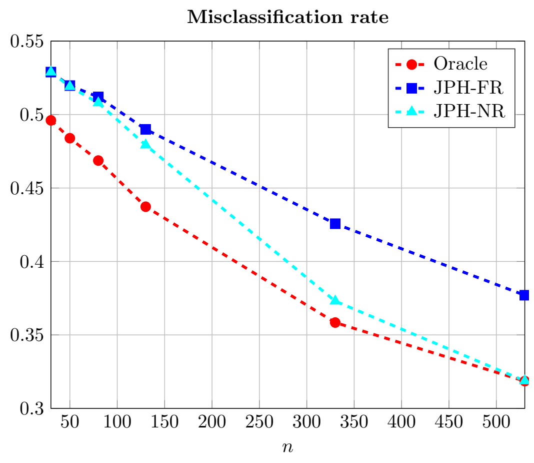

Fig. 9. Misclassification rate as a function of the number of edges n from N (cycle graph) to 500, with $N = 30$, $\gamma = 0.3$, $\theta = 0.1$, $C = 3$ and $R = 3$.

Finally, we report an additional case to highlight the useful insights given by the soft classifier. We considered a network of $N = 10$ agents, whose score graph G_S is shown in Fig. 11. We drew the states and scores in the given score graph according to the previous distributions, and then used the social ranking model to solve the learning problem as before, by means of the JPH-FR estimator.

The contour of a node has a color that indicates the true state of the node. Inside the node we have represented the outcome of the soft classification, i.e., the output of the local self-classifier, as a pie-chart. The colors used are: red for state 1, blue for state 2, gray for state 3. Moreover, each edge is depicted by a different pattern based on its evaluation result r_h: solid lines are related to scores equal to 3, dash dot lines are related to scores equal to 2, while dotted lines are related

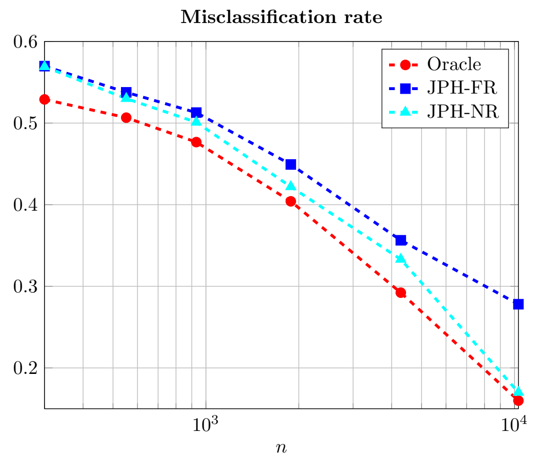

Fig. 10. Misclassification rate as a function of the number of edges n from N (cycle graph) to 10^4, with $N = 300$, $\gamma = 0.3$, $\theta = 0.1$, $C = 5$ and $R = 2$.

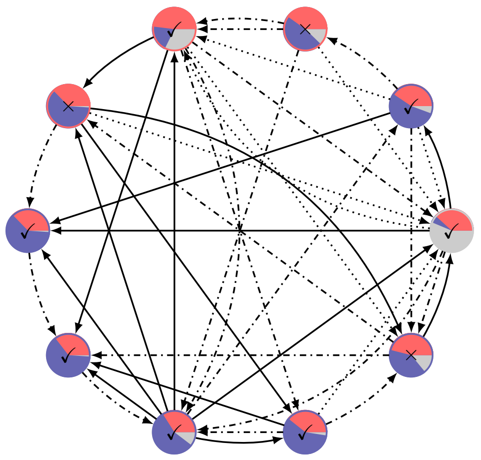

Fig. 11. Soft classifier representation of a particular score graph.

to scores equal to 1. We associated to each node a symbol ✓ or $\times$ indicating if the MAP classifier correctly decided for the true state or not. We show a realization with three misclassification errors; remarkably, all of them correspond to a lower confidence level given by the soft classifier, which is an important indicator of the lack of enough information to reasonably trust the decision. It can be observed that the edge patterns concur to determine the decision. Indeed, the only gray-state node is correctly classified thanks to the predominant number of dotted edges insisting on it, and similarly for the blue-state nodes which mostly have solid incoming edges. When a mix of scores are available, clearly there is more uncertainty and the learning may fail, as for two of the red-state nodes.

As a final remark, we point out that in some scenarios symmetries may arise in the model, thus creating ambiguities in the labeling of the communities. Specifically, for this scenario, it can be shown that the relaxed likelihoods L_{NR} and L_{FR} take on the same value when γ is replaced by $1-\gamma$, e.g., $L_{NR}(\boldsymbol{y}_{E_S};\boldsymbol{\theta},\boldsymbol{\gamma}) = L_{NR}(\boldsymbol{y}_{E_S};\boldsymbol{\theta},1-\boldsymbol{\gamma})$. Thus the hyperparameter estimate is not unique in this case. However, this has just the effect of swapping the label of communities c_1 and c_3. Notice that, in the node-based relaxed case, agents reach directly a consensus on the same value, thus circumventing possible inconsistencies in the labeling. For the fully relaxed case agents can easily agree on the same value in a number of steps equal to the diameter.

