Distributed Primal-Dual Optimization for Online Multi-Task Learning

Peng Yang, Ping Li
Cognitive Computing Lab
Baidu Research
10900 NE 8th ST, Bellevue WA, 98004, USA
{pengyang01, liping11}@baidu.com

Abstract

Conventional online multi-task learning algorithms suffer from two critical limitations: 1) Heavy communication caused by delivering high velocity of sequential data to a central machine; 2) Expensive runtime complexity for building task relatedness. To address these issues, in this paper we consider a setting where multiple tasks are geographically located in different places, where one task can synchronize data with others to leverage knowledge of related tasks. Specifically, we propose an adaptive primal-dual algorithm, which not only captures task-specific noise in adversarial learning but also carries out a projection-free update with runtime efficiency. Moreover, our model is well-suited to decentralized periodic-connected tasks as it allows the energy-starved or bandwidth-constraint tasks to postpone the update. Theoretical results demonstrate the convergence guarantee of our distributed algorithm with an optimal regret. Empirical results confirm that the proposed model is highly effective on various real-world datasets.

Introduction

Multi-task learning (MTL) is widely used learning framework where similar tasks are considered jointly for the purpose of improving performance compared to learning the tasks separately (Caruana 1997). By transferring information between tasks it is hoped that samples will be better utilized, leading to improved generalization performance. MTL has been successfully applied in practical scenarios, e.g., speech recognition (Seltzer and Droppo 2013), image classification (Lapin, Schiele, and Hein 2014), disease gene prediction (Zhou et al. 2013), etc. Recent years also witness extensive studies on streaming data, known as online multi-task learning (OMTL) (Dekel, Long, and Singer 2006; Saha et al. 2011; Yang, Zhao, and Gao 2017), for the merits of capturing the dynamically changing and uncertain nature of the environment, which is in contrast to the offline setting in which the objective functions are fixed (Liu, Pan, and Ho 2017; Smith et al. 2017).

Existing OMTL techniques suffer from the heavy communication caused by centralizing the high velocity of sequential data from different locations to a single machine. In this paper, we address OMTL problem in a distributed manner, i.e., multiple tasks are geographically located in different places, where task models can synchronize data with others to leverage knowledge of related tasks. In such setting, each task i is endowed with a sequence of objective functions $(f_t^i)_{t=1}^T$, where f_t^i is the loss function of i-th task at round t. The goal boils down to minimizing the sequential objective functions across m different tasks,

$$\min_{\mathbf{W}:\mathbf{W}\in\mathcal{K}} \sum_{t=1}^{T} F_t(\mathbf{W}) \tag{1}$$

where $F_t(\mathbf{W}) = \sum_{i=1}^m f_t^i(\mathbf{w}^i)$ is denoted as the instantaneous loss at round t, while $\mathbf{W} = [\mathbf{w}^1, \ldots, \mathbf{w}^m] \in \mathbb{R}^{d\times m}$ is parameter matrix for m tasks. Note that $\mathcal{K}$ is a closed convex subset characterized by an inequality, i.e., $\mathcal{K} = \{\mathbf{W}|g(\mathbf{W}) \leq 0\}$. It aims to constrain $\mathbf{W}$ into simple sets, e.g., hyperplanes, balls, bound constraints, etc. We assume $m \leq d$ without loss of generality. The algorithm for solving problem (1) has to be distributed, in the sense that each task, accessing only local task data, is required to communicate with other tasks in a bandwidth-limited network. Moreover, the distributed tasks have to learn incrementally from data streams, for the merits of capturing the dynamic nature of the changing environment (Zhang et al. 2018).

The success of multitask learning relies on the relatedness between tasks. To build the task relationship, existing algorithms used low-rank matrices to enforce different tasks to share a common structure (Smith et al. 2017; Yang et al. 2019). The low-rank optimization problem can be solved by the first-order optimization methods, e.g., subgradient descent (Bauschke et al. 2011). Although these methods are guaranteed to converge, they are inefficient because a singular value decomposition (SVD), which takes $O(dm^2)$ time, is required at each round. To reduce the computation complexity, many efficient solvers have been developed by replacing the full SVD with a partial SVD. However, those approaches either require the function to be smooth (Wang, Kolar, and Srebro 2016) or are designed for the constraint optimization (Zheng, Bellet, and Gallinari 2018). Furthermore, the problem that different tasks generally have different noise levels was ignored by those approaches mentioned above. The adversarial noise with large loss residues may corrupt the task relativeness by the conventional convex loss functions. To deal with noise, a calibrated multivariate regression approach was developed in (Liu, Wang, and Zhao 2014), and

then it was further improved in (Gong et al. 2014). Nevertheless, both of them are based on feature learning and the optimization techniques are computationally expensive.

