Decentralized Online Nonparametric Learning

Alec Koppel[§], Santiago Paternain[⋆], Cédric Richard[†] and Alejandro Ribeiro[⋆]

Abstract—**We consider decentralized online supervised learning where estimators are chosen from a reproducing kernel Hilbert space (RKHS). Here a multi-agent network aims to learn nonlinear statistical models that are optimal in terms of a global convex functional that aggregates data across the network, while only having access to locally observed streaming data. We address this problem by allowing each agent to learn a local copy of the global regression function while enforcing consensus constraints. We use a penalized variant of functional stochastic gradient descent operating simultaneously with low-dimensional subspace projections. The resulting algorithm allows each individual agent to learn, based upon its locally observed data stream and message passing with its neighbors, a function that is provably close to globally optimal and satisfies the consensus constraints. Moreover, the complexity of the learned regression functions is guaranteed to be finite. We then validate this approach on the Brodatz textures dataset for the case of decentralized online multi-class kernel logistic regression.**

I. Introduction

We focus on decentralized online statistical learning problems, in which agents aim to make inferences as well as one which has access to all data at a centralized location in advance. Instead of assuming agents seek a common parameter vector $\mathbf{w} \in \mathbb{R}^p$, we focus on the case where agents seek to learn a common *decision function* $f(\mathbf{x})$ belonging to a reproducing kernel Hilbert space (RKHS). Such functions represent, e.g., nonlinear statistical models [3] or trajectories in a continuous space [4].

Optimization tools for multi-agent online learning have thus far been restricted to the case where each agent learns a linear statistical model [5] or a task-driven dictionary [6], which exclude state of the art nonlinear interpolators: kernel methods [7], [8] and neural networks [9], [10]. We note that instabilities associated with non-convexity which are only a minor issue in centralized settings [11], but become both theoretically and empirically difficult to overcome in constrained settings [6], and therefore efforts to extend neural network learning to multi-agent online learning must overcome duality gap issues associated to constrained non-convex settings. Instead, we focus on extending kernel methods to decentralized online settings, motivated both by its advantageous empirical performance, as well as the theoretical benefits of convexity. This stochastic convex problem, however, is defined over an infinite space, and therefore one must both solve the optimization problem and sure it is optimally sparse. Doing so in multi-agent settings, which underlie Internet of Things [12], [13] and multi-robot [14], [15] applications, is our goal.

To understand our approach, consider centralized vector-valued stochastic convex programming, which has been classically solved with stochastic gradient descent (SGD) [16]. SGD involves descending along the negative of the stochastic gradient rather than the true gradient to avoid the fact that computing the gradient of the average objective has complexity comparable to the training sample size, which could be infinite. In contrast, a stochastic program defined over a function space is an intractable variational inference problem in general, but when the function space is a RKHS [17], the Representer Theorem allows us to reduce an infinite space into a parameterization of weights and data samples [18]. Unfortunately, the resulting feasible set has complexity comparable to sample size N (intractable for $N \to \infty$ [19]). Efforts to mitigate this complexity explosion have been developed that combine functional extensions of stochastic gradient method (FSGD) with compressions of the function parameterization [20]–[24]. Mostly, such methods compress the function representation independent of the iterative sequence to which they are applied. In contrast, a method was recently proposed that combines greedily constructed [25] sparse subspace projections with functional SGD, and tailors the compression to preserve optimality properties of FSGD [26].

Here we extend [26] to multi-agent settings (Section II). Multiple distributed optimization tools may be used to develop such an extension; however, the Representer Theorem [18] has not been established for stochastic saddle point problems in RKHSs. Thus, we adopt an approximate primal-only approach via penalty methods [27], [28], which in decentralized optimization is called distributed gradient descent (DGD) (Section III). Using functional stochastic extensions of DGD, together with the greedy projections designed in [26], we develop a method such that each agent, through its local data stream and neighborhood message passing, learns a memory-efficient approximation to the *globally optimal* function almost surely (Section IV), which contrasts with other nonlinear interpolation techniques such as dictionary learning [6], [11], [29] or neural networks [10] which exhibit instability due to non-convexity. In Section V, we apply our method to decentralized online multi-class kernel logistic regression on the Brodatz textures [30], and observe stable learning, memory efficiency, and competitive error rates. Compared to [2], we have corrected our main convergence result, establish that consensus is attained in the full RKHS, rather than only in mean square, and further validate the proposed method on a practically challenging visual identification task.

II. Online Learning with Kernels

Consider the problem of distributed expected risk minimization, where the goal is to learn a regressor that minimizes a loss function quantifying the merit of a statistical model averaged

Thanks to NSF CCF-0952867, ONR N00014-12-1-0997, ASEE SMART. Additional derivations are given in [1]. Related work appeared as [2].

