ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-6654-0540-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICASSP43922.2022.9746612

DECENTRALIZED BILEVEL OPTIMIZATION FOR PERSONALIZED CLIENT LEARNING

Songtao Lu, Xiaodong Cui*, Mark S. Squillante, Brian Kingsbury, and Lior Horesh*

IBM Research, Thomas J. Watson Research Center, Yorktown Heights, New York 10598, USA

ABSTRACT

Decentralized optimization with multiple networked clients/learners has advanced machine learning significantly over the past few years. When data distributions at different nodes/locations are heterogeneous, consensus-based decentralized algorithms ignore distinctive features of local data samples. In this paper, we propose a decentralized client adaptation strategy for personalized learning by taking local client data structures into account. It turns out that optimizing the model parameters can be formulated as a decentralized bilevel programming problem. Motivated by this application, we propose a stochastic primal-dual framework for solving decentralized bilevel nonconvex problems and show that the devised algorithm achieves the Karush–Kuhn–Tucker (KKT) points for this class of problems at a rate of $\mathcal{O}(1/\sqrt{nT})$, where n denotes the number of total learners and T the total number of iterations. Multiple experiments show the superiority of our proposed method compared to state-of-the-art methods in terms of both training speed and testing accuracy for decentralized learning problems on real datasets.

Index Terms— Stochastic primal dual decentralized algorithm (SPD), decentralized bilevel optimization (DBO), personalized client learning (PCL)

1. INTRODUCTION

Deep neural networks trained in a supervised manner on massive datasets can make remarkably accurate predictions. However, when data samples are limited or there are multiple training tasks, data heterogeneity becomes one of the major barriers that prevent an increase in testing/validation accuracy. Hence, there is a need for learning algorithms that can balance the personalized data structure of each task and the permutation-invariant latent space/features among all the tasks. For example, model-agnostic meta-learning (MAML) [1], a hierarchical learning structure, exhibits fast adaptation performance of learning unseen tasks with some good initialization learned from other tasks, implying the potential benefits of leveraging the similarities among tasks over the heterogeneous datasets.

The key technique used in MAML is building two levels of learners, i.e., a meta-learner and a task-specific learner, which respectively minimize the task-averaged loss and individual loss of each task. Besides establishing the strong empirical performance of MAML [1–3], prior works have studied regret bounds of online MAML for either the convex case [4, 5] or nonconvex case [6], convergence rate guarantees of MAML algorithms [7], generalization error bounds [8], and federated [9, 10] or decentralized [11] MAML. A recent deeper analysis proposed an almost no inner loop (ANIL) framework [12] and discovered that the effectiveness of MAML is actually mainly due to feature reuse rather than meta-initialization, which further justifies that only the task-specific head of neural nets is sufficient for rapid learning. ANIL is also computationally much more efficient than MAML, as only a small set of parameters are involved in the adaptation step, and it is shown in [13] that ANIL achieves the same order of convergence rate as MAML to the first-order stationary points of general nonconvex problems.

Inspired by MAML and ANIL, it is of interest to design efficient and simple training algorithms for finding the optimal solutions of this problem. Mathematically, training a MAML model can be formulated as one class of bilevel optimization problems (a.k.a. min-max Stackelberg games) in which two levels of optimization problems need to be solved with coupled variables in both the upper and lower levels. A classical method is a double-loop algorithm (a.k.a. bilevel stochastic approximation (BSA) method) for solving the stochastic nonconvex bilevel programming problem when the lower level problem is strongly convex [14]. The idea of BSA is to solve the lower level sub-problem up to some tolerance first, and then to perform one step of gradient descent on the loss of the upper level problem. It turns out that, to achieve an ϵ-stationary point for a class of nonconvex problems, BSA requires $\mathcal{O}(1/\epsilon^2)$ total number of iterations. Subsequently, it is shown in [15] that a so-called stochastic bilevel optimizer (stoBiO) only needs a constant number of inner loop steps to attain a convergence rate of $\mathcal{O}(1/\epsilon^2)$ to the stationary points. There are also single-loop algorithms such as two-timescale stochastic approximation (TTSA) developed in [16], which employs different learning rates in the inner and outer optimization steps to guarantee a convergence rate of $\mathcal{O}(1/\epsilon^{2.5})$, and the Single-Timescale stochAstic BiLevEl optimization (STABLE) method proposed in [17], which adopts an error correction term so that STABLE uses only a single-timescale learning rate to obtain an $\mathcal{O}(1/\epsilon^2)$ convergence rate. Recently, a tighter convergence analysis [18] points out that the simple TTSA-type of alternating stochastic gradient method can still reach the convergence rate of $\mathcal{O}(1/\epsilon^2)$ with only a single timescale scheduled learning rate for updating the optimization variables in both upper and lower problems. When the momentum-based techniques are used, the convergence rate of bilevel algorithms can be further increased to $\mathcal{O}(1/\epsilon^{1.5})$ [19, 20].