In this work, we propose an efficient distributed algorithm to address both issues simultaneously. The main contributions of this work are summarized as follows:

1. We introduce a capped L_p-norm loss function to capture the adversarial noise. We derive a weighted loss function to iteratively reduce the negative impact of noise according to noise level of specific tasks.
2. The constrained task relatedness is learned by a projection free primal-dual algorithm. In each round, it only needs to compute the leading singular vectors instead of a full SVD, reducing time complexity from $O(dm^2)$ to $O(dm)$.
3. The proposed algorithm is well-suited to decentralized periodic-connected tasks, as it allows the energy-starved or bandwidth-limited tasks to alleviate synchronization delay.
4. Theoretical results demonstrate the convergence guarantee of our distributed model with an optimal regret. Empirical results confirm that the proposed algorithm is effective.

Algorithm

In this problem, we are faced with m different but related classification problems also known as tasks. The task model is learned on a sequence of instance-label pairs, i.e., $\{(\mathbf{x}_t^i, y_t^i)\}_{1\le t\le T}^{1\le i\le m}$, where the instance $\mathbf{x}_t^i \in \mathbb{R}^d$ is drawn from a distinct distribution p^i, and $y_t^i \in \{\pm 1\}$. The algorithm maintains m separate models in parallel, one for each task. When the instances $\{\mathbf{x}_t^1, \ldots, \mathbf{x}_t^m\}$ are observed at round t, the model generates a decision matrix $\mathbf{W}_t = [\mathbf{w}_t^1, \ldots, \mathbf{w}_t^m] \in \mathbb{R}^{d\times m}$ under a constraint set $\mathcal{K}$. Then it suffers the corresponding loss $F_t(\mathbf{W}_t) = \sum_{i=1}^m f_t^i(\mathbf{w}_t^i)$ where f_t^i is a convex loss function. The goal of online learner is to generate a sequence of decision points $\{\mathbf{W}_t\}_{t=1}^T$, so that the regret regarding to the best fixed decision can be minimized,

$$\mathrm{Reg}_T := \sum_{t=1}^T F_t(\mathbf{W}_t) - \sum_{t=1}^T F_t(\mathbf{W}^*), \tag{2}$$

where $\mathbf{W}^* = \mathrm{argmin}_{\mathbf{W}\in\mathcal{K}} \sum_{t=1}^T F_t(\mathbf{W})$ is the best decision in hindsight. An algorithm achieves nontrivial performance if its regret is sublinear over the number of total rounds T.

The success of multi-task learning relies on the relatedness between tasks. To learn the task relationship, existing algorithms exploit low-rank constraints to enforce different tasks to share a common structure (Smith et al. 2017; Xie et al. 2017; Baytas et al. 2016). To yield a low-rank solution in (2), these methods aim to minimize the following constrained problem:

$$\min_{\mathbf{W}} \sum_{t=1}^T \sum_{i=1}^m f_t^i(\mathbf{w}^i), \ \text{ s.t. } \mathrm{rank}(\mathbf{W}) \le r$$

where r is a predefined value with $r \ll \min(d, m)$, and $\mathrm{rank}(\cdot)$ denotes the matrix rank, i.e., the number of nonzero singular values. Note that the constrained objective is equivalent to the regularized objective function with a proper parameter $\lambda > 0$,

$$\min_{\mathbf{W}} \sum_{t=1}^T \sum_{i=1}^m f_t^i(\mathbf{w}^i) + \lambda\, \mathrm{rank}(\mathbf{W}). \tag{3}$$

Although above problems are equivalent, specific optimization techniques could be more suitable for one particular type of objective functions[1]. For convenience, we won't distinguish between these two formulations in this work.