[§]Computational and Information Sciences Directorate, U.S. Army Research Laboratory [⋆]Department of Electrical and Systems Engineering, University of Pennsylvania [†]Laboratory Lagrange, Observatory of the French Riviera, University of Nice Sophia-Antipolis

Algorithm 1 Greedy Projected Penalty Method

```
Require: $\{\mathbf{x}_t, \mathbf{y}_t, \eta, \epsilon\}_{t=0,1,2,\ldots}$
  initialize $f_{i,0}(\cdot) = 0, \mathbf{D}_{i,0} = [], \mathbf{w}_0 = []$, i.e. initial dictionary, coefficients are empty for each $i \in \mathcal{V}$
  for $t = 0, 1, 2, \ldots$ do
    loop in parallel for agent $i \in \mathcal{V}$
      Observe local training example realization $(\mathbf{x}_{i,t}, y_{i,t})$
      Send obs. $\mathbf{x}_{i,t}$ to nodes $j \in n_i$, receive scalar $f_{j,t}(\mathbf{x}_{i,t})$
      Receive obs. $\mathbf{x}_{j,t}$ from nodes $j \in n_i$, send $f_{i,t}(\mathbf{x}_{j,t})$
      Compute unconstrained stochastic grad. step [cf. (8)]
      $\tilde{f}_{i,t+1}(\cdot) = (1-\eta\lambda) f_{i,t} - \eta \nabla_{f_i} \hat{\psi}_{i,c}(f_i(\mathbf{x}_{i,t}), \mathbf{y}_{i,t})$ .
      Update params: $\tilde{\mathbf{D}}_{i,t+1} = [\mathbf{D}_{i,t}, \ \mathbf{x}_{i,t}]$, $\tilde{\mathbf{w}}_{i,t+1}$ [cf. (10)]
      Greedily compress function using matching pursuit
      $(f_{i,t+1}, \mathbf{D}_{i,t+1}, \mathbf{w}_{i,t+1}) = \textbf{KOMP}(\tilde{f}_{i,t+1}, \tilde{\mathbf{D}}_{i,t+1}, \tilde{\mathbf{w}}_{i,t+1}, \epsilon)$
    end loop
  end for
```

over a data set scattered across an interconnected network that represents, for instance, robotic teams [6], communication systems [31], or sensor networks [32]. To do so, we define a symmetric, connected, and directed network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $|\mathcal{V}| = V$ nodes and $|\mathcal{E}| = E$ edges and denote as $n_i := \{j : (i,j) \in \mathcal{E}\}$ the neighborhood of agent i. Each agent $i \in \mathcal{V}$ observes a local data sequence as realizations $(\mathbf{x}_{i,n}, y_{i,n})$ from random pair $(\mathbf{x}_i, y_i) \in \mathcal{X} \times \mathcal{Y} \subset \mathbb{R}^p \times \mathbb{R}$ and seeks to learn a common globally optimal regression function f from the class function $\mathcal{H}$. This setting may be mathematically captured by associating to each node i a convex loss functional $\ell_i : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ that quantifies the merit of the estimator $\tilde{f}(\mathbf{x}_i)$ evaluated at feature vector $\mathbf{x}_i$. This loss averaged over all possible $\mathbf{x}_i$ defines the statistical loss $L(\tilde{f}) := \sum_{i \in \mathcal{V}} \mathbb{E}_{\mathbf{x}_i, y_i}\Big[\ell_i(\tilde{f}(\mathbf{x}_i), y_i)\Big]$, which we combine with a Tikhonov regularizer to construct the regularized loss $R(\tilde{f}) := \text{argmin}_{\tilde{f} \in \mathcal{H}} L(\tilde{f}) + (\lambda/2)\|\tilde{f}\|^2_{\mathcal{H}}$ [33], [34]. Then the globally optimal regression function $\tilde{f}^*$ is defined as

$$\tilde{f}^* = \underset{\tilde{f} \in \mathcal{H}}{\text{argmin}} \sum_{i \in \mathcal{V}} \left(\mathbb{E}_{\mathbf{x}_i, y_i}\Big[\ell_i(\tilde{f}(\mathbf{x}_i), y_i)\Big] + \frac{\lambda}{2}\|\tilde{f}\|^2_{\mathcal{H}} \right) . \quad (1)$$

Observe that this global loss is a network-wide average (scaled by V) of all local losses, and therefore the minimizers of (1) and a centralized agent with access to all data coincide when $(\mathbf{x}_i, y_i)$ have a common joint distribution for each i. However, in multi-agent optimization, agents select a regression function f with only local data, which yields different decision functions f_i^* that are not as good as one selected with data aggregated across the network. To overcome this limitation, we allow message passing between agents and consider a setting where agents seek to select decisions as good as a centralized meta-agent. Thus, constrain the regression functions among neighbors to be equal $f_i = f_j$, $(i,j) \in \mathcal{E}$, yielding

$$\begin{aligned} f^* = \ & \underset{\{f_i\} \subset \mathcal{H}}{\text{argmin}} && \sum_{i \in \mathcal{V}} \left(\mathbb{E}_{\mathbf{x}_i, y_i}\Big[\ell_i(f_i(\mathbf{x}), y_i)\Big] + \frac{\lambda}{2}\|f_i\|^2_{\mathcal{H}} \right) \\ & \text{such that} && f_i = f_j \ , (i,j) \in \mathcal{E} \ . \end{aligned} \quad (2)$$

Define the product Hilbert space $\mathcal{H}^V$ of functions aggregated over the network, whose elements are stacked functions $f(\cdot) = [f_1(\cdot); \cdots ; f_V(\cdot)]$. Further define stacked random vectors $\mathbf{x} = [\mathbf{x}_1; \cdots ; \mathbf{x}_V] \in \mathbb{R}^{Vp}$ and $\mathbf{y} = [y_1; \cdots y_V] \in \mathbb{R}^V$ that represents V labels or physical measurements, for instance. We seek to solve (2) when nodes do not know the distribution of the random pair $(\mathbf{x}_i, y_i)$ but sequentially observe local independent training examples $(\mathbf{x}_{i,n}, y_{i,n})$, but are allowed to do local message passing with one another. Next, we discuss details of function space $\mathcal{H}^V$ that make (2) tractable.