V. Conclusion

In this paper we have proposed a novel probabilistic framework for distributed learning, which is particularly relevant to emerging contexts such as cyber-physical and social networks. In the proposed set-up, network nodes can learn their (unknown) state by exploiting interactions with neighbors, rather than direct measurements. For this problem we have proposed a (hierarchical) Bayesian framework in which the parameters of the interaction model as well as hyperparameters of the prior distributions may be unknown. Node classification is performed by means of a local Bayesian classifier and a joint parameter-hyperparameter ML estimator. The resulting scheme is very general but, unfortunately, the ML estimation part is computationally intractable in both centralized and distributed set-ups. Therefore, we have proposed two approximated ML estimators (exploiting proper relaxations of the conditional dependences among the involved random variables) that can be computed in a distributed way. To validate the proposed schemes, we have addressed two example scenarios from anomaly detection in cyber-physical networks and user profiling in social networks. Monte Carlo simulation results show that the proposed distributed learning schemes, although based on relaxations of the exact likelihood function, exhibit performance comparable with the ideal classifier that has perfect knowledge of all parameters.

Appendix

A. Preliminaries on Bayesian Networks

We recall some definitions and results from graphical model theory. We need to assess when two random variables Z_{in} and Z_{fin} in a Bayesian Network structure $\mathcal{K}$ are conditionally independent given another variable Z_g. We use the shorthand notation $Z_i \rightleftharpoons Z_{i+1}$ meaning that either $Z_i \to Z_{i+1}$, or $Z_i \leftarrow Z_{i+1}$, or both hold.

Definition A.1. *Given a graphical model $\mathcal{K}$, we say that $Z_0, \dots, Z_r$ form a* trail *in $\mathcal{K}$ if, for every $i = 0, \dots, r-1$, we have $Z_i \rightleftharpoons Z_{i+1}$. If, for every $i = 0, \dots, r-1$, we have that $Z_i \to Z_{i+1}$, then the trail is called a* directed path.

Definition A.2. *We say that Z_d is a* descendant *of Z in the graph $\mathcal{K}$ if there exists a directed path from Z to Z_d.*

When influence can flow from Z_{in} to Z_{fin} via Z_g, we say that the two-arrow trail $Z_{\text{in}} \rightleftharpoons Z_g \rightleftharpoons Z_{\text{fin}}$ is *active*. For each of the four possible two-arrow trails, we detail the condition under which it is active:

- *Causal trail* ($Z_{\text{in}} \to Z_g \to Z_{\text{fin}}$): active if and only if Z_g is not observed.
- *Evidential trail* ($Z_{\text{in}} \leftarrow Z_g \leftarrow Z_{\text{fin}}$): active if and only if Z_g is not observed.
- *Common cause* ($Z_{\text{in}} \leftarrow Z_g \to Z_{\text{fin}}$): active if and only if Z_g is not observed.

- *Common effect* ($Z_{\text{in}} \to Z_g \leftarrow Z_{\text{fin}}$): active if and only if either Z_g or one of its descendants is observed.

Now, consider the case of a longer trail $Z_{\text{in}} = Z_0 \rightleftharpoons \cdots \rightleftharpoons Z_r = Z_{\text{fin}}$. Intuitively, for influence to *flow* from Z_{in} to Z_{fin}, it needs to flow through every single node on the trail. In other words, Z_0 can influence Z_r if for every $i = 1, \ldots, r-1$, then $Z_{i-1} \rightleftharpoons Z_i \rightleftharpoons Z_{i+1}$ is active.

Obviously, it can happen that there is more than one trail between two nodes; in these cases one node can influence another if and only if there exists a trail along which influence can flow. If there is no active trail between random variables Z_{in} and Z_{fin}, given Z_g, they are said to be *d-separated.*

B. Proof of Theorem III.1

The following Lemma is used in the proof of Theorem III.1.

Lemma A.3. *Let $i, j, k \in \{1, \ldots, N\}$ with $j \neq k$. The following statements hold:*

1) *if $(i, j), (k, i) \in E_S$ then Y_{ij} and Y_{ki} are conditionally independent given X_i;*
2) *if $(i, j), (i, k) \in E_S$ then Y_{ij} and Y_{ik} are conditionally independent given X_i;*
3) *if $(j, i), (k, i) \in E_S$ then Y_{ji} and Y_{ki} are conditionally independent given X_i;*

Proof. We prove only the first statement, the other two can be proven in the same way. Consider a trail $Y_{ij} = Z_0 \rightleftharpoons \cdots \rightleftharpoons Z_r = Y_{ki}$. Denoting by s the number of state random variables traversed along the trail, then the length of the trail (number of arrows) is $r = 2s$. This property can be easily visualized in Figure 2. For each $u = 0, \ldots, s-1$ it results:

$$Z_{2u} \leftarrow Z_{2u+1} \rightarrow Z_{2u+2}$$

with $Z_{2u} = Y_{i_u j_u}$, $Z_{2u+1} = X_{v_u}$, $Z_{2u+2} = Y_{i_{u+1} j_{u+1}}$, where we have that $v_u \in \{i_u, j_u\} \cap \{i_{u+1}, j_{u+1}\}$, while $(i_u, j_u), (i_{u+1}, j_{u+1}) \in E_S$.