In this paper, we make the following assumptions:

- The loss function $f_t(\mathbf{w})$ is convex, i.e., $\forall\ \mathbf{w}, \mathbf{w}'$ in the domain of f_t, $f_t(\mathbf{w}) \ge f_t(\mathbf{w}') + \langle \nabla f_t(\mathbf{w}'), \mathbf{w} - \mathbf{w}' \rangle$.
- The loss function $f_t(\mathbf{w})$ is β-*Lipschitz* on a convex set, i.e., $\forall\ \mathbf{w}, \mathbf{w}'$ in the domain of f_t, $|f_t(\mathbf{w}) - f_t(\mathbf{w}')| \le \beta \|\mathbf{w} - \mathbf{w}'\|_2$.
- The concave function $h(u) = \min(u^p, \xi)$ $(\xi > 0)$ has a bounded supergradient at any point $u = f(\mathbf{w})$ with $p \in (0, 1)$, i.e., $\|\nabla_u h(u)\|_2 \le \kappa$.
- Euclidean diameter of primal variable $\mathbf{w}$ or dual variable $\mathbf{a}$ is bounded by D, i.e., $\|\mathbf{w} - \mathbf{w}'\|_2 \le D$ and $\|\mathbf{a} - \mathbf{a}'\|_2 \le D$.

Adversarial Learning

In adversarial learning, the feedback observed by the learner is malicious inputs designed to fool machine learning models. Adversarial noise with large loss residues may corrupt task relativeness by the conventional convex loss f_t. To be resistant to noise, each task should have a specific regularization parameter that depends on the specific noise level. To achieve this goal, we exploit a capped L_p-norm function with $p \in (0, 1)$:

$$h(f_t^i(\mathbf{w}^i)) = \min\left(f_t^i(\mathbf{w}^i)^p, \xi\right), \tag{4}$$

where $h(f_t^i(\cdot))$ enforces a capped L_p-norm over loss function f_t^i with an upper bound $\xi > 0$. It indicates that no matter how misclassified the data point is, the loss residue in (4) is capped by ξ. This makes the loss function robust to noise since their effect to the model is bounded. However, optimizing this problem is difficult since the function $\min(f_t^i(\cdot)^p, \xi)$ is a concave non-smooth function in the domain of f_t^i. Motivated by concave duality (Rockafellar 1970), Lemma 1 provides an iterative weighted function to solve it.

Lemma 1. *Problem (4) can be relaxed to minimizing a weighted convex formulation:*

$$\min_{\mathbf{w}^i} \gamma_t^i f_t^i(\mathbf{w}^i), \tag{5}$$

where $\gamma_t^i = \nabla_u h(u)|_{u=f_t^i(\mathbf{w})}$ *is the supergradient of the concave function* $h(u)$ *at the point* $u = f_t^i(\mathbf{w}_t^i)$,

$$\gamma_t^i = \begin{cases} p f_t^i(\mathbf{w}_t^i)^{p-1}, & f_t^i(\mathbf{w}_t^i)^p \le \xi \\ 0, & \textit{otherwise} \end{cases} \tag{6}$$

[1] Alternating direction method of multipliers for regularized objective function and frank-wolfe for constrained objective function. Gradient descent methods can be adopted for both, leads to proximal and projected methods, respectively.

Proof. Let $u = f(\cdot)$ and $h(u) = \min(u^p, \xi)$. Since $f(\cdot)$ is a convex function, $h(u)$ can be formulated as:

$$\min(u^p, \xi) = \inf_{\gamma \geq 0} [\gamma u - h^*(\gamma)], \tag{7}$$

where $h^*(\gamma)$ is the concave dual of $h(u)$, defined as:

$$h^*(\gamma) = \inf_{u>0} (\gamma u - h(u)) \overset{(7)}{=} \inf_{u>0} [\gamma u - \min(u^p, \xi)]$$
$$= \begin{cases} \frac{p-1}{p} p^{\frac{1}{1-p}} \gamma^{\frac{1-p}{p}}, & \text{if } u^p < \xi \\ \gamma \xi^{\frac{1}{p}} - \xi, & \text{if } u^p \geq \xi. \end{cases}$$

Equipped $h^*(\gamma)$ back to Eq. (7), we obtain γ as in Eq. (6). □

Remark 1. We observe that γ_t^i depends on the loss $f_t^i(\mathbf{w}_t^i)$. In particular, a misclassified point with $f_t^i(\mathbf{w}_t^i)^p > \xi$ will be considered as an outlier, and ignored, i.e., $\gamma_t^i = 0$.