However, all of the above bilevel algorithms are centralized. If there are multiple computing resources/clients connected through communication channels, to the best of our knowledge, none of the existing decentralized optimization algorithms [21–23], e.g., distributed stochastic gradient descent (DSGD) [24] and/or gradient-tracking based nonconvex stochastic decentralized (GNSD) algorithms [25], can solve the bilevel programming problems over a network. Targeting this issue, we propose a stochastic primal-dual algorithm for solving decentralized bilevel (SPDB) optimization problems and build a general decentralized training framework for personalized client learning (PCL). As an application of decentralized bilevel optimization (DBO), PCL aims to separate the shared feature space and the private space so that the training process can be accelerated by leveraging the data over the network without loss of personalized local features. We show that, under the standard assumptions, the proposed SPDB can find the Karush–Kuhn–Tucker (KKT) points of a class of DBO problems at a rate of $\mathcal{O}(1/(n\epsilon^2))$ which matches the same iteration and sample complexities as DSGD

*The authors contributed to this work equally.

for classical nonconvex problems, where n denotes the total number of clients. The main contributions of this work are as follows:

- The application of our proposed decentralized PCL framework covers a wide range of distributed hierarchical optimization problems, especially with personalized consideration of learning models.
- SPDB is a single-loop and single-timescale algorithm that can achieve a linear speedup of the convergence rate to the KKT points with respect to (w.r.t.) the number of clients. To the best of our knowledge, SPDB is the first stochastic algorithm that solves the DBO problem.
- Numerical results that illustrate the proposed decentralized PCL model with SPDB achieves the best performance compared with the state-of-the-art decentralized methods for training over heterogeneous networks in terms of both validation/test-accuracy and convergence speed.

2. MOTIVATION AND PROBLEM FORMULATION

In this work, we propose a decentralized PCL scheme for training over heterogeneous networks. To be more specific, consider that there are n learners connected through a graph denoted by $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where each node i has a set of data samples following distribution $\mathcal{D}_i$. The goal of the networked learners is to jointly minimize the following (possibly nonconvex) optimization problem:

$$\min_{\{\boldsymbol{w}_i, \forall i\}} \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}_{\xi\sim\mathcal{D}_i}[l(\boldsymbol{w}_i, \mathcal{T}_i^*(\boldsymbol{w}_i;\xi))],\ \text{s.t. } \boldsymbol{w}_i = \boldsymbol{w}_j, \forall j \in \mathcal{N}_i,$$

where $l(\cdot)$ denotes a general loss function, $\boldsymbol{w}_i$ is the model parameters at each node, $\mathcal{N}_i$ denotes the neighboring learners of node i, and the transform function $\mathcal{T}_i^*(\boldsymbol{w}_i;\xi) = \arg\min_{\mathcal{M}_i} l(\boldsymbol{w}_i, \mathcal{M}_i(\xi)), \xi \sim \mathcal{D}_i$, extracts and processes the distinct features (relative to the rest of the nodes) to canonicalize data heterogeneity. In practice, mapping $\mathcal{M}_i(\cdot)$ is parameterized either in a linear or nonlinear way [26].

Motivated by decentralized PCL, we consider the following linearly constrained bilevel optimization problem:

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}_{\xi\in\mathcal{D}_i^{(u)}}[f_i(\boldsymbol{x}_i, \boldsymbol{y}_i^*(\boldsymbol{x}_i);\xi)] \tag{1a}$$

$$\text{s.t. } \boldsymbol{A}\boldsymbol{x} = 0,\ \boldsymbol{y}_i^*(\boldsymbol{x}_i) = \arg\min_{\boldsymbol{y}_i\in\mathbb{R}^m} \mathbb{E}_{\zeta\in\mathcal{D}_i^{(l)}}[g_i(\boldsymbol{x}_i, \boldsymbol{y}_i;\zeta)], \tag{1b}$$

where $f_i(\boldsymbol{x}_i, \boldsymbol{y}_i^*(\boldsymbol{x}_i)) \triangleq \mathbb{E}_{\xi\in\mathcal{D}_i^{(u)}} f_i(\boldsymbol{x}_i, \boldsymbol{y}_i^*(\boldsymbol{x}_i);\xi)$ is a smooth loss function (possibly nonconvex), $\mathcal{D}_i^{(u)}$ and $\mathcal{D}_i^{(l)}$ respectively denote the local data distributions at the upper and lower levels of this optimization problem, $g_i(\cdot)$ denotes the lower-level loss function at node i, $\boldsymbol{x} \triangleq [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]^T$, and $\boldsymbol{A} \in \mathbb{R}^{|\mathcal{E}|\times n}$ represents the incidence matrix.[1]

3. STOCHASTIC PRIMAL-DUAL FRAMEWORK

Towards this end, we first introduce the augmented Lagrangian

$$\mathcal{L}_\rho(\boldsymbol{x}, \boldsymbol{\lambda}) = n^{-1}\mathbf{1}^T f(\boldsymbol{x}, \boldsymbol{y}^*(\boldsymbol{x})) + \langle \boldsymbol{\lambda}, \boldsymbol{A}\boldsymbol{x}\rangle + \frac{\rho}{2}\|\boldsymbol{A}\boldsymbol{x}\|^2, \tag{2}$$

where $f(\boldsymbol{x}, \boldsymbol{y}^*(\boldsymbol{x})) \triangleq [f_1(\boldsymbol{x}_1, \boldsymbol{y}_1^*(\boldsymbol{x}_1)), \ldots, f_n(\boldsymbol{x}_n, \boldsymbol{y}_n^*(\boldsymbol{x}_n))]^T$, $\boldsymbol{\lambda} \in \mathbb{R}^n$ denotes the dual variable that enforces the consensus of the primal variables, i.e., $\boldsymbol{A}\boldsymbol{x}$, and $\rho > 0$ is the penalty parameter of the augmented term.