Our objective is to prove that the previous trail is blocked (i.e., not active); this will imply that Y_{ij}, Y_{ki} are d-separated given X_i, so that the proof follows, [33]. We observe that if $X_{v_0} = X_i$, then we have, inside the trail, the common cause $Z_0 \leftarrow X_i \rightarrow Z_2$ in which X_i is observed; thus the previous common cause is blocked, implying that also the trail is blocked. Next, we prove that block occurs at most for $u = 1$. Consider $X_{v_0} \neq X_i$. In this case, from the assumption $j \neq k$, by contradiction, we find that $s > 1$. By truncating the trail at the fourth element, we have:

$$Z_0 \leftarrow X_{v_0} \rightarrow Z_2 \leftarrow X_{v_1}.$$

In the common effect $X_{v_0} \rightarrow Z_2 \leftarrow X_{v_1}$, Z_2 is not observed, and $Z_2 = Y_{i_1 j_1}$ has no descendant in the graphical model; thus the previous common effect is blocked, implying that also the trail is blocked. □

After these preliminaries, we can proceed with the proof of Theorem III.1. We will omit the dependency on $\boldsymbol{\theta}$ and $\boldsymbol{\gamma}$.

From the Bayes theorem, we know that:

$$u_i(c_\ell) = \frac{v_i(c_\ell)}{\mathbb{P}(\boldsymbol{Y}_{N_i} = \boldsymbol{y}_{N_i})},$$

where $v_i(c_\ell) := \mathbb{P}(\boldsymbol{Y}_{N_i} = \boldsymbol{y}_{N_i}, X_i = c_\ell)$. Our goal is to prove that $v_i(c_\ell) = p_\ell(\hat{\gamma})\pi_i^{\leftrightarrow}(c_\ell)\pi_i^{\leftarrow}(c_\ell)\pi_i^{\rightarrow}(c_\ell)$. First of all, from the chain rule we have:

$$v_i(c_\ell) = \mathbb{P}(X_i = c_\ell)\mathbb{P}(\boldsymbol{Y}_{N_i} = \boldsymbol{y}_{N_i} | X_i = c_\ell), \tag{18}$$

and from Lemma A.3, we know that

$$\begin{aligned} \mathbb{P}(\boldsymbol{Y}_{N_i} = \boldsymbol{y}_{N_i} | X_i = c_\ell) &= \mathbb{P}(\boldsymbol{Y}_{N_i^{\leftrightarrow}} = \boldsymbol{y}_{N_i^{\leftrightarrow}} | X_i = c_\ell) \\ &\quad \times \mathbb{P}(\boldsymbol{Y}_{N_i^{\leftarrow}} = \boldsymbol{y}_{N_i^{\leftarrow}} | X_i = c_\ell) \\ &\quad \times \mathbb{P}(\boldsymbol{Y}_{N_i^{\rightarrow}} = \boldsymbol{y}_{N_i^{\rightarrow}} | X_i = c_\ell). \end{aligned} \tag{19}$$

The next step is to study each one of the three factors. Starting from $\mathbb{P}(\boldsymbol{Y}_{N_i^{\leftrightarrow}} = \boldsymbol{y}_{N_i^{\leftrightarrow}} | X_i = c_\ell)$, we obtain:

$$\begin{aligned} &\mathbb{P}(\boldsymbol{Y}_{N_i^{\leftrightarrow}} = \boldsymbol{y}_{N_i^{\leftrightarrow}} | X_i = c_\ell) \\ &= \prod_{j \in N_i^{\leftrightarrow}} \mathbb{P}(Y_{ij} = y_{ij}, Y_{ji} = y_{ji} | X_i = c_\ell) \\ &= \prod_{h,k=1}^{R} \mathbb{P}(Y_{ij_0} = r_h, Y_{j_0 i} = r_k | X_i = c_\ell)^{n_i^{\leftrightarrow}(h,k)} \\ &= \prod_{h,k=1}^{R} \Big(\sum_{m=1}^{C} \mathbb{P}(Y_{ij_0} = r_h, Y_{j_0 i} = r_k, X_{j_0} = c_m | X_i = c_\ell) \Big)^{n_i^{\leftrightarrow}(h,k)} \\ &= \prod_{h,k=1}^{R} \Big(\sum_{m=1}^{C} \mathbb{P}(X_{j_0} = c_m) \mathbb{P}(Y_{ij_0} = r_h | X_i = c_\ell, X_{j_0} = c_m) \\ &\qquad \times \mathbb{P}(Y_{j_0 i} = r_k | X_{j_0} = c_m, X_i = c_\ell) \Big)^{n_i^{\leftrightarrow}(h,k)}. \end{aligned} \tag{20}$$

In the first equation we have used again Lemma A.3; in the second equation we have used the fact that $Y_{ij}, (i, j) \in E_S$ are identically distributed, and we have also aggregated the agents in $N_i^{\leftrightarrow}$ that receive/give score r_h/r_k from/to agent i; in the third equation we have marginalized with respect to the random variable X_{j_0}; in the fourth equation we have factorized according to the score Bayesian network.