Remark 2. The derived solution in (5) minimizes the upper bound of the concave problem (4) iteratively. Since $h(f_t(\mathbf{w}^i))$ is a concave function, for any $\mathbf{w}^i$ we obtain an upper bound of $h(f_t^i(\mathbf{w}^i))$ via a linear approximation,

$$h(f_t^i(\mathbf{w}^i)) \leq h(f_t^i(\mathbf{w}_t^i)) + \langle \gamma_t^i, f_t^i(\mathbf{w}^i) - f_t^i(\mathbf{w}_t^i) \rangle,$$

where $\gamma_t^i = \nabla_u h(u)|_{u=f_t^i(\mathbf{w}_t^i)}$. Since $f_t(\mathbf{w}_t^i)$ is constant, $\min_{\mathbf{w}^i} h(f_t^i(\mathbf{w}_t^i)) + \langle \gamma_t^i, f_t^i(\mathbf{w}^i) - f_t^i(\mathbf{w}_t^i) \rangle \equiv \min_{\mathbf{w}^i} \langle \gamma_t^i, f_t^i(\mathbf{w}^i) \rangle$, which obtains a convex loss to minimize the upper bound of $h(f_t^i(\mathbf{w}^i))$.

Projection-free Optimization

The refined problem, $\sum_{t=1}^T \sum_{i=1}^m \langle \gamma_t^i, f_t^i(\mathbf{w}^i) \rangle + \lambda \text{rank}(\mathbf{W})$, is non-convex and computationally intractable (Amaldi and Kann 1998). We relax rank$(\cdot)$ to its convex surrogate, i.e., nuclear norm $\|\cdot\|_*$, then the problem becomes

$$\min_{\mathbf{W}} \sum_{t=1}^T \sum_{i=1}^m \langle \gamma_t^i, f_t^i(\mathbf{w}^i) \rangle + \lambda \|\mathbf{W}\|_*. \tag{8}$$

The nuclear norm minimization can be solved by gradient descent and proximal gradient descent. Although these methods are guaranteed to converge, they have to perform a full SVD of $\mathbf{W}_t$ in each round, which suffers a hight runtime complexity of $O(dm^2)$. (Hazan and Kale 2012) provided a linear optimization method to solve this issue, but its computational effectiveness is achieved at the expense of a suboptimal regret. To reduce runtime complexity, we study the dual form of the nuclear norm, $\|\mathbf{W}\|_* = \max_{\|\mathbf{A}\|_2 \leq 1} \text{tr}(\mathbf{A}^\top \mathbf{W})$ where $\|\cdot\|_2$ is the spectral norm, and then cast the problem (8) into the following primal-dual formulation:

$$\min_{\mathbf{W}} \max_{\mathbf{A}} \sum_{t=1}^T \sum_{i=1}^m \langle \gamma_t^i, f_t^i(\mathbf{w}^i) \rangle + \lambda \text{tr}(\mathbf{A}^\top \mathbf{W}) \text{ s.t. } \|\mathbf{A}\|_2 \leq 1.$$

Since the above optimization problem is convex-concave, we can apply the online subgradient method to solve it. However, due to the spectral norm constraint of $\mathbf{A}$, we have to project the intermediate solution onto the unit spectral norm ball, which again requires a full SVD operation (Xiao et al. 2017).

To address this issue, we replace the constraint $\|\mathbf{A}\|_2 \leq 1$ with a regularization term to control the spectral norm of $\mathbf{A}$,

$$\min_{\mathbf{W}} \max_{\mathbf{A}} \sum_{t=1}^T \sum_{i=1}^m \gamma_t^i f_t^i(\mathbf{w}^i) + \lambda \text{tr}(\mathbf{A}^\top \mathbf{W}) - \rho [\|\mathbf{A}\|_2 - 1]_+,$$

where $\rho > 0$ is a trade-off parameter and $[\cdot]_+ = \max(0, \cdot)$. We assign $\rho = 1$, $\lambda = 1$ since such setting can control the rank, i.e., $\|\mathbf{W}\|_* \leq \rho/\lambda$. To solve the above problem, we can use the online subgradient method (Shalev-Shwartz 2012), which iterates as follows:

$$\mathbf{A}_{t+1} = \mathbf{A}_t + \eta_t \left(\mathbf{W}_t - \partial [\|\mathbf{A}_t\|_2 - 1]_+ \right),$$
$$\mathbf{W}_{t+1} = \mathbf{W}_t - \eta_t \left(\mathbf{A}_{t+1} + \nabla F_t(\mathbf{W}_t) \Gamma_t \right),$$

where $\Gamma_t = \text{diag}(\gamma_t^1, \ldots, \gamma_t^m) \in \mathbb{R}^{m \times m}$.