A. Reproducing Kernel Hilbert Spaces

The problem in (2) is intractable in general, since it defines a variational inference problem integrated over the unknown joint distribution $\mathbb{P}(\mathbf{x}, y)$. However, when $\mathcal{H}$ is equipped with a *reproducing kernel* $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ (see [8], [35]), a functional problem of the form (1) may be reduced to a parametric form via the Representer Theorem [19], [36]. Thus, we restrict the Hilbert space in (2) to be one equipped with a kernel κ that satisfies for all functions $\tilde{f} : \mathcal{X} \to \mathbb{R}$ in $\mathcal{H}$ and for all $\mathbf{x}_i \in \mathcal{X}$:

$$(i) \ \langle \tilde{f}, \kappa(\mathbf{x}_i, \cdot)) \rangle_{\mathcal{H}} = \tilde{f}(\mathbf{x}_i) \qquad (ii) \ \mathcal{H} = \overline{\text{span}\{\kappa(\mathbf{x}_i, \cdot)\}}. \quad (3)$$

Here $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ denotes the Hilbert inner product for $\mathcal{H}$. Further assume that κ is positive semidefinite, i.e. $\kappa(\mathbf{x}_i, \mathbf{x}_i') \geq 0$ for all $\mathbf{x}_i, \mathbf{x}_i' \in \mathcal{X}$. For kernelized regularized empirical risk minimization, the Representer Theorem [37] states that optimal $\tilde{f}$ in the function class $\mathcal{H}$ may be written in terms of kernel evaluations only of the training set

$$\tilde{f}(\mathbf{x}_i) = \sum_{n=1}^{N} w_{i,n} \kappa(\mathbf{x}_{i,n}, \mathbf{x}_i) \ , \quad (4)$$

where $\mathbf{w}_i = [w_{i,1}, \cdots, w_{i,N}]^T \in \mathbb{R}^N$ denotes a set of weights. The upper index N in (4) is referred to as the model order, and for ERM the model order and training sample size are equal. By exploiting the Representer Theorem, we transform an infinite dimensional optimization problem in $\mathcal{H}^V$ into a finite NV-dimensional parametric problem. Thus, the RKHS provides a principled framework to solve nonparametric regression problems as a search over $\mathbb{R}^{VN}$ for a set of coefficients. However, when training examples $(\mathbf{x}_{i,n}, y_{i,n})$ become sequentially available or their total number N is not finite, the complexity of representing a function $\tilde{f}$ in (4) approaches infinity as well, and thus requires an intractable amount of memory. Thus, our goal is to solve (2) in an approximate manner such that each f_i admits a finite representation near f_i^*, while satisfying the consensus constraints $f_i = f_j$ for $(i,j) \in \mathcal{E}$.

III. Decentralized Collaborative Learning

We now develop an online decentralized iterative solution to (2) when the functions $\{f_i\}_{i \in \mathcal{V}}$ are elements of a RKHS, as detailed in Section II-A. To exploit the properties of this function space, we require the applicability of the Representer Theorem [cf. (4)], but this result holds for any regularized minimization problem with a convex functional. Thus, we may address the consensus constraint $f_i = f_j$ $,(i,j) \in \mathcal{E}$ in (2) by enforcing approximate consensus on estimates $f_i(\mathbf{x}_i) = f_j(\mathbf{x}_i)$ in expectation. Thus, we introduce the penalty function

$$\psi_c(f)=\sum_{i\in\mathcal{V}}\Big(R_i(f_i)+\frac{c}{2}\sum_{j\in n_i}\mathbb{E}_{\mathbf{x}_i}\left\{[f_i(\mathbf{x}_i)-f_j(\mathbf{x}_i)]^2\right\}\Big), \tag{5}$$

where $R_i(f_i) := \mathbb{E}_{\mathbf{x}_i,y_i}\big[\ell_i(f_i\big(\mathbf{x}_i), y_i\big)\big] + (\lambda/2)\|f_i\|^2_{\mathcal{H}}$. Further define $f_c^* = \operatorname{argmin}_{f\in\mathcal{H}^V}\psi_c(f)$ and the local penalty as

$$\psi_{i,c}(f_i)= R_i(f_i)+\frac{c}{2}\sum_{j\in n_i}\mathbb{E}_{\mathbf{x}_i}\left\{[f_i(\mathbf{x}_i)-f_j(\mathbf{x}_i)]^2\right\}. \tag{6}$$

Observe from (5) - (6) that $\psi_c(f) = \sum_i \psi_{i,c}(f_i)$.