For the second factor $\mathbb{P}(\boldsymbol{Y}_{N_i^{\leftarrow}} = \boldsymbol{y}_{N_i^{\leftarrow}} | X_i = c_\ell)$, we obtain:

$$\begin{aligned} &\mathbb{P}(\boldsymbol{Y}_{N_i^{\leftarrow}} = \boldsymbol{y}_{N_i^{\leftarrow}} | X_i = c_\ell) \\ &\quad = \prod_{j \in N_i^{\leftarrow}} \mathbb{P}(Y_{ji} = y_{ji} | X_i = c_\ell) \\ &\quad = \prod_{h=1}^{R} \mathbb{P}(Y_{j_0 i} = r_h | X_i = c_\ell)^{n_i^{\leftarrow}(h)} \\ &\quad = \prod_{h=1}^{R} \Big(\sum_{m=1}^{C} \mathbb{P}(Y_{j_0 i} = r_h, X_{j_0} = c_m | X_i = c_\ell) \Big)^{n_i^{\leftarrow}(h)} \\ &\quad = \prod_{h=1}^{R} \Big(\sum_{m=1}^{C} \mathbb{P}(X_i = c_m) \\ &\qquad \times \mathbb{P}(Y_{j_0 i} = r_h | X_{j_0} = c_m, X_i = c_\ell) \Big)^{n_i^{\leftarrow}(h)}. \end{aligned} \tag{21}$$

Here, we point out that, when using Lemma A.3 in the first equation, all the random variables $\boldsymbol{Y}_{N_i^{\leftarrow}}$ are conditionally independent given X_i. Also, factors in the second equation are aggregated based on the agents in the set $N_i^{\leftarrow}$ that give score r_h to agent i.

Finally, the third factor turns out to be:

$$\mathbb{P}(\boldsymbol{Y}_{N_i^{\rightarrow}} = \boldsymbol{y}_{N_i^{\rightarrow}} | X_i = c_\ell) \\ = \prod_{h=1}^{R} \Big(\sum_{c_m=1}^{C} \mathbb{P}(X_i = c_m) \\ \times \mathbb{P}(Y_{ij_0} = r_h | X_i = c_\ell, X_{j_0} = c_m) \Big)^{n_i^{\rightarrow}(h)}. \tag{22}$$

Plugging together (20), (21), (22) into (19), and then into (18), the proof is complete.

C. *Proof of Proposition III.4*

We omit the dependency on $\boldsymbol{\theta}$ and $\boldsymbol{\gamma}$ for notational purposes.

Consider a factor $\mathbb{P}(\boldsymbol{Y}_{N_i^I} = \boldsymbol{y}_{N_i^I})$ of the node-based relaxed likelihood, with i an agent in the score graph. Marginalizing with respect to X_i, we have

$$\mathbb{P}(\boldsymbol{Y}_{N_i^I} = \boldsymbol{y}_{N_i^I}) = \sum_{\ell=1}^{C} \mathbb{P}(\boldsymbol{Y}_{N_i^I} = \boldsymbol{y}_{N_i^I}, X_i = c_\ell).$$

Now, applying the chain rule we obtain

$$\mathbb{P}(\boldsymbol{Y}_{N_i^I} = \boldsymbol{y}_{N_i^I}) = \sum_{\ell=1}^{C} \mathbb{P}(X_i = c_\ell) \mathbb{P}(\boldsymbol{Y}_{N_i^I} = \boldsymbol{y}_{N_i^I} | X_i = c_\ell).$$

We are then ready to use Lemma A.3, which implies that

$$\mathbb{P}(\boldsymbol{Y}_{N_i^I} = \boldsymbol{y}_{N_i^I}) = \sum_{\ell=1}^{C} \mathbb{P}(X_i = c_\ell) \prod_{j \in N_i^I} \mathbb{P}(Y_{ji} = y_{ji} | X_i = c_\ell).$$

Recalling that $Y_{ij}, (i,j) \in E_S$ are identically distributed, we aggregate all the agents in N_i^I that give score r_h to agent i:

$$\mathbb{P}(\boldsymbol{Y}_{N_i^I} = \boldsymbol{y}_{N_i^I}) = \sum_{\ell=1}^{C} \mathbb{P}(X_i = c_\ell) \prod_{h=1}^{R} \mathbb{P}(Y_{j_0 i} = r_h | X_i = c_\ell)^{n_i^{(h)}},$$

and marginalize with respect to X_{j_0}, thus obtaining

$$\mathbb{P}(\boldsymbol{Y}_{N_i^I} = \boldsymbol{y}_{N_i^I}) = \sum_{\ell=1}^{C} \mathbb{P}(X_i = c_\ell) \\ \times \prod_{h=1}^{R} \Big(\sum_{m=1}^{C} \mathbb{P}(Y_{j_0 i} = r_h, X_{j_0} = c_m | X_i = c_\ell) \Big)^{n_i^{(h)}}.$$

From the structure of the score Bayesian network the following factorization is obtained:

$$\mathbb{P}(\boldsymbol{Y}_{N_i^I} = \boldsymbol{y}_{N_i^I}) = \sum_{\ell=1}^{C} \mathbb{P}(X_i = c_\ell) \prod_{h=1}^{R} \Big(\sum_{m=1}^{C} \mathbb{P}(X_{j_0} = c_m) \\ \times \mathbb{P}(Y_{j_0 i} = r_h | X_{j_0} = c_m, X_i = c_\ell) \Big)^{n_i^{(h)}}.$$

Then, the node-based relaxed likelihood L_{NR} is the product over $i = 1, \dots, N$ of the factors above, so that

$$L_{NR}(\boldsymbol{\theta}, \boldsymbol{\gamma}) = \prod_{i=1}^{N} \sum_{\ell=1}^{C} p_\ell(\boldsymbol{\gamma}) \prod_{h=1}^{R} \Big(\sum_{m=1}^{C} p_{h|m,\ell}(\boldsymbol{\theta}) p_m(\boldsymbol{\gamma}) \Big)^{n_i^{(h)}},$$

where we have used the shorthand notation introduced in (2) and (1).