Stochastic primal-dual structure. Define $g_i(\boldsymbol{x}_i, \boldsymbol{y}_i) \triangleq \mathbb{E}_{\zeta\in\mathcal{D}_i^{(l)}}[g_i(\boldsymbol{x}_i, \boldsymbol{y}_i;\zeta)], \forall i$, and define $\overline{\nabla}_{\boldsymbol{x}_i} f(\boldsymbol{x}_i, \boldsymbol{y}_i)$ to be the vector obtained by replacing $\boldsymbol{y}_i^*(\boldsymbol{x})$ in $\nabla_{\boldsymbol{x}_i} f_i(\boldsymbol{x}_i, \boldsymbol{y}_i^*(\boldsymbol{x}_i))$ by $\boldsymbol{y}_i$.[2] The proposed stochastic primal-dual algorithm for solving (1) is

$$\boldsymbol{y}^{r+1} = \boldsymbol{y}^r - \beta\boldsymbol{h}_g^r, \tag{3a}$$

$$\boldsymbol{x}^{r+1} = \arg\min_{\boldsymbol{x}} \langle \boldsymbol{h}_f^r + \gamma\boldsymbol{A}^T(\boldsymbol{\lambda}^r + \rho\boldsymbol{A}\boldsymbol{x}^r), \boldsymbol{x} - \boldsymbol{x}^r\rangle + \frac{\alpha}{2}\|\boldsymbol{x} - \boldsymbol{x}^r\|^2,$$

$$\boldsymbol{\lambda}^{r+1} = \boldsymbol{\lambda}^r + \rho\boldsymbol{A}\boldsymbol{x}^{r+1}, \tag{3b}$$

where r indexes iterations, $\boldsymbol{y} \triangleq [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n]^T$, $g(\boldsymbol{x}, \boldsymbol{y}) \triangleq [g_1(\boldsymbol{x}_1, \boldsymbol{y}_1), \ldots, g_n(\boldsymbol{x}_n, \boldsymbol{y}_n)]^T$, $\boldsymbol{h}_g^r$ and $\boldsymbol{h}_f^r$ are stochastic estimates of $\nabla_{\boldsymbol{y}} g(\boldsymbol{x}^r, \boldsymbol{y}^r)$ and $n^{-1}\overline{\nabla}_{\boldsymbol{x}} f(\boldsymbol{x}^r, \boldsymbol{y}^{r+1})$, $1/\alpha$ and $\rho > 0$ respectively denote the step-sizes of the primal and dual updates, and β is the step-size of the lower-level iterates' update.

As the objective function in the $\boldsymbol{x}$ sub-problem is quadratic, we can obtain a closed-form expression for the $\boldsymbol{x}$ update:

$$\boldsymbol{x}^{r+1} = \boldsymbol{x}^r - \frac{1}{\alpha}\left(\boldsymbol{h}_f^r + \gamma\boldsymbol{A}^T\boldsymbol{\lambda}^r + \rho\gamma\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{x}^r\right). \tag{4}$$

Then, subtracting (4) with the same from its previous iteration, we can have the following update of $\boldsymbol{x}$ after using (3b)

$$\boldsymbol{x}^{r+1} = 2\boldsymbol{W}\boldsymbol{x}^r - \boldsymbol{W}\boldsymbol{x}^{r-1} - \frac{1}{\alpha}\left(\boldsymbol{h}_f^r - \boldsymbol{h}_f^{r-1}\right), \tag{5}$$

where $\boldsymbol{W} \triangleq \boldsymbol{I} - \rho\boldsymbol{A}^T\boldsymbol{A}/\tau$ and $\tau \triangleq \alpha/\gamma$. Therefore, the above primal and dual updates can be merged into a single step.

The detailed implementation of SPDB from a local view is shown in Algorithm 1, where $\boldsymbol{h}_{g,i}^r, \boldsymbol{h}_{f,i}^r, \forall i$, denote the gradient estimates of $\nabla_{\boldsymbol{y}_i} g_i(\boldsymbol{x}_i^r, \boldsymbol{y}_i^r), n^{-1}\overline{\nabla}_{\boldsymbol{x}_i} f_i(\boldsymbol{x}_i^r, \boldsymbol{y}_i^{r+1}), \forall i$, and $\boldsymbol{W}_{ij}$ denotes the entry at the ith row and jth column of matrix $\boldsymbol{W}$.

Remark 1. When there is no lower-level optimization problem, SPDB reduces to a stochastic primal-dual decentralized (SPD) algorithm that is almost identical to the D^2 algorithm [27], with only a difference in the ordering of the local update and weights aggregation from neighbors. When full gradient is used, SPD reduces to the deterministic gradient primal-dual algorithm [28].

Remark 2. Note that SPDB/SPD only needs one communication round per iteration, half that of gradient tracking technique algorithms like GNSD [25] and in-Network succEssive conveX approximaTion (NEXT) [29].

Algorithm 1 Stochastic Primal-dual Decentralized algorithm for Bilevel Optimization (SPDB)

Initialization: $\boldsymbol{x}^0, \boldsymbol{y}^0, \alpha, \beta$
for $r = 1, \ldots, T$ **do**
  $\boldsymbol{y}_i^{r+1} = \boldsymbol{y}_i^r - \beta\boldsymbol{h}_{g,i}^r, \forall i.$ ▷ adaptation step
  $\boldsymbol{x}_i^{r+1} = \sum_{j\in\mathcal{N}_i} \boldsymbol{W}_{ij}\left(2\boldsymbol{x}_j^r - \boldsymbol{x}_j^{r-1}\right) - \frac{1}{\alpha}\left(\boldsymbol{h}_{f,i}^r - \boldsymbol{h}_{f,i}^{r-1}\right).$
end for

4. THEORETICAL CONVERGENCE RESULTS

Assumptions. The theoretical results are based on the following assumptions on the properties of the loss functions in both the upper- and lower-level optimization problems, which are mainly related to the continuity of the objective function and stochasticity of the gradient estimates:

A1. (Lipschitz continuity) Assume that functions $f_i(\cdot), \nabla f_i(\cdot,), \nabla g_i(\cdot)$, $\nabla^2 g(\cdot), \forall i$, are Lipschitz continuous with constants $L_{f,0}, L_{f,1}, L_{g,1}, L_{g,2}$ for both $\boldsymbol{x}$ and $\boldsymbol{y}$.