Note that the subgradient $\partial [\|\mathbf{A}\|_2 - 1]_+$ can be computed efficiently. We denote $\sigma_1(\mathbf{A})$ as the leading singular value of $\mathbf{A}$, $\mathbf{u}$ and $\mathbf{v}$ as the corresponding left and right singular vectors. Then we have

$$\mathbf{u}\mathbf{v}^\top \mathbb{I}(\sigma_1(\mathbf{A}) > 1) \in \partial [\|\mathbf{A}\|_2 - 1]_+.$$

In each round, we only need to compute the leading singular vector of $\mathbf{A}_t$ with $O(dm)$ time. In contrast, a full SVD takes $O(dm^2)$ time.

Distributed Learning

Though above algorithm is efficient, heavy communication is caused by centralizing the high velocity of sequential data to a central machine. To address this issue, we show how to perform the primal-dual optimization in a distributed manner.

Assume that tasks are distributed on local worker machines, i.e., one worker for each task, our core idea is to solve the *local problem* on each local machine independently, and then centralize the updated information of each task to efficiently solve a *central problem*. The proposed algorithm, namely DROM, is summarized in Algorithm 1. It runs an alternating optimization procedure that comprises two steps: 1) Local-step: solving $\{\mathbf{w}^i, \mathbf{a}^i\}_{i=1}^m$ in a distributed manner among local workers independently; 2) Central-step: solving $\partial [\|\mathbf{A}\|_2 - 1]_+$ with aggregated $\{\mathbf{a}^i\}_{i=1}^m$ from all workers on central server. Figure 1 illustrates the procedure of DROM.

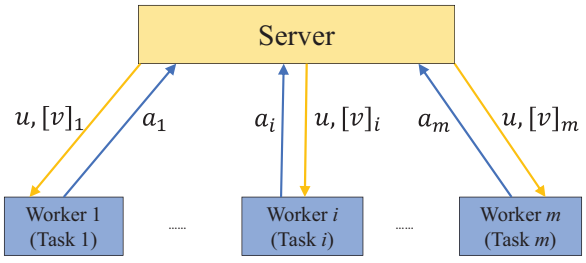

Figure 1: Distributed Primal-Dual Optimization

Here we elaborate on the details of DROM:

- **Model Variables**: At round t, the m workers have primal variables $\{\mathbf{w}_t^1, ..., \mathbf{w}_t^m\}$ for m tasks, respectively. In addition, m dual variables $\{\mathbf{a}_t^1, ..., \mathbf{a}_t^m\}$ are stored at each

Table 1: Comparison with distributed variants of proximal gradient descent (ProxGD), alternating direction method of multipliers (ADMM), online frank-wolfe (OFW) in terms of computational complexity and communication efficiency.

Algorithms	Worker Comp.	Server Comp.	Communication	Time Complexity	Regret
ProxGD	Gradient Comp.	SV Shrinkage	$2d$	m^2d	$T^{1/2}$
ADMM	ERM	SV Shrinkage	$3d$	m^2d	$T^{1/2}$
OFW	Gradient Comp.	Leading SV Comp.	$2d$	md	$T^{3/4}$
DROM	Gradient Comp.	Leading SV Comp.	$2d$	md	$T^{1/2}$

Algorithm 1 DROM: Distributed Primal-dual optimization for Online MTL

Input: data $\{\mathbf{x}_t^i, y_t^i\}$ with $i \in [m]$ and $t \in [T]$ distributed over m machines, parameters p and ξ
Initialize: $\mathbf{w}_0^i = \mathbf{0}$, $\mathbf{a}_0^i = \mathbf{0}$ for all workers $i \in [m]$
for $t = 1, \ldots, T$ **do**
  for all workers (Local-step) do in parallel
    $$\gamma_t^i = \begin{cases} pf_t^i(\mathbf{w}_t^i)^{p-1}, & f_t^i(\mathbf{w}_t^i)^p \le \xi \\ 0, & \text{otherwise} \end{cases}$$
    When $\gamma_t^i > 0$, local update with $\eta_t = \frac{1}{\sqrt{t}}$:
    $$\begin{aligned} \mathbf{a}_{t+1}^i &= \mathbf{a}_t^i + \eta_t \left(\mathbf{w}_t^i - [\mathbf{u}\mathbf{v}^\top]_i\right); \\ \mathbf{w}_{t+1}^i &= \mathbf{w}_t^i - \eta_t \left(\mathbf{a}_{t+1}^i + \gamma_t^i \nabla f_t^i(\mathbf{w}_t^i)\right); \end{aligned} \tag{9}$$
    send $\mathbf{a}_{t+1}^i$ to the server, wait to receive $\{\mathbf{u}, [\mathbf{v}]_i\}$;
  Reduce (Central-step): The server aggregates $\mathbf{A}$ and computes $\mathbf{u}\mathbf{v}^\top \mathbb{I}(\sigma_1(\mathbf{A}) > 1) \in \partial[\|\mathbf{A}\|_2 - 1]_+$;
  Server sends back $\{\mathbf{u}, \mathbf{v}\}$ when $\sigma_1(\mathbf{A}) > 1$;
end for
Output: $\mathbf{W}_T$