A. Functional Stochastic Gradient Method

The data distribution $\mathbb{P}(\mathbf{x}, \mathbf{y})$ is unknown, so minimizing $\psi_c(f)$ directly via variational inference is not possible. Rather than postulate a distribution for $(\mathbf{x}, \mathbf{y})$, we only require sequentially available independent and identically distributed samples $(\mathbf{x}_t, \mathbf{y}_t)$ from their joint density. Then, we address (5) using stochastic methods. Thus, compute $\nabla_f \hat{\psi}_{i,c}(f(\mathbf{x}_t), \mathbf{y}_t)$, the functional stochastic gradient (5), as in [22]

$$\begin{aligned}\nabla_f\hat{\psi}_{i,c}(f(\mathbf{x}_t),\mathbf{y}_t)= \ell_i'(f_i(\mathbf{x}_{i,t}),y_{i,t})\kappa(\mathbf{x}_{i,t},\cdot)+\lambda f_i \\ +c\sum_{j\in n_i}(f_i(\mathbf{x}_{i,t})-f_j(\mathbf{x}_{i,t}))\kappa(\mathbf{x}_{i,t},\cdot).\end{aligned} \tag{7}$$

Now we may define the distributed stochastic gradient method for the kernelized λ-regularized multi-agent problem in (2) as

$$f_{i,t+1} = f_{i,t} - \eta\nabla_f\hat{\psi}_{i,c}(f(\mathbf{x}_t), \mathbf{y}_t)\ , \tag{8}$$

where $\eta > 0$ is an algorithm step-size. We further require that, given $\lambda > 0$, the step-size satisfies $\eta < 1/\lambda$ and the global sequence is initialized as $f_0 = 0 \in \mathcal{H}^V$. With this initialization, the Representer Theorem (c.f. (4)) implies that, at time t, the function $f_{i,t}$ admits an expansion in terms of feature vectors $\mathbf{x}_{i,t}$ observed thus far as

$$f_{i,t}(\mathbf{x}) = \sum_{n=1}^{t-1} w_{i,n}\kappa(\mathbf{x}_{i,n}, \mathbf{x}) = \mathbf{w}_{i,t}^T\boldsymbol{\kappa}_{\mathbf{X}_{i,t}}(\mathbf{x})\ . \tag{9}$$

On the right-hand side of (9) we have introduced the notation $\mathbf{X}_{i,t} = [\mathbf{x}_{i,1},\ldots,\mathbf{x}_{i,t-1}] \in \mathbb{R}^{p\times(t-1)}$, $\boldsymbol{\kappa}_{\mathbf{X}_{i,t}}(\cdot) = [\kappa(\mathbf{x}_{i,1},\cdot),\ldots,\kappa(\mathbf{x}_{i,t-1},\cdot)]^T$, and $\mathbf{w}_{i,t} = [w_{i,1},\ \ldots\ w_{i,t-1}] \in \mathbb{R}^{t-1}$. Moreover, observe that the kernel expansion in (9), together with the update (8), yields the fact that performing the stochastic gradient method in $\mathcal{H}^V$ amounts to V parallel parametric updates on the dictionaries $\mathbf{X}_i$ and coefficients $\mathbf{w}_i$:

$$\mathbf{X}_{i,t+1} = [\mathbf{X}_{i,t},\ \ \mathbf{x}_{i,t}]\ , \tag{10}$$

$$[\mathbf{w}_{i,t+1}]_u=\begin{cases}(1-\eta\lambda)[\mathbf{w}_{i,t}]_u & \text{for } 0\leq u\leq t-1\\ -\eta\Big(\ell_i'(f_{i,t}(\mathbf{x}_{i,t}),y_{i,t})+c\sum_{j\in n_i}(f_{i,t}(\mathbf{x}_{i,t})-f_{j,t}(\mathbf{x}_{i,t}))\Big),\end{cases}$$

where the second case on the last line of (10) is for $u = t$. This update causes $\mathbf{X}_{i,t+1}$ to have one more column than $\mathbf{X}_{i,t}$. We define the *model order* as number of data points $M_{i,t}$ in the dictionary of agent i at time t (the number of columns of $\mathbf{X}_t$). FSGD is such that $M_{i,t} = t - 1$, and hence grows unbounded with t. Next we address this intractable memory growth such that we may execute stochastic descent through low-dimensional projections of the stochastic gradient [26].

B. Sparse Subspace Projections

Reduce the complexity noted in Section III-A, we approximate the function sequence (8) by one that is orthogonally projected onto subspaces $\mathcal{H}_{\mathbf{D}} \subseteq \mathcal{H}$ that consist only of functions that can be represented using some dictionary $\mathbf{D} = [\mathbf{d}_1,\ \ldots,\ \mathbf{d}_M] \in \mathbb{R}^{p\times M}$, i.e., $\mathcal{H}_{\mathbf{D}} = \{f\ :\ f(\cdot) = \sum_{n=1}^{M} w_n\kappa(\mathbf{d}_n,\cdot) = \mathbf{w}^T\boldsymbol{\kappa}_{\mathbf{D}}(\cdot)\} = \text{span}\{\kappa(\mathbf{d}_n,\cdot)\}_{n=1}^M$, and $\{\mathbf{d}_n\} \subset \{\mathbf{x}_u\}_{u\leq t}$. For convenience we define $[\boldsymbol{\kappa}_{\mathbf{D}}(\cdot) = \kappa(\mathbf{d}_1,\cdot)\ldots\kappa(\mathbf{d}_M,\cdot)]$, and $\mathbf{K}_{\mathbf{D},\mathbf{D}}$ as the resulting kernel matrix from this dictionary. We enforce efficiency in function representation by selecting dictionaries $\mathbf{D}_i$ that $M_{i,t} << \mathcal{O}(t)$ for each i, following [26]. To be specific, we propose replacing the local update (8) in which the dictionary grows at each iteration by its projection onto subspace $\mathcal{H}_{\mathbf{D}_{i,t+1}} = \text{span}\{\kappa(\mathbf{d}_{i,n},\cdot)\}_{n=1}^{M_{t+1}}$ as