[1] Here, we assume the problem dimension is 1, without loss of generality, to simplify the notation in this paper.

[2] When $g_i(\cdot)$ is strongly convex, $\overline{\nabla}_{\boldsymbol{x}_i} f(\boldsymbol{x}_i, \boldsymbol{y}_i) \triangleq \nabla_{\boldsymbol{x}_i} f(\boldsymbol{x}_i, \boldsymbol{y}_i) - \nabla^2_{\boldsymbol{x}_i\boldsymbol{y}_i} g(\boldsymbol{x}_i, \boldsymbol{y}_i)[\nabla^2_{\boldsymbol{y}_i\boldsymbol{y}_i} g_i(\boldsymbol{x}_i, \boldsymbol{y}_i)]^{-1}\nabla_{\boldsymbol{y}_i} f_i(\boldsymbol{x}_i, \boldsymbol{y}_i)$ [14, 16, 18].

A2. (Strong convexity of $g(\cdot)$ w.r.t. $\boldsymbol{y}$) Function $g(\cdot)$ is μ-strongly convex w.r.t. $\boldsymbol{y}$.

A3. (Connectivity of graph) The communication graph $\mathcal{G}$ is well connected, i.e., $1^T\boldsymbol{L}=0$ where $\boldsymbol{L}=\boldsymbol{A}^T\boldsymbol{A}$, and the second smallest eigenvalue of $\boldsymbol{L}$ is strictly positive, i.e., $\widetilde{\sigma}_{\min}(\boldsymbol{A}^T\boldsymbol{A})>0$.

A4. (Stochasticity of gradient estimate) The stochastic estimates $\nabla_{\boldsymbol{x}_i} f_i(\boldsymbol{x}_i,\boldsymbol{y}_i;\xi)$, $\nabla_{\boldsymbol{y}_i} g_i(\boldsymbol{x}_i,\boldsymbol{y}_i;\zeta)$, $\nabla^2_{\boldsymbol{y}_i\boldsymbol{y}_i} g_i(\boldsymbol{x}_i,\boldsymbol{y}_i;\zeta), \forall i$, are unbiased and their variances are bounded by $\sigma_f^2,\sigma_{g,1}^2,\sigma_{g,2}^2$.

Convergence analysis of SPDB. First, we can show that the difference between two successive $\boldsymbol{x}$-iterates is upper bounded on the order of $1/\alpha^2$ by the following lemma.

Lemma 1. *Under A1, A3, A4, suppose that iterates $\{\boldsymbol{x}^r,\forall r\}$ are generated by* (5)*. Then, there exists a constant C such that $\mathbb{E}\|\boldsymbol{x}^{r+1}-\boldsymbol{x}^r\|^2\le C^2/\alpha^2$, $\forall r$, where C only depends on the constants defined in A1, A3, A4.*

Define $\mathcal{F}^r\triangleq\sigma\{\boldsymbol{y}^0,\boldsymbol{x}^0,\ldots,\boldsymbol{x}^r,\boldsymbol{y}^{r+1}\}$ to be the filtration of the random variables up to iteration r where $\sigma\{\cdot\}$ denotes the σ-algebra generated by the random variables. Then, we can quantify the changes of the Lagrangian $\mathcal{L}(\boldsymbol{x},\boldsymbol{\lambda})=n^{-1}\mathbf{1}^T f(\boldsymbol{x},\boldsymbol{y}^*(\boldsymbol{x}))+\langle\boldsymbol{\lambda},\boldsymbol{A}\boldsymbol{x}\rangle$ from point $(\boldsymbol{x}^r,\boldsymbol{\lambda}^r)$ to $(\boldsymbol{x}^{r+1},\boldsymbol{\lambda}^{r+1})$ after one round of variable updates in the following lemma.

Lemma 2. *Under A1-A4, when $\|\mathbb{E}[\boldsymbol{h}_f^r|\boldsymbol{x}^r,\boldsymbol{y}^{r+1}]-\overline{\nabla}_{\boldsymbol{x}} f(\boldsymbol{x}^r,\boldsymbol{y}^{r+1})\|\le b_r$, the sequence generated by SPDB satisfies*

$$\begin{aligned}\mathbb{E}\left[\mathcal{L}(\boldsymbol{x}^{r+1},\boldsymbol{\lambda}^{r+1})|\mathcal{F}^r\right]-\mathcal{L}(\boldsymbol{x}^r,\boldsymbol{\lambda}^r)\le&-\frac{1}{2\alpha}\|\nabla\mathcal{L}(\boldsymbol{x}^r,\boldsymbol{\lambda}^r)\|^2\\&-\left(\frac{\alpha}{2}-\frac{L_{f,1}}{2}\right)\|\mathbb{E}[\boldsymbol{x}^{r+1}-\boldsymbol{x}^r]\|^2+\frac{2L_{f,1}^2}{n\alpha}\|\boldsymbol{y}^{r+1}-\boldsymbol{y}^*(x^r)\|^2\\&+\left(\frac{\sigma_{\max}(\boldsymbol{A}^T\boldsymbol{A})}{\tau}+\frac{1}{\rho}\right)\mathbb{E}\|\boldsymbol{\lambda}^{r+1}-\boldsymbol{\lambda}^r\|^2+\frac{2b_r^2}{n\alpha}+\frac{L_{f,1}\sigma_f^2}{2n\alpha^2},\end{aligned}$$

where $\sigma_{\max}(\boldsymbol{A}^T\boldsymbol{A})$ denotes the maximum eigenvalue of $\boldsymbol{A}^T\boldsymbol{A}$.