worker. Each worker accesses local task data and updates variables independently.

- **Local Update**: The primal-dual optimization is conducted in a distributed manner. At round t, each worker i is assigned a problem that accesses local data $(\mathbf{x}_t^i, y_t^i)$. Local problem is solved in two steps: 1) Computing the weight γ_t^i based on $f_t^i(\mathbf{w}_t^i)$; 2) Performing an alternative learning procedure as in (9): optimizing $\mathbf{a}^i$ with vector $[\mathbf{u}\mathbf{v}^\top]_i$; optimizing $\mathbf{w}^i$ with gradient $\nabla f_t^i(\mathbf{w}_t^i)$. Note that synchronization with the server is not allowed if noise or outlier is identified, i.e., $\gamma_t^i = 0$.
- **Central Update**: When the local update ends, the worker i sends $\mathbf{a}_{t+1}^i$ to the central server. As we know that $\mathbf{u}\mathbf{v}^\top \mathbb{I}(\sigma_1(\mathbf{A}) > 1) \in \partial[\|\mathbf{A}\|_2 - 1]_+$, the server aggregates the local updates on $\mathbf{A}$ from all workers to calculate $(\mathbf{u}, \mathbf{v})$, and then sends back $\mathbf{u}[\mathbf{v}]_i$ to the corresponding worker i. It is efficient for central computing with $O(dm)$ time. Note that the server sends back $(\mathbf{u}, \mathbf{v})$ only when the corresponding spectrum $\sigma_1(\mathbf{A}) > 1$, which alleviates communication cost per round.

Motivated by the analysis in (Xiao et al. 2017), we provide the theoretical guarantee for this distributed algorithm regarding the regret. The regret is based on the function f. Recall that minimizing the (5) with $f(\cdot)$ is to minimize the upper bound of the (4) with $h(f(\cdot))$. When $f(\cdot)$ is converged to the optimal points, it infers an optimal upper bound for $h(f(\cdot))$.

Theorem 1. *For all $t > 1$, the algorithm DROM runs over arbitrary instance-label pairs $\{(\mathbf{x}_t^i, y_t^i)\}_{i=1}^m$ with the update rule (9). Assume $\mathbf{A}_t^* = \operatorname{argmax}_{\|\mathbf{A}\|_2 \le 1} tr(\mathbf{A}^\top \mathbf{W}_t)$ and $\|\mathbf{W}_t\|_* \le \rho/\lambda$ satisfied at all $t > 1$. When $\eta_t = 1/\sqrt{t}$ the following regret is hold*

$$R_T \le m\sqrt{T}\left(D^2 + (\kappa\beta + \lambda D)^2 + (\rho + \lambda D)^2\right).$$

Remark 3. *The above theorem implies that the proposed algorithm is in the order of $O(\sqrt{T})$. This order is optimal, since the objective function is not strongly convex.*

Table 1 compares DROM with state-of-the-art baselines, e.g., Proximal Gradient Descent (ProxGD) (Duchi et al. 2010), Alternating Direction Method of Multipliers (ADMM) (Boyd et al. 2011) and Online Frank-Wolfe (OFW) (Hazan and Kale 2012), in terms of runtime complexity and regret. From that table, we observe that DROM achieves a lower computational complexity with an optimal regret. Note that our method is different from OFW, since DROM directly constrains model parameters while OFW constrains the gradient descent. This work is different from ProxGD as well, since DROM is primal-dual algorithm while ProxGD optimizes only the primal variable.