$$\begin{aligned}f_{i,t+1} := \mathcal{P}_{\mathcal{H}_{\mathbf{D}_{i,t+1}}}\Big[(1-\eta\lambda)f_{i,t} - \eta\Big(\nabla_{f_i}\ell_i(f_{i,t}(\mathbf{x}_{i,t}), y_{i,t}) \\ +c\sum_{j\in n_i}(f_{i,t}(\mathbf{x}_{i,t})-f_{j,t}(\mathbf{x}_{i,t}))\kappa(\mathbf{x}_{i,t},\cdot)\Big)\Big].\end{aligned} \tag{11}$$

We define projection $\mathcal{P}$ onto subspace $\mathcal{H}_{\mathbf{D}_{i,t+1}}$by use of kernel orthogonal matching pursuit [25] applied to the sequence of kernel dictionaries and weights with stopping tolerance ϵ. See [1][Section 3.2] for details. The coefficients and dictionary updates are given in Algorithm 1.

IV. Optimally Sparse Function Representation

We turn to establishing – based on extending Section IV of [26] to multi-agent settings – that Algorithm 1 converges with probability 1 to a neighborhood of the minimizer of the penalty function $\psi_c(f)$ [cf. (5)] and that the kernel dictionary that parameterizes the regression function f_i for each agent i remains finite. Let us define the local projected stochastic functional gradient associated with the update in (11) as

$$\begin{aligned}&\tilde{\nabla}_{f_i}\hat{\psi}_{i,c}(f_{i,t}(\mathbf{x}_{i,t}), y_{i,t}) = \\ &\qquad\Big(f_{i,t} - \mathcal{P}_{\mathcal{H}_{\mathbf{D}_{i,t+1}}}\Big[f_{i,t} - \eta\nabla_{f_i}\hat{\psi}_{i,c}(f_{i,t}(\mathbf{x}_{i,t}),y_{i,t})\Big]\Big)/\eta\end{aligned} \tag{12}$$

such that the local update of Algorithm 1 [cf. (11)] may be expressed as a stochastic projected functional gradient descent

$$f_{i,t+1} = f_{i,t} - \eta\tilde{\nabla}_{f_i}\hat{\psi}_{i,c}(f_{i,t}(\mathbf{x}_{i,t}), y_{i,t})\ . \tag{13}$$

Subsequently, we require the some technicalities common to kernelized stochastic methods, see [38]–[40].

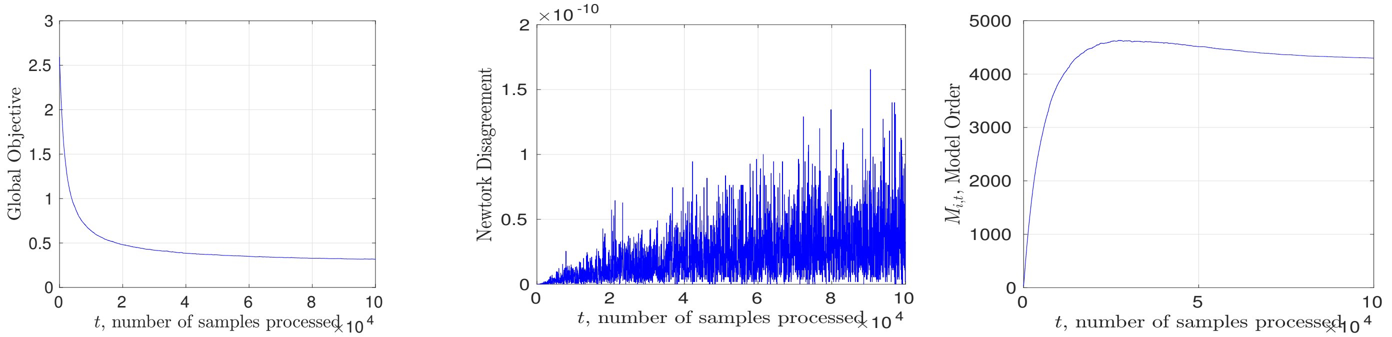
i) The feature space $\mathcal{X} \subset \mathbb{R}^p$ and target domain $\mathcal{Y} \subset \mathbb{R}$ are compact, and the reproducing kernel map satisfies

$$\sup_{\mathbf{x}\in\mathcal{X}} \sqrt{\kappa(\mathbf{x},\mathbf{x})} = X < \infty. \tag{14}$$

ii) The local losses $\ell_i(f_i(\mathbf{x}), y)$ are convex and differentiable w.r.t. the scalar argument $f_i(\mathbf{x})$ on $\mathbb{R}$ for all $\mathbf{x} \in \mathcal{X}$, $y \in \mathcal{Y}$. Moreover, the instantaneous losses $\ell_i : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ are C_i-Lipschitz continuous for all $z \in \mathbb{R}$ for a fixed $y \in \mathcal{Y}$.