Applying the logarithm to the expression of $L_{NR}(\boldsymbol{\theta}, \boldsymbol{\gamma})$ derived above, it follows immediately that $\log(L_{NR}(\boldsymbol{\theta}, \boldsymbol{\gamma})) = \sum_{i=1}^{N} g(\boldsymbol{\theta}, \boldsymbol{\gamma}; \boldsymbol{n}_i)$, with each $g(\boldsymbol{\theta}, \boldsymbol{\gamma}; \boldsymbol{n}_i)$ as in (12). Finally, since the logarithm is increasing, the maximum argument is invariant under this transformation, thus concluding the proof.

D. *Proof of Proposition III.5*

Before proving Proposition III.5 we give a useful lemma:

Lemma A.4. *The fully relaxed likelihood can be written as*

$$L_{FR}(\boldsymbol{\theta}, \boldsymbol{\gamma}) = \prod_{h=1}^{R} \Big(\sum_{\ell,m=1}^{C} p_{h|\ell,m}(\boldsymbol{\theta}) p_\ell(\boldsymbol{\gamma}) p_m(\boldsymbol{\gamma}) \Big)^{n^{(h)}},$$

where $n^{(h)} := |\{(i,j) \in E_S : y_{ij} = r_h\}|$.

Proof. In what follows, we omit the dependency on $\boldsymbol{\theta}$ and $\boldsymbol{\gamma}$. We start focusing our attention on the product $\prod_{(i,j)\in E_S} \mathbb{P}(Y_{ij} = y_{ij})$, which gives the fully relaxed likelihood. Recalling that $Y_{ij}, (i,j) \in E_S$, are identically distributed, and aggregating the edges (i,j) for which $y_{ij} = r_h$, we obtain

$$\prod_{(i,j)\in E_S} \mathbb{P}(Y_{ij} = y_{ij}) = \prod_{h=1}^{R} \mathbb{P}(Y_{i_0 j_0} = r_h)^{n^{(h)}},$$

with (i_0, j_0) being an arbitrary edge in E_S. Then, marginalizing with respect to X_{i_0} and X_{j_0}, it follows that

$$\prod_{(i,j)\in E_S} \mathbb{P}(Y_{ij} = y_{ij}) \\ = \prod_{h=1}^{R} \Big(\sum_{\ell,m=1}^{C} \mathbb{P}(Y_{i_0 j_0} = r_h, X_{i_0} = c_\ell, X_{j_0} = c_m) \Big)^{n^{(h)}}.$$

Finally, exploiting again the structure of the Bayesian network, we have that

$$\prod_{(i,j)\in E_S} \mathbb{P}(Y_{ij} = y_{ij}) = \prod_{h=1}^{R} \Big(\sum_{\ell,m=1}^{C} \mathbb{P}(X_{i_0} = c_\ell) \mathbb{P}(X_{j_0} = c_m) \\ \times \mathbb{P}(Y_{i_0 j_0} = r_h | X_{i_0} = c_\ell, X_{j_0} = c_m) \Big)^{n^{(h)}},$$

so that the proof follows. □

Now we are ready to prove Proposition III.5. Since the logarithm is a monotone transformation, it follows straight that

$$(\hat{\boldsymbol{\theta}}_{\text{FR}}, \hat{\boldsymbol{\gamma}}_{\text{FR}}) = \operatorname*{argmin}_{(\boldsymbol{\gamma},\boldsymbol{\theta}) \in \mathcal{S}_{\boldsymbol{\gamma}} \times \mathcal{S}_{\boldsymbol{\theta}}} -\frac{1}{n} \log(L_{FR}(\boldsymbol{\theta}, \boldsymbol{\gamma})). \tag{23}$$

Using Lemma A.4, we have that:

$$-\frac{1}{n} \log(L_{FR}(\boldsymbol{\theta}, \boldsymbol{\gamma})) = \\ -\sum_{h=1}^{R} \frac{n^{(h)}}{n} \log \Big(\sum_{\ell,m=1}^{C} p_{h|\ell,m}(\boldsymbol{\theta}) p_\ell(\boldsymbol{\gamma}) p_m(\boldsymbol{\gamma}) \Big), \tag{24}$$

so that it is easy to show that

$$n^{(h)} = \sum_{i=1}^{N} n_i^{(h)}. \tag{25}$$

Plugging together (23), (24), and (25), the proof follows.

E. Proof of Theorem III.7

Before proving Theorem III.7, we give two Lemmas.

Lemma A.5. *Let Assumption II.1 holds, and consider $i \in \{1,\dots,N\}$. Then there exist $C > 0$, $d \in [0,1)$ and $T > 0$, such that,*

$$\|\boldsymbol{\phi} - \boldsymbol{\phi}_i(t)\| \leq d^t C, \tag{26}$$

for all $t \geq T$.

Proof. The proof follows the same steps as in [30], details are omitted for the sake of conciseness. □

Lemma A.6. *Let A_t, B_t, C_t be three sequences such that B_t is nonnegative for all t. Assume that*

$$A_{t+1} \leq A_t - B_t + C_t,$$

and that the series $\sum_{t=0}^{\infty} C_t$ converges. Then either $A_t \to -\infty$ or else A_t converges to a finite value and $\sum_{t=0}^{\infty} B_t < \infty$.