Decentralized Periodic Communication

The algorithm DROM requires a central server to synchronize with all workers. This limits its applications in practical scenarios where each worker can only connect with its local neighbors in a bandwidth-limited network.

For this reason, we propose a variant of DROM for decentralized periodic-connected tasks and summarize the whole procedure, namely DROM-D, in Algorithm 2. This algorithm is parameterized by $\mathcal{P}(\mathbf{S}, \tau)$, where $\mathbf{S} \in \mathbb{R}^{m \times m}$ is an adjacency matrix used for inter-worker communication, and $\tau > 0$ is a synchronous interval for periodic update. These parameters improve the communication-efficiency in three different ways:

- **Group Synchronization**: The learning process does not rely on a fusion center or network-wide communication. Instead of synchronizing with all workers, a local worker just needs to exchange information with its neighbors, where the network topology is captured by the weight matrix $\mathbf{S}$. Therefore, using a sparse weight matrix $\mathbf{S}$ reduces the overall communication cost per round. Specifically, each

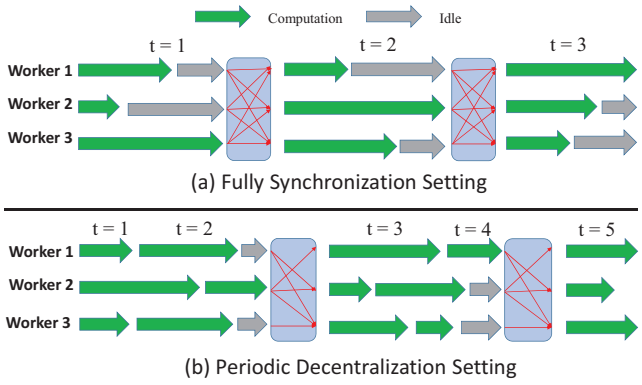

Figure 2: Illustration of communication-reduction strategies for $\tau = 2$. Green, red, grey arrows represent gradient computation, communication, and idle state, respectively.

worker i becomes a "local server", and aggregates $\mathbf{A}^{(i)}$ from its neighbors,

$$\mathbf{A}^{(i)} = \mathbf{A} \times \text{Diag}([\mathbf{S}]_i), \quad [\mathbf{A}^{(i)}]_j = \begin{cases} \mathbf{a}^j, & j \in \mathcal{N}_i \\ \mathbf{0}, & j \notin \mathcal{N}_i \end{cases}$$

with $\mathcal{N}_i = \{j \mid S_{ij} = 1\}$ as the neighbors of the worker i.

- **Periodic Optimization**: The synchronization delay time is amortized over τ synchronous interval and is τ times smaller than fully synchronous update. Moreover, periodic optimization alleviates the synchronization delay in waiting for slow workers. Observe in Figure 2 that the idle time of workers is significantly reduced. To capture the synchronous interval, we use a time-varying matrix $\mathbf{S}_t$ that varies as:

$$\mathbf{S}_t = \begin{cases} \mathbf{S}, & (t \bmod \tau) = 0 \\ \mathbf{I}_{m \times m}, & \text{otherwise} \end{cases}$$

where the identity matrix $\mathbf{I}_{m \times m}$ means that there is no iter-worker communication during the τ local updates.

- **Non-blocking Execution**: As local update of $\mathbf{A}$ does not learn the gradient, the singular vector $\{\mathbf{u}, \mathbf{v}\}$ remains the same while worker nodes conduct local updates, i.e., $\{\mathbf{u}, \mathbf{v}\}_t = \{\mathbf{u}, \mathbf{v}\}_{t-1} = \ldots = \{\mathbf{u}, \mathbf{v}\}_{t-\tau+1}$ for $(t \bmod \tau) = 0$. Note that the workers only need $\{\mathbf{u}, \mathbf{v}\}_{t-\tau+1}$ before dual variable is updated from $\mathbf{A}_t$ to $\mathbf{A}_{t+1}$. Thus, there is no synchronous update until the workers perform next τ rounds of local updates, which reduces synchronization delay.

Remark 4. We study the update rule for existing synchronized algorithms since full synchronous algorithm corresponds to the special case $\mathbf{S} = \mathbf{J} = \mathbf{1}\mathbf{1}^\top, \tau = 1$. We show how existing communication-efficient algorithms are special cases of the general decentralized framework $\mathcal{P}(\mathbf{S}, \tau)$:

- *Fully Synchronization* $\mathcal{P}(\mathbf{J}, 1)$*:* The local models are synchronized with all other workers after every round.
- *Periodic Synchronization* $\mathcal{P}(\mathbf{J}, \tau)$*:* The local models are synchronized with all other workers after every τ rounds.