iii) Let $\mathcal{F}_t$ be the filtration measuring the algorithm history: $\mathcal{F}_t = \{\mathbf{x}_u, y_u, f_u\}_{u=1}^t$. The projected functional stochastic gradient of the penalty which stacks (12) has finite conditional variance

$$\mathbb{E}[\|\tilde{\nabla}_f\hat{\psi}_c(f_t(\mathbf{x}_t), \mathbf{y}_t)\|^2_{\mathcal{H}} \mid \mathcal{F}_t] \leq \sigma^2. \tag{15}$$



(a) Global Objective vs. samples processed

(b) Disagreement vs. samples processed

(c) Model Order $M_{i,t}$ vs. samples processed

Fig. 1: In Fig. 1a, we plot the global objective $\sum_{i\in\mathcal{V}}(\mathbb{E}_{\mathbf{x}_i,y_i}[\ell_i(f_{i,t}(\mathbf{x}),y_i)])$ versus the number of samples processed, and observe convergence. In Fig. 1b we display the Hilbert-norm network disagreement $\sum_{(i,j)\in\mathcal{E}}\|f_{i,t}-f_{j,t}\|^2_{\mathcal{H}}$ with a penalty parameter $c=0.02$. In Fig. 1c, we plot the model order of a randomly chosen agent's regression function, which stabilizes to 4299.

Under the previous assumptions, the iterates of Algorithm 1 converge to a neighborhood of the minimizer of $\psi_c(f)$.

Theorem 1: Consider the sequence $\{f_t\}$ generated by Algorithm 1 with $f_0=0$ and regularizer $\lambda>0$. Suppose Assumptions i-iii hold, and we select $\eta<1/\lambda$ and compression budget $\epsilon=K\eta^{3/2}$ for any $K>0$. Then we have convergence to a neighborhood with probability 1 as

$$\liminf_{t\to\infty}\|f_t-f_c^*\|_{\mathcal{H}}\le\frac{\sqrt{\eta}}{\lambda}\Big[KV+\sqrt{K^2V^2+\lambda\sigma^2}\Big]=\mathcal{O}(\sqrt{\eta})\quad\text{a.s.}\tag{16}$$

Empirically, use of constant step-sizes maintains consistent algorithm adaptivity in the face of new data. Moreover, we may apply Theorem 3 of [26], which guarantees the model order of the function sequence remains finite.

Corollary 1: Denote $f_t\in\mathcal{H}^V$ as the sequence defined by Algorithm 1 with η and ϵ as in Theorem 1. Let M_t be the model order of function f_t. Then there exists a finite upper bound M^∞ such that, for all $t\ge 0$, the model order is always bounded as $M_t\le M^\infty$, and the model order of the limiting function $f_c^\infty=\lim_t f_t$ is finite.

Thus, use of constant step-sizes yields approximate convergence to f_c^* while ensuring finite memory (Corollary 1). Under an additional hypothesis, we have that consensus in the RKHS is attained, as we state next.

Theorem 2: Let Assumptions i - iii hold. Let f_c^* be the minimizer of the penalty function (5). Then, suppose the penalty parameter c in (5) approaches infinity $c\to\infty$, and that the node-pair differences $f_{i,c}^*-f_{j,c}^*$ are not orthogonal to mean transformation $\mathbb{E}_{\mathbf{x}_i}[\kappa(\mathbf{x}_i,\cdot)]$ of the local input spaces $\mathbf{x}_i$ for all $(i,j)\in\mathcal{E}$. Then $f_{i,c}^*=f_{j,c}^*$ for all $(i,j)\in\mathcal{E}$.

Thus, as long as a specific condition holds on the feature map induced by the kernelization of node i's data, consensus is achieved when we send the penalty parameter to infinity. Next, we asses Algorithm 1 in practice.

V. Numerical Experiments

Consider the task of kernel logistic regression from multi-class training data that is scattered across a multi-agent network. In this case, the merit of a particular regressor for agent i is quantified by its contribution to the class-conditional probability. Define a set of class-specific functions $f_{i,k}:\mathcal{X}\to\mathbb{R}$, and denote them jointly as $\mathbf{f}_i\in\mathcal{H}^C$, where $\{1,\ldots,C\}$ denotes the set of classes. Then, define the probabilistic model of the odds ratio of being in class c vs. all others

$$P(y_i=c\,|\,\mathbf{x}_i):=\frac{\exp(f_{i,k}(\mathbf{x}_i))}{\sum_{k'}\exp(f_{i,k'}(\mathbf{x}_i))}.\tag{17}$$

The negative log likelihood defined by (17) is the instantaneous loss (see, e.g., [37]) at sample $(\mathbf{x}_{i,n},y_{i,n})$:

$$\ell(\mathbf{f}_i,\mathbf{x}_{i,n},y_{i,n})=-\log P(y_i=y_{i,n}|\mathbf{x}_{i,n})+\frac{\lambda}{2}\sum_k\|f_{i,k}\|^2_{\mathcal{H}}\tag{18}$$