Next, we will use the successive difference of the primal variables and the distance from $\boldsymbol{y}^r$ to the minimizer of the lower-level optimization problem at point $\boldsymbol{x}^r$ to quantify the successive difference of the dual variables in the ascent part of the Lagrangian.

Lemma 3. *Under A1-A4, define $\boldsymbol{D}\triangleq\alpha\boldsymbol{I}-\rho\gamma\boldsymbol{A}^T\boldsymbol{A}$. Suppose that the sequence $\{\boldsymbol{x}^r,\boldsymbol{y}^r,\boldsymbol{\lambda}^r\}$ is generated by SPDB. Then, we have*

$$\begin{aligned}\frac{1}{\rho}\|\mathbb{E}[\boldsymbol{\lambda}^{r+1}-\boldsymbol{\lambda}^r]\|^2\le&\frac{4L_{f,1}^2}{n^2\rho\gamma\tilde{\sigma}_{\min}(\boldsymbol{A}^T\boldsymbol{A})}\mathbb{E}\left\|\boldsymbol{x}^r-\boldsymbol{x}^{r-1}\right\|^2\\&+\frac{4}{\rho\gamma\tilde{\sigma}_{\min}(\boldsymbol{A}^T\boldsymbol{A})}\left\|\mathbb{E}[\boldsymbol{\omega}^{r+1}]\right\|^2_{\boldsymbol{D}^T\boldsymbol{D}}+\frac{8\beta^2L_{f,1}^2L_{g,1}^2}{\rho\gamma\tilde{\sigma}_{\min}(\boldsymbol{A}^T\boldsymbol{A})}\frac{\sigma_{g,1}^2}{n}\\&+\frac{8\beta^2L_{f,1}^2L_{g,1}^2}{n^2\rho\gamma\widetilde{\sigma}_{\min}(\boldsymbol{A}^T\boldsymbol{A})}\mathbb{E}\|\boldsymbol{y}^r-\boldsymbol{y}^*(x^r)\|^2+\frac{4(b_r+b_{r-1})^2}{\rho\gamma},\end{aligned}\tag{6}$$

where $\boldsymbol{\omega}^{r+1}\triangleq(\boldsymbol{x}^{r+1}-\boldsymbol{x}^r)-(\boldsymbol{x}^r-\boldsymbol{x}^{r-1})$.

Now, the ascent part measured by the successive difference of the dual variables is partially transferred to the term $\|\mathbb{E}[\boldsymbol{\omega}^{r+1}]\|^2$. Using (4), we can construct the following recursion that establishes descent w.r.t. $\|\mathbb{E}[\boldsymbol{\omega}^{r+1}]\|^2$.

Lemma 4. *Under A1-A4, suppose that the sequence is generated by SPDB. Then, there exists a constant $\vartheta>0$ such that*

$$\begin{aligned}\mathcal{Q}^{r+1}-\mathcal{Q}^r\le&-\frac{1}{2}\|\mathbb{E}[\boldsymbol{\omega}^{r+1}]\|^2_{\boldsymbol{D}}+\frac{2L_{f,1}L_{g,1}^2\beta^2}{n^2\alpha}\mathbb{E}\|\boldsymbol{y}^r-\boldsymbol{y}^*(\boldsymbol{x}^r)\|^2\\&+\left(\left(\frac{\alpha\vartheta}{2}+\frac{1}{n^2\alpha}\right)L_{f,1}+\frac{\vartheta}{4}\right)\|\mathbb{E}[\boldsymbol{x}^{r+1}-\boldsymbol{x}^r]\|^2\\&+\left(2\beta^2\sigma_{g,1}^2+\frac{\sigma_f^2L_{f,1}}{\alpha^2}\right)\frac{1}{\alpha n\vartheta}+\frac{(b_r+b_{r-1})^2}{\vartheta},\end{aligned}\tag{7}$$

where $\mathcal{Q}^r\triangleq\frac{\rho\gamma}{2}\|\boldsymbol{A}\mathbb{E}[\boldsymbol{x}^r]\|^2+\frac{1}{2}\|\mathbb{E}[\boldsymbol{x}^r-\boldsymbol{x}^{r-1}]\|^2_{\boldsymbol{D}}+\frac{L_{f,1}}{n^2\alpha}\|\boldsymbol{x}^r-\boldsymbol{x}^{r-1}\|^2$.

Combining the contraction property of the lower-level optimization update w.r.t. $\boldsymbol{y}$ shown in [18, Lemma 3], we have the following descent lemma by applying Lemma 1 – Lemma 4.

Lemma 5. *Under A1-A4, suppose that sequence $\{\boldsymbol{x}^r,\boldsymbol{y}^r,\boldsymbol{\lambda}^r,\forall r\}$ is generated by SPDB. When*

$$\vartheta=\frac{1}{16},c\triangleq\frac{16\alpha\sigma_{\max}(\boldsymbol{A}^T\boldsymbol{A})}{\kappa\widetilde{\sigma}_{\min}(\boldsymbol{A}^T\boldsymbol{A})},\kappa\triangleq\frac{\rho\sigma_{\max}(\boldsymbol{A}^T\boldsymbol{A})}{\alpha}<1\tag{8}$$

and the step-sizes satisfy $\alpha>\widetilde{C}'L_{f,1}$ and $\widetilde{C}''/\alpha<\beta\le2/(\mu+L_{g,1})$, then there exist constants $C_1,C_2,C_3,C_4,C_5>0$ such that