Proof. For the proof see Lemma 1 in [34]. □

We now prove Theorem III.7. For notational purposes, we define:

$$\begin{aligned} z_i^t &:= (\hat{\boldsymbol{\theta}}_i(t), \hat{\boldsymbol{\gamma}}_i(t)), \\ \overline{z}_i^t &:= [z_i^t - \alpha \boldsymbol{\phi}_i(t)^\top \nabla \boldsymbol{g}(z_i^t)]^+, \\ g_\phi(\boldsymbol{\theta}, \boldsymbol{\gamma}) &:= \boldsymbol{\phi}^\top \boldsymbol{g}(\boldsymbol{\theta}, \boldsymbol{\gamma}). \end{aligned} \tag{27}$$

Let $\hat{z}_i$ be a limit-point of $\{z_i^t\}_{t\geq 0}$; taking into account that the stepsize is constant, our strategy is to prove that $\hat{z}_i$ is a stationary point of g_ϕ over $\mathcal{S}_{\boldsymbol{\theta}} \times \mathcal{S}_{\boldsymbol{\gamma}}$ by showing that $\hat{z}_i$ is a fixed point of the projected gradient method related to the objective function g_ϕ and to the feasible set $\mathcal{S}_{\boldsymbol{\theta}} \times \mathcal{S}_{\boldsymbol{\gamma}}$, i.e.:

$$\hat{z}_i = [\hat{z}_i - \alpha \nabla g_\phi(\hat{z}_i)]^+. \tag{28}$$

Clearly $z_i^{t+1} = \overline{z}_i^t$, thus we have:

$$g_\phi(z_i^{t+1}) - g_\phi(z_i^t) = g_\phi(\overline{z}_i^t) - g_\phi(z_i^t), \tag{29}$$

moreover, using the fact that ∇g_ϕ is Lipschitz continuous with constant $L > 0$, it follows that:

$$g_\phi(\overline{z}_i^t) - g_\phi(z_i^t) \leq \nabla g_\phi(z_i^t)^\top (\overline{z}_i^t - z_i^t) + \frac{L}{2}\|\overline{z}_i^t - z_i^t\|^2. \tag{30}$$

Simple calculations show that

$$\nabla g_\phi(z_i^t) = \boldsymbol{\phi}_i(t)^\top \nabla \boldsymbol{g}(z_i^t) + (\boldsymbol{\phi} - \boldsymbol{\phi}_i(t))^\top \nabla \boldsymbol{g}(z_i^t), \tag{31}$$

where $\nabla \boldsymbol{g}(z_i^t)$ is the Jacobian of $\boldsymbol{g}$ at z_i^t. Putting together equations (29), (30), and (31) we obtain:

$$\begin{aligned} g_\phi(z_i^{t+1}) - g_\phi(z_i^t) &\leq \boldsymbol{\phi}_i(t)^\top \nabla \boldsymbol{g}(z_i^t)(\overline{z}_i^t - z_i^t) \\ &+ \frac{L}{2}\|\overline{z}_i^t - z_i^t\|^2 + (\boldsymbol{\phi} - \boldsymbol{\phi}_i(t))^\top \nabla \boldsymbol{g}(z_i^t)(\overline{z}_i^t - z_i^t). \end{aligned} \tag{32}$$

We know that $\overline{z}_i^t$ is the projection of $z_i^t - \alpha \boldsymbol{\phi}_i(t)^\top \nabla \boldsymbol{g}(z_i^t)$ onto the set $\mathcal{S}_{\boldsymbol{\gamma}} \times \mathcal{S}_{\boldsymbol{\theta}}$, therefore:

$$(z_i^t - \alpha \boldsymbol{\phi}_i(t)^\top \nabla \boldsymbol{g}(z_i^t) - \overline{z}_i^t)^\top (z - \overline{z}_i^t) \leq 0, \qquad z \in \mathcal{S}_{\boldsymbol{\gamma}} \times \mathcal{S}_{\boldsymbol{\theta}}.$$

In particular, for $z = z_i^t$, remembering that $\alpha > 0$, we have:

$$\boldsymbol{\phi}_i(t)^\top \nabla \boldsymbol{g}(z_i^t)(\overline{z}_i^t - z_i^t) \leq -\frac{1}{\alpha}\|\overline{z}_i^t - z_i^t\|^2. \tag{33}$$

From the boundedness of $\nabla \boldsymbol{g}$ and from equation (26), ∇g_ϕ is bounded. Moreover, from Lemma A.5, $\boldsymbol{\phi}_i(t)$ converges exponentially fast to $\boldsymbol{\phi}$. Thus, since by assumption the feasible set is bounded, there exist $C > 0$ and $d \in [0,1)$ such that

$$(\boldsymbol{\phi} - \boldsymbol{\phi}_i(t))^\top \nabla \boldsymbol{g}(z_i^t)(\overline{z}_i^t - z_i^t) \leq d^t C. \tag{34}$$

Now, let G be the following auxiliary function:

$$G(t) := g_\phi(z_i^{t+1}) - g_\phi(z_i^t) - d^t C,$$

combining equation (34) with (32) and (33), and recalling that $0 < \alpha < \frac{2}{L}$, we have:

$$G(t) \leq \left(\frac{L}{2} - \frac{1}{\alpha}\right)\|\overline{z}_i^t - z_i^t\|^2 \leq 0. \tag{35}$$