We generated the *brodatz* data set from a subset of [30]: from 13 texture images (i.e. D=13), we generate a set of 256 textons [41]. Next, for each overlapping patch of size 24-pixels-by-24-pixels within these images, we took the feature to be the associated $p=256$-dimensional texton histogram. The corresponding label was given by the index of the image from which the patch was selected. We then randomly selected $N=10000$ feature-label pairs for training and 5000 for testing. Each agent in network with $V=5$ observes a unique stream of samples from this data set. Here the communication graph is a random network with edges generated randomly between nodes with probability $1/5$ repeatedly until we obtain one that is connected, and then symmetrize it. We run Algorithm 1 for ten epoches: in each epoch we stream the entire training set to each agent. A Gaussian kernel is used with bandwidth $\sigma^2=0.1$, step-size $\eta=4$, compression budget $\epsilon=\eta^{3/2}$ with parsimony constant $K=0.04$, mini-batch size 32 and regularizer $\lambda=10^{-5}$. The penalty coefficient is $c=0.02$.

Results of this experiment are in Figure 1: Fig. 1a displays the global objective $\sum_{i\in\mathcal{V}}(\mathbb{E}_{\mathbf{x}_i,y_i}[\ell_i(f_{i,t}(\mathbf{x}),y_i)])$ relative to no. of samples, where we can clearly observe global convergence; Fig. 1b plots the network disagreement $\sum_{(i,j)\in\mathcal{E}}\|f_{i,t}-f_{j,t}\|^2_{\mathcal{H}}$, which remains small during training; and the model order of an agent chosen at random versus samples processed is given in Fig. 1c. The resulting decision function achieves 93.5% accuracy over the test set which is comparable with the accuracy of the centralized version (95.6%) [26]. However, the model order required is more than twice the model order in the centralized case (4358 in average v.s. 1833 [26]). Compared to other distributed classification algorithms, we outperform the state of the art by a significant margin: D4L achieves 75% accuracy [6].

References

[1] A. Koppel, S. Paternain, C. Richard, and A. Ribeiro, "Decentralized online learning with kernels," *arXiv preprint arXiv:1710.04062 (submitted to IEEE TSP, Oct. 2017)*, 2017.

[2] ——, "'decentralized efficient nonparametric stochastic optimization'," in *Signal and Information Processing (GlobalSIP), 2017 IEEE Global Conference on (to appear)*. IEEE, 2017.

[3] M. Anthony and P. L. Bartlett, *Neural network learning: Theoretical foundations*. cambridge university press, 2009.

[4] Z. Marinho, B. Boots, A. Dragan, A. Byravan, G. J. Gordon, and S. Srinivasa, "Functional gradient motion planning in reproducing kernel hilbert spaces," in *Proceedings of Robotics: Science and Systems*, Ann Arbor, MI, July 2016.

[5] A. Koppel, F. Jakubiec, and A. Ribeiro, "A saddle point algorithm for networked online convex optimization," *IEEE Trans. Signal Process.*, p. 15, Oct 2015.

[6] A. Koppel, G. Warnell, E. Stump, and A. Ribeiro, "D4l: Decentralized dynamic discriminative dictionary learning," *IEEE Trans. Signal and Info. Process. over Networks*, vol. (submitted), June 2017, available at http://www.seas.upenn.edu/ aribeiro/wiki.

[7] K. Slavakis, P. Bouboulis, and S. Theodoridis, "Online learning in reproducing kernel hilbert spaces," *Signal Processing Theory and Machine Learning*, pp. 883–987, 2013.

[8] J.-B. Li, S.-C. Chu, and J.-S. Pan, *Kernel Learning Algorithms for Face Recognition*. Springer, 2014.

[9] S. Haykin, "Neural networks: A comprehensive foundation," 1994.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[11] J. Mairal, F. Bach, and J. Ponce, "Task-driven dictionary learning," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 4, pp. 791–804, 2012.

[12] J. Liu, Q. Chen, and H. D. Sherali, "Algorithm design for femtocell base station placement in commercial building environments," in *INFOCOM, 2012 Proceedings IEEE*. IEEE, 2012, pp. 2951–2955.

[13] A. Ghosh and S. Sarkar, "Pricing for profit in internet of things," in *Information Theory (ISIT), 2015 IEEE International Symposium on*. IEEE, 2015, pp. 2211–2215.

[14] A. Koppel, J. Fink, G. Warnell, E. Stump, and A. Ribeiro, "Online learning for characterizing unknown environments in ground robotic vehicle models," in *Proc. Int. Conf. Intelligent Robots and Systems*.

[15] M. Schwager, P. Dames, D. Rus, and V. Kumar, "A multi-robot control policy for information gathering in the presence of unknown hazards," in *Robotics Research*. Springer, 2017, pp. 455–472.

[16] H. Robbins and S. Monro, "A stochastic approximation method," *Ann. Math. Statist.*, vol. 22, no. 3, pp. 400–407, 09 1951.

[17] G. Kimeldorf and G. Wahba, "Some results on tchebycheffian spline functions," *Journal of mathematical analysis and applications*, vol. 33, no. 1, pp. 82–95, 1971.