$$\begin{aligned}\mathcal{P}^{r+1}-\mathcal{P}^r\le&-\frac{1}{2\alpha}\mathbb{E}\|\nabla\mathcal{L}(\boldsymbol{x}^r,\boldsymbol{\lambda}^r)\|^2-C_1\|\mathbb{E}[\boldsymbol{x}^{r+1}-\boldsymbol{x}^r]\|^2\\&-C_2\mathbb{E}\|\boldsymbol{y}^r-\boldsymbol{y}^*(\boldsymbol{x}^r)\|^2+C_3\frac{b_r^2+b_{r-1}^2}{\alpha}+C_4\frac{\sigma_f^2}{n\alpha^2}+C_5\beta^2\frac{\sigma_{g,1}^2}{n}\end{aligned}$$

where the potential/Lyapunov-like function is defined as

$$\begin{aligned}\mathcal{P}^r\triangleq&\,\mathbb{E}[\mathcal{L}(\boldsymbol{x}^r,\boldsymbol{\lambda}^r)]+\frac{c\rho\gamma}{2}\|\boldsymbol{A}\mathbb{E}[\boldsymbol{x}^r]\|^2+\frac{c}{2}\|\mathbb{E}[\boldsymbol{x}^r-\boldsymbol{x}^{r-1}]\|^2_{\boldsymbol{D}}\\&+\frac{1}{n^2}\left(\left(\frac{\sigma_{\max}(\boldsymbol{A}^T\boldsymbol{A})}{\tau}+\frac{1}{\rho}\right)\frac{4L_{f,1}^2}{\tilde{\sigma}_{\min}(\boldsymbol{A}^T\boldsymbol{A})}+\frac{cL_{f,1}}{\alpha}\right)\mathbb{E}\|\boldsymbol{x}^r-\boldsymbol{x}^{r-1}\|^2\\&+\frac{1}{n^2}\mathbb{E}\|\boldsymbol{y}^r-\boldsymbol{y}^*(\boldsymbol{x}^r)\|^2,\end{aligned}\tag{9}$$

and the constants $\widetilde{C}',\widetilde{C}''$ only depend on the parameters defined in A1–A4.

Theoretical convergence rate of SPDB. From the above analysis, we know that the potential function $\mathcal{P}^r$ is monotonically decreasing up to some error. Combining the facts that b_r in Lemma 2 shrinks exponentially w.r.t. the mini-batch size of $\boldsymbol{h}_f^r$ under a certain sampling scheme [14, 16] and that $\mathcal{P}^r$ is lower bounded will immediately yield the following theoretical convergence rate guarantees.

Theorem 1. *Suppose that A1-A4 hold. When step-sizes are chosen as $\alpha\sim\mathcal{O}(\sqrt{T/n})$, $\beta\sim\mathcal{O}(\sqrt{n/T})$, $\tau\ge\rho\sigma_{\max}(\boldsymbol{A}^T\boldsymbol{A})$ the mini-batch size of $\boldsymbol{h}_f^r$ is $\mathcal{O}(\log(T))$, and $T\ge\mathcal{O}(n^4)$, then the iterates $\{\boldsymbol{x}_i^r,\boldsymbol{\lambda}^r,\boldsymbol{y}_i^r,\forall i,r\}$ generated by SPDB satisfy*

$$\frac{1}{T}\sum_{r=1}^T\mathbb{E}[\|\nabla\mathcal{L}(\boldsymbol{x}^r,\boldsymbol{\lambda}^r)\|^2]\sim\frac{1}{T}\sum_{r=1}^T\mathbb{E}[\|\boldsymbol{A}\boldsymbol{x}^r\|^2]\sim\mathcal{O}(1/\sqrt{nT})$$

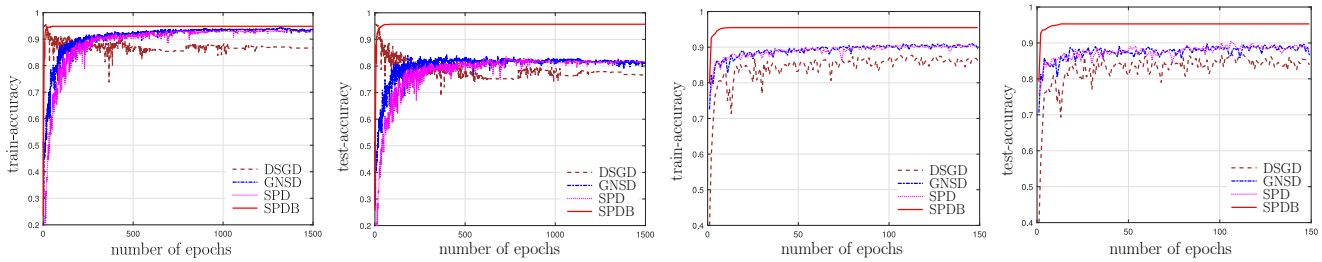
with $T^{-1}\sum_{r=1}^T\mathbb{E}[\|\boldsymbol{y}^r-\boldsymbol{y}^*(\boldsymbol{x}^r)\|^2]\sim\mathcal{O}(1/\sqrt{nT})$, where T denotes the total number of iterations.

Remark 3. Passing the limit of the sequence, we can get that the limit point $(\boldsymbol{x}^*,\boldsymbol{y}^*,\boldsymbol{\lambda}^*)$ is an exact KKT point of problem (1) .