In particular, we have:

$$g_\phi(z_i^{t+1}) \leq g_\phi(z_i^t) + d^t C,$$

so that, applying Lemma A.6 with the sequences $A_t = g_\phi(z_i^t)$, $B_t = 0$ and $C_t = d^t C$, and recalling that the feasible set is bounded, $g_\phi(z_i^t)$ converges to a finite value. Therefore:

$$\lim_{t\to\infty} G(t) = 0. \tag{36}$$

We have supposed that $\hat{z}_i$ is a limit point of $\{z_i^t\}_{t\geq 0}$, thus:

$$\lim_{k\to\infty} z_i^{t_k} = \hat{z}_i. \tag{37}$$

Combining equations (35), (36) and (37), it follows that

$$\lim_{k\to\infty} \overline{z}_i^{t_k} = \hat{z}_i. \tag{38}$$

By hypothesis $\nabla \boldsymbol{g}$ is continuous; moreover, $[\cdot]^+$ is non-expansive, thus continuous. Using these facts, from Lemma A.5 and equations (27), (37), we can conclude:

$$\lim_{k\to\infty} \overline{z}_i^{t_k} = [\hat{z}_i - \alpha \nabla g_\phi(\hat{z}_i)]^+.$$

that plugged into (38) gives (28), thus concluding the proof.

REFERENCES

[1] F. Sasso, A. Coluccia, and G. Notarstefano, "Distributed learning from interactions in social networks," in *European Control Conference*, 2018.

[2] S. Barbarossa and G. Scutari, "Decentralized maximum-likelihood estimation for sensor networks composed of nonlinearly coupled dynamical systems," *IEEE Transactions on Signal Processing*, vol. 55, no. 7, pp. 3456–3470, 2007.

[3] I. D. Schizas, A. Ribeiro, and G. B. Giannakis, "Consensus in ad hoc WSNs with noisy links Part I: Distributed estimation of deterministic signals," *IEEE Transactions on Signal Processing*, vol. 56, no. 1, pp. 350–364, 2008.

[4] A. Chiuso, F. Fagnani, L. Schenato, and S. Zampieri, "Gossip algorithms for simultaneous distributed estimation and classification in sensor networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 4, pp. 691–706, 2011.

[5] A. Coluccia and G. Notarstefano, "Distributed estimation of binary event probabilities via hierarchical bayes and dual decomposition," in *52nd IEEE Conference on Decision and Control*, 2013.

[6] A. Coluccia and G. Notarstefano, "A hierarchical bayes approach for distributed binary classification in cyber-physical and social networks," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 7406–7411, 2014.

[7] A. Coluccia and G. Notarstefano, "A bayesian framework for distributed estimation of arrival rates in asynchronous networks," *IEEE Transactions on Signal Processing*, vol. 64, no. 15, pp. 3984–3996, 2016.

[8] F. Fagnani, S. M. Fosson, and C. Ravazzi, "A distributed classification/estimation algorithm for sensor networks," *SIAM Journal on Control and Optimization*, vol. 52, no. 1, pp. 189–218, 2014.