[18] B. Schölkopf, R. Herbrich, and A. J. Smola, "A generalized representer theorem," *Subseries of Lecture Notes in Computer Science Edited by JG Carbonell and J. Siekmann*, p. 416.

[19] V. Norkin and M. Keyzer, "On stochastic optimization and statistical learning in reproducing kernel hilbert spaces by support vector machines (svm)," *Informatica*, vol. 20, no. 2, pp. 273–292, 2009.

[20] Y. Engel, S. Mannor, and R. Meir, "The kernel recursive least-squares algorithm," *IEEE Transactions on Signal Processing*, vol. 52, no. 8, pp. 2275–2285, Aug 2004.

[21] C. Richard, J. C. M. Bermudez, and P. Honeine, "Online prediction of time series data with kernels," *IEEE Transactions on Signal Processing*, vol. 57, no. 3, pp. 1058–1067, 2009.

[22] J. Kivinen, A. J. Smola, and R. C. Williamson, "Online Learning with Kernels," *IEEE Transactions on Signal Processing*, vol. 52, pp. 2165–2176, August 2004.

[23] O. Dekel, S. Shalev-Shwartz, and Y. Singer, "The forgetron: A kernel-based perceptron on a fixed budget," in *Advances in Neural Information Processing Systems 18*. MIT Press, 2006, p. 259266. [Online]. Available: http://research.microsoft.com/apps/pubs/default.aspx?id=78226

[24] J. Zhu and T. Hastie, "Kernel Logistic Regression and the Import Vector Machine," *Journal of Computational and Graphical Statistics*, vol. 14, no. 1, pp. 185–205, 2005.

[25] Y. Pati, R. Rezaiifar, and P. Krishnaprasad, "Orthogonal Matching Pursuit: Recursive Function Approximation with Applications to Wavelet Decomposition," in *Proceedings of the Asilomar Conference on Signals, Systems and Computers*, 1993.

[26] A. Koppel, G. Warnell, E. Stump, and A. Ribeiro, "Parsimonious online learning with kernels via sparse projections in function space," *arXiv preprint arXiv:1612.04111*, 2016.

[27] B. Johansson, T. Keviczky, M. Johansson, and K. Johansson, "Subgradient methods and consensus algorithms for solving convex optimization problems," in *Proc. of the 47th IEEE Conference on Decision and Control*, Cancun, Mexico, 2008, pp. 4185–4190.

[28] S. Ram, A. Nedic, and V. Veeravalli, "Distributed stochastic subgradient projection algorithms for convex optimization," *J Optimiz. Theory App.*, vol. 147, no. 3, pp. 516–545, Sep. 2010.

[29] P. Chainais and C. Richard, "Learning a common dictionary over a sensor network," in *Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), 2013 IEEE 5th International Workshop on*. IEEE, 2013, pp. 133–136.

[30] P. Brodatz, *Textures: A Photographic Album for Artists and Designers*. Dover, 1966.

[31] A. Ribeiro, "Ergodic stochastic optimization algorithms for wireless communication and networking," *IEEE Transactions on Signal Processing*, vol. 58, no. 12, pp. 6369–6386, 2010.

[32] R. J. Kozick and B. M. Sadler, "Source localization with distributed sensor arrays and partial spatial coherence," *IEEE Transactions on Signal Processing*, vol. 52, no. 3, pp. 601–616, 2004.

[33] S. Shalev-Shwartz, O. Shamir, N. Srebro, and K. Sridharan, "Learnability, stability and uniform convergence," *Journal of Machine Learning Research*, vol. 11, no. Oct, pp. 2635–2670, 2010.

[34] T. Evgeniou, M. Pontil, and T. Poggio, "Regularization networks and support vector machines," *Advances in computational mathematics*, vol. 13, no. 1, pp. 1–50, 2000.

[35] K. Müller, T. Adali, K. Fukumizu, J. C. Principe, and S. Theodoridis, "Special issue on advances in kernel-based learning for signal processing [from the guest editors]," *IEEE Signal Process. Mag.*, vol. 30, no. 4, pp. 14–15, 2013. [Online]. Available: http://dx.doi.org/10.1109/MSP.2013.2253031

[36] R. Wheeden, R. Wheeden, and A. Zygmund, *Measure and Integral: An Introduction to Real Analysis*, ser. Chapman & Hall/CRC Pure and Applied Mathematics. Taylor & Francis, 1977. [Online]. Available: https://books.google.com/books?id=YDkDmQ_hdmcC

[37] K. Murphy, *Machine Learning: A Probabilistic Perspective*. MIT press, 2012.

[38] M. Pontil, Y. Ying, and D. xuan Zhou, "Error analysis for online gradient descent algorithms in reproducing kernel hilbert spaces," Tech. Rep., 2005.

[39] Y. Ying and D. X. Zhou, "Online regularized classification algorithms," *IEEE Transactions on Information Theory*, vol. 52, no. 11, pp. 4775–4788, Nov 2006.

[40] Y. Nesterov, "Introductory lectures on convex programming volume i: Basic course," 1998.

[41] T. Leung and J. Malik, "Representing and Recognizing the Visual Appearence of Materials using Three-dimensional Textons," *International Journal of Computer Vision*, vol. 43, no. 1, pp. 29–44, 1999.