Remark 4. Theorem 1 quantifies the number of iterations required to achieve the ϵ-approximate KKT points of (1) (including both the first-order stationarity of the solutions and the constraints violation) to be on the order of $1/(n\epsilon^2)$. Therefore, it follows that a linear speedup w.r.t. the number of learners can be achieved by SPDB. Note that SPDB is a single timescale algorithm as the learning rates satisfy $1/\alpha\sim\beta\sim\mathcal{O}(\sqrt{n/T})$.

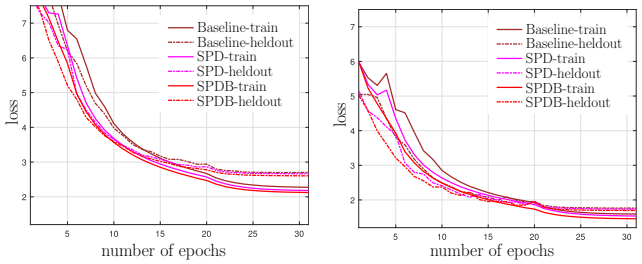
5. NUMERICAL RESULTS

Toy example. We first compare the performance of our proposed SPDB algorithm to DSGD [24], GNSD [25], and SPD/D^2 [27] on a decentralized classification learning problem using the MNIST dataset. The neural network for each agent has only one hidden layer with 32 neurons followed by sigmoid activation functions, where the weights at the last layer are used for adaptation and the rest of the weights must agree across the learners, and where there are 10 learners connected over a random Erdős–Rényi graph. We split the



(a) Training (limited data) (b) Testing (limited data) (c) Training (full dataset) (d) Testing (full dataset)

Fig. 1: Convergence performance comparison of SPDB with state-of-the-art decentralized training methods.



(a) Learner 1 (b) Learner 4

Fig. 2: Convergence performance comparison of SPDB, SPD/D^2 and DSGD on the training and heldout datasets.

whole dataset such that in the ith agent 95% of the data samples have label i and the remaining samples are drawn randomly from the full dataset. The initial learning rate of all the algorithms is 0.02, the mini-batch sizes for both $\boldsymbol{h}_f^r$ and $\boldsymbol{h}_g^r$ are 16, and the ratio ρ/τ in SPDB is 0.2. In Figure 1a and Figure 1b, only 128 training data samples at each learner are chosen to create a limited data scenario. In Figure 1c and Figure 1d, the full training dataset is used. It can be readily observed that SPDB achieves the best training and testing accuracy with a fast convergence rate, showing the merits of SPDB in terms of both training speedup over multiple learners and personalized adaptation at each node.

Automatic speech recognition (ASR). Experiments are also conducted on decentralized training of acoustic models for ASR. There are 50 hours of wideband speech data from five sources with each source contributing 10 hours of speech. The five sources consist of data from the following five domains: Broadcast News data, IBM internal dictation data, IBM internal meeting data, hospitality (travel and hotel reservation) data, and accented data. There are five learners, each with access to only one data source, which gives rise to heterogeneous data distributions across learners. In addition to the 10 hours of training data, each learner also has about 2 hours of speech used as a heldout set. The acoustic models are evaluated on four test sets from domains of broadcast news (2.21 hours), hospitality (0.34 hours), Asian-accented (2.41 hours) and Latin-accented (3.12 hours) speech, respectively. They are selected from public and real-world client data. They match some of the domains in the training data.

The acoustic model is a bi-directional LSTM (BLSTM) based deep neural network-hidden Markov model (DNN-HMM) containing 5 bidirectional LSTM layers with 256 cells per layer per direction. There is a linear projection layer of 256 hidden units between the topmost BLSTM layer and the softmax output layer. There are $9,300$ output units in the softmax output layer corresponding to context-dependent HMM states. The LSTM is unrolled over 21 frames and trained with non-overlapping feature subsequences of that length. The dimensionality of the input features is 120 which is a 40-dim logmel and its Δ and Δ^2 coefficients. Therefore, a batch of size M consists of M 21-frame subsequences and the corresponding tensor is of size $M \times 120 \times 21$. A learner-specific 121×120 affine transform layer is used to transform the input features prior to the first layer of the LSTM. It is initialized to an identity matrix and zero bias vector. The decoding vocabulary comprises 260K words and the language model (LM) is a 4-gram LM with 200M n-grams and modified Kneser-Ney smoothing built using publicly available training data from a broad variety of sources.

Training minimizes cross-entropy loss on 5 K80 GPUs using SGD without momentum. The communication graph among learners is a ring. In each iteration, every learner only communicates with its left and right neighbors. The initial learning rate is 1.0, it is annealed by $1/\sqrt{2}$ after the 20th epoch, and training finishes in 30 epochs. The batch size is 256 21-frame subsequences.

In PCL, the affine transform layer is optimized locally. In each iteration, the local transform is first optimized by one-step SGD. Then, the remaining model parameters are averaged between left and right neighbors and updated by one-step SGD. For optimization of the local transform (i.e., the bottom layer of the model), the initial learning rate is 0.02 and it anneals by $1/\sqrt{2}$ after the 20th epoch, same as the model learning rate schedule. No momentum is used. Due to the page limit, we only plot the training and heldout losses for learner 1 and 4, which are shown in Figure 2. The comparison of the word error rates (WER) on the four test sets is reported in Table 1.

	S1	S2	S3	S4
DSGD [24] (baseline)	22.0	12.6	22.8	21.3
SPD/D^2 [27]	21.9	12.5	22.3	20.5
SPDB (PCL)	**21.4**	**12.4**	**19.9**	**18.7**

Table 1: Word error rates of four test sets.