[9] E. Montijano, S. Martínez, and C. Sagues, “Distributed robust consensus using ransac and dynamic opinions,” *IEEE Transactions on Control Systems Technology*, vol. 23, no. 1, pp. 150–163, 2015.
[10] S. Dandach, R. Carli, and F. Bullo, “Accuracy and decision time for sequential decision aggregation,” *Proceedings of the IEEE*, vol. 100, no. 3, pp. 687–712, 2012.
[11] A. Jadbabaie, P. Molavi, A. Sandroni, and A. Tahbaz-Salehi, “Non-bayesian social learning,” *Games and Economic Behavior*, vol. 76, no. 1, pp. 210–225, 2012.
[12] S. Shahrampour and A. Jadbabaie, “Exponentially fast parameter estimation in networks using distributed dual averaging,” in *52nd IEEE Conference on Decision and Control*, 2013, pp. 6196–6201.
[13] A. Lalitha, T. Javidi, and A. Sarwate, “Social learning and distributed hypothesis testing,” *IEEE Transactions on Information Theory*, 2018.
[14] A. Nedić, A. Olshevsky, and C. A. Uribe, “A tutorial on distributed (non-bayesian) learning: Problem, algorithms and results,” in *55th IEEE Conference on Decision and Control*, 2016, pp. 6795–6801.
[15] S. Shahrampour, A. Rakhlin, and A. Jadbabaie, “Distributed detection: Finite-time analysis and impact of network topology,” *IEEE Transactions on Automatic Control*, vol. 61, no. 11, pp. 3256–3268, 2016.
[16] P. Molavi, A. Tahbaz-Salehi, and A. Jadbabaie, “Foundations of non-bayesian social learning,” *report*, 2016.
[17] L. Su and N. H. Vaidya, “Non-bayesian learning in the presence of byzantine agents,” in *International Symposium on Distributed Computing*, 2016, pp. 414–427.
[18] A. Nedic, A. Olshevsky, and C. A. Uribe, “Fast convergence rates for distributed non-bayesian learning,” *IEEE Transactions on Automatic Control*, 2017.
[19] A. Mirtabatabaei and F. Bullo, “Opinion dynamics in heterogeneous networks: convergence conjectures and theorems,” *SIAM Journal on Control and Optimization*, vol. 50, no. 5, pp. 2763–2785, 2012.
[20] A. Mirtabatabaei, P. Jia, N. E. Friedkin, and F. Bullo, “On the reflected appraisals dynamics of influence networks with stubborn agents,” in *2014 American Control Conference*, 2014, pp. 3978–3983.
[21] N. E. Friedkin, A. V. Proskurnikov, R. Tempo, and S. E. Parsegov, “Network science on belief system dynamics under logic constraints,” *Science*, vol. 354, no. 6310, pp. 321–326, 2016.
[22] P. Frasca, H. Ishii, C. Ravazzi, and R. Tempo, “Distributed randomized algorithms for opinion formation, centrality computation and power systems estimation: A tutorial overview,” *European Journal of Control*, 2015.
[23] W. Li, F. Bassi, L. Galluccio, and M. Kieffer, “Self-rating in a community of peers,” in *55th IEEE Conference on Decision and Control*, 2016, pp. 5888–5893.
[24] D. Koller and N. Friedman, *Probabilistic graphical models: principles and techniques*. MIT press, 2009.
[25] F. Preparata, G. Metze, and R. Chien, “On the connection assignment problem of diagnosable systems,” *IEEE Transactions on Computers*, vol. EC-16, pp. 848–854, 1967.
[26] M. A. Fligner and J. S. Verducci, “Distance based ranking models,” *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 359–369, 1986.
[27] J. I. Marden, *Analyzing and modeling rank data*. CRC Press, 1996.
[28] C. L. Mallows, “Non-null rankings models,” *Biometrika*, vol. 44, pp. 114–130, 1957.
[29] R. Carli, G. Notarstefano, L. Schenato, and D. Varagnolo, “Analysis of newton-raphson consensus for multi-agent convex optimization under asynchronous and lossy communications,” in *54th IEEE Conference on Decision and Control*, 2015, pp. 418–424.
[30] A. Nedić and A. Olshevsky, “Distributed optimization over time-varying directed graphs,” *IEEE Transactions on Automatic Control*, vol. 60, no. 3, pp. 601–615, 2015.
[31] P. Di Lorenzo and G. Scutari, “Next: In-network nonconvex optimization,” *IEEE Transactions on Signal and Information Processing over Networks*, vol. 2, no. 2, pp. 120–136, 2016.
[32] F. Bénézit, V. Blondel, P. Thiran, J. Tsitsiklis, and M. Vetterli, “Weighted gossip: Distributed averaging using non-doubly stochastic matrices,” in *IEEE International Symposium on Information Theory Proceedings*, 2010, pp. 1753–1757.
[33] J. Pearl and R. Dechter, “Identifying independencies in causal graphs with feedback,” in *Proceedings of the Twelfth international conference on Uncertainty in artificial intelligence*, 1996, pp. 420–426.
[34] D. P. Bertsekas and J. N. Tsitsiklis, “Gradient convergence in gradient methods with errors,” *SIAM Journal on Optimization*, vol. 10, no. 3, pp. 627–642, 2000.

Francesco Sasso is a Ph.D. student in Engineering of Complex Systems at the Università del Salento, Italy, working on the ERC-granted project OPT4SMART. He received the Laurea degree “summa cum laude” in Mathematics (curriculum “PDE and applications”) from the Università degli Studi di Bari, Italy, in 2015. His research interests include distributed optimization, estimation theory and differential equations.

Angelo Coluccia received the Eng. degree in Telecommunication Engineering (summa cum laude) in 2007 and the PhD degree in Information Engineering in 2011, both from the University of Salento (Lecce, Italy). He is currently Assistant Professor (tenure-track for the rank of Associate) at the Dipartimento di Ingegneria dell’Innovazione, University of Salento, where he teaches the course of Telecommunication Systems. Since 2008 he has been engaged in European research projects on different topics, including small-drone detection, tracking, and data fusion. He has been research fellow at Forschungszentrum Telekommunikation Wien (Vienna, Austria), working on mobile Internet traffic modeling and anomaly detection. He has held a visiting position at the Department of Electronics, Optronics, and Signals (DEOS) of the Institut Supérieur de l’Aéronautique et de l’Espace (ISAE-Supaero, Toulouse, France), in the Signal, Communication, Antennas and Navigation (SCAN) Group. His research interests are in the area of multi-sensor and multi-agent statistical signal processing for detection, estimation, learning, and localization problems. Relevant application fields are radar, wireless networks (including 5G), intelligent cyber-physical systems, and emerging network contexts (including smart devices and social networks). He is Senior Member of IEEE, Member of CNIT (National Inter-University Consortium for Telecommunications), and Member of the Special Area Team in Signal Processing for Multisensor Systems of EURASIP (European Association for Signal Processing).

Giuseppe Notarstefano is a Professor in the Department of Electrical, Electronic, and Information Engineering G. Marconi at Alma Mater Studiorum Università di Bologna. He was Associate Professor (from June ‘16 to June ‘18) and previously Assistant Professor, Ricercatore, (from February ‘07) at the Università del Salento, Lecce, Italy. He received the Laurea degree summa cum laude in Electronics Engineering from the Università di Pisa in 2003 and the Ph.D. degree in Automation and Operation Research from the Università di Padova in 2007. He has been visiting scholar at the University of Stuttgart, University of California Santa Barbara and University of Colorado Boulder. His research interests include distributed optimization, cooperative control in complex networks, applied nonlinear optimal control, and trajectory optimization and maneuvering of aerial and car vehicles. He serves as an Associate Editor for IEEE Transactions on Automatic Control, IEEE Transactions on Control Systems Technology and IEEE Control Systems Letters. He is also part of the Conference Editorial Board of IEEE Control Systems Society and EUCA. He is recipient of an ERC Starting Grant 2014.