6. CONCLUDING REMARKS

In this work, we proposed a stochastic primal-dual based optimization framework for decentralized bilevel problems, where the resulting SPDB is a single-loop algorithm that can achieve the same iteration complexity as DSGD for classical nonconvex objective functions. We performed numerical experiments on both toy and practical ASR problems with real datasets. It is shown that our proposed decentralized parameter splitting approach outperforms state-of-the-art algorithms in terms of both accuracy and convergence speed, shedding light on personalized machine learning models over a consensus network.

7. REFERENCES

[1] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. of International Conference on Machine Learning (ICML)*, 2017, pp. 1126–1135.

[2] H. Liu, R. Socher, and C. Xiong, "Taming MAML: Efficient unbiased meta-reinforcement learning," in *Proc. of International Conference on Machine Learning (ICML)*, 2019, pp. 4061–4071.

[3] A. A. Rusu, D. Rao, J. Sygnowski, O. Vinyals, R. Pascanu, S. Osindero, and R. Hadsell, "Meta-learning with latent embedding optimization," in *Proc. of International Conference on Learning Representations (ICLR)*, 2019.

[4] M.-F. Balcan, M. Khodak, and A. Talwalkar, "Provable guarantees for gradient-based meta-learning," in *Proc. of International Conference on Machine Learning (ICML)*, 2019, pp. 424–433.

[5] C. Finn, A. Rajeswaran, S. Kakade, and S. Levine, "Online meta-learning," in *Proc. of International Conference on Machine Learning (ICML)*, 2019, pp. 1920–1930.

[6] Z. Zhuang, Y. Wang, K. Yu, and S. Lu, "No-regret non-convex online meta-learning," in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 3942–3946.

[7] A. Fallah, A. Mokhtari, and A. Ozdaglar, "On the convergence theory of gradient-based model-agnostic meta-learning algorithms," in *Proc. of International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2020, pp. 1082–1092.

[8] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Generalization of model-agnostic meta-learning algorithms: Recurring and unseen tasks," in *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[9] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach," in *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[10] D. A. E. Acar, Y. Zhao, R. Zhu, R. Matas, M. Mattina, P. Whatmough, and V. Saligrama, "Debiasing model updates for improving personalized federated training," in *Proc. of International Conference on Machine Learning (ICML)*, 2021, pp. 21–31.

[11] M. Kayaalp, S. Vlaski, and A. H. Sayed, "Dif-MAML: Decentralized multi-agent meta-learning," *arXiv preprint arXiv:2010.02870*, 2020.

[12] A. Raghu, M. Raghu, S. Bengio, and O. Vinyals, "Rapid learning or feature reuse? towards understanding the effectiveness of MAML," in *Proc. of International Conference on Learning Representations (ICLR)*, 2020.

[13] K. Ji, J. D. Lee, Y. Liang, and H. V. Poor, "Convergence of meta-learning with task-specific adaptation over partial parameters," in *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, 2020, vol. 33, pp. 11490–11500.

[14] S. Ghadimi and M. Wang, "Approximation methods for bilevel programming," *arXiv preprint arXiv:1802.02246*, 2018.

[15] K. Ji, J. Yang, and Y. Liang, "Bilevel optimization: Convergence analysis and enhanced design," in *Proc. of International Conference on Machine Learning (ICML)*, 2021, pp. 4882–4892.

[16] M. Hong, H.-T. Wai, Z. Wang, and Z. Yang, "A two-timescale framework for bilevel optimization: Complexity analysis and application to actor-critic," *arXiv preprint arXiv:2007.05170*, 2020.

[17] T. Chen, Y. Sun, and W. Yin, "A single-timescale stochastic bilevel optimization method," *arXiv preprint arXiv:2102.04671*, 2021.

[18] T. Chen, Y. Sun, and W. Yin, "Tighter analysis of alternating stochastic gradient method for stochastic nested problems," *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[19] P. Khanduri, S. Zeng, M. Hong, H.-T. Wai, Z. Wang, and Z. Yang, "A near-optimal algorithm for stochastic bilevel optimization via double-momentum," *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[20] J. Yang, K. Ji, and Y. Liang, "Provably faster algorithms for bilevel optimization," in *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[21] A. Koppel, B. M. Sadler, and A. Ribeiro, "Proximity without consensus in online multiagent optimization," *IEEE Transactions on Signal Processing*, vol. 65, no. 12, pp. 3062–3077, 2017.

[22] R. Nassif, S. Vlaski, C. Richard, J. Chen, and A. H. Sayed, "Multitask learning over graphs: An approach for distributed, streaming machine learning," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 14–25, 2020.

[23] R. Xin, S. Pu, A. Nedić, and U. A. Khan, "A general framework for decentralized optimization with first-order methods," *Proceedings of the IEEE*, vol. 108, no. 11, pp. 1869–1889, 2020.

[24] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, "Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent," in *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, 2017, vol. 30.

[25] S. Lu, X. Zhang, H. Sun, and M. Hong, "GNSD: a gradient-tracking based nonconvex stochastic algorithm for decentralized optimization," in *Proc. of IEEE Data Science Workshop (DSW)*, 2019, pp. 315–321.

[26] X. Cui, S. Lu, and B. Kingsbury, "Federated acoustic modeling for automatic speech recognition," in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 6748–6752.

[27] H. Tang, X. Lian, M. Yan, C. Zhang, and J. Liu, "D^2: Decentralized training over decentralized data," in *Proc. of International Conference on Machine Learning (ICML)*, 2018, pp. 4848–4856.

[28] S. Lu, J. D. Lee, M. Razaviyayn, and M. Hong, "Linearized ADMM converges to second-order stationary points for non-convex problems," *IEEE Transactions on Signal Processing*, vol. 69, pp. 4859–4874, 2021.

[29] 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.