Algorithm 2 DROM-D: The DROM algorithm in Decentralized Periodic setting

Input: $\{\mathbf{x}_t^i, y_t^i\}$ with $i \in [m]$ and $t \in [T]$, the metrics $\mathcal{P}(\mathbf{S}, \tau)$, parameters p and ξ
Initialize: $\mathbf{w}_0^i = \mathbf{0}$, $\mathbf{a}_0^i = \mathbf{0}$ for all workers $i \in [m]$
for $t = 1, \ldots, T$ **do**
  for all workers: $i = 1, \ldots, m$ **in parallel do**
    Solve **local problem** with $\eta_t = \frac{1}{\sqrt{\lceil t/\tau \rceil}}$:

$$\gamma_t^i = \begin{cases} p f_t^i(\mathbf{w}_t^i)^{p-1}, & f_t^i(\mathbf{w}_t^i)^p \le \xi \\ 0, & \text{otherwise} \end{cases}$$

$$\mathbf{w}_{t+1}^i = \mathbf{w}_t^i - \eta_t \left(\mathbf{a}_t^i + \gamma_t^i \nabla f_t^i(\mathbf{w}_t^i)\right);$$

$$\mathbf{a}_{t+1}^i = \mathbf{a}_t^i + \eta_t \left(\mathbf{w}_{t+1}^i - [\mathbf{u}\mathbf{v}^\top]_i\right);$$

  If t **mod** $\tau = 0$ **do central problem**:
    Broadcast $\mathbf{a}_{t+1}^i$ to its neighbors;
    Wait to receive $\mathbf{a}_{t+1}^j$ from task $j \in \mathcal{N}_i$;
    Aggregate $\mathbf{A}^{(i)} = \mathbf{A}_{t+1} \times \text{Diag}([\mathbf{S}]_i)$:
    $\mathbf{u}\mathbf{v}^\top \mathbb{I}(\sigma_1(\mathbf{A}^{(i)}) > 1) \in \partial[\|\mathbf{A}^{(i)}\|_2 - 1]_+$;
end for
Output: $\mathbf{W}_T$

- *Periodic decentralization* $\mathcal{P}(\mathbf{S}, \tau)$*:* The matrix $\mathbf{S}$ is fixed as a sparse weight matrix. Local model is updated via aggregating with few neighbors after every τ rounds.

Below provides theoretical guarantee of the decentralized periodic algorithm DROM-D regarding the regret.

Theorem 2. *The algorithm DROM-D runs over arbitrary sequential instance-label pairs. Assume that $\tau \ge 1$ and $\mathbf{S} \in \mathbb{R}^{m \times m}$ is a random matrix with $S_{ij} \in [0, 1]$. Let $\mathbf{A}_t^* = \text{argmax}_{\|\mathbf{A}\|_2 \le 1} tr(\mathbf{A}^\top \mathbf{W}_t)$ and $\|\mathbf{W}_t\|_* \le \rho/\lambda$ are satisfied on any $t > 1$. When $\eta_t = 1/\sqrt{\lceil t/\tau \rceil}$, the regret holds,*

$$R_T \le \sqrt{T} m \tau^{3/2} \left((D/\tau)^2 + (\kappa\beta + \lambda D)^2 + (\lambda D + \rho)^2\right).$$

Remark 5. *The regret is affected by the parameters λ and ρ that are related to task structure since the regret is hold when $\|\mathbf{W}\|_* \le \rho/\lambda$. Assume that $D \le 1$. If all tasks are identical, we have $\|\mathbf{W}\|_* = 1$, then regret becomes $\mathcal{O}(m\sqrt{T}\lambda^2\tau^{3/2})$ due to $\rho = \lambda$. If tasks are independent and unrelated with others, i.e., $\|\mathbf{W}\|_* = m$ leads to $\rho = m\lambda$, then regret becomes $O(m^3\sqrt{T}\lambda^2\tau^{3/2})$. It infers that a low-rank task structure yields to a small regret.*

Table 2: Description of the datasets

	Spam Email	MHC-I	EachMovie
#Tasks	4	12	30
#Sample	7,068	18,664	6,000
#Dimesion	1,458	400	1,783
#MaxSample	4,129	3,793	200
#MinSample	710	415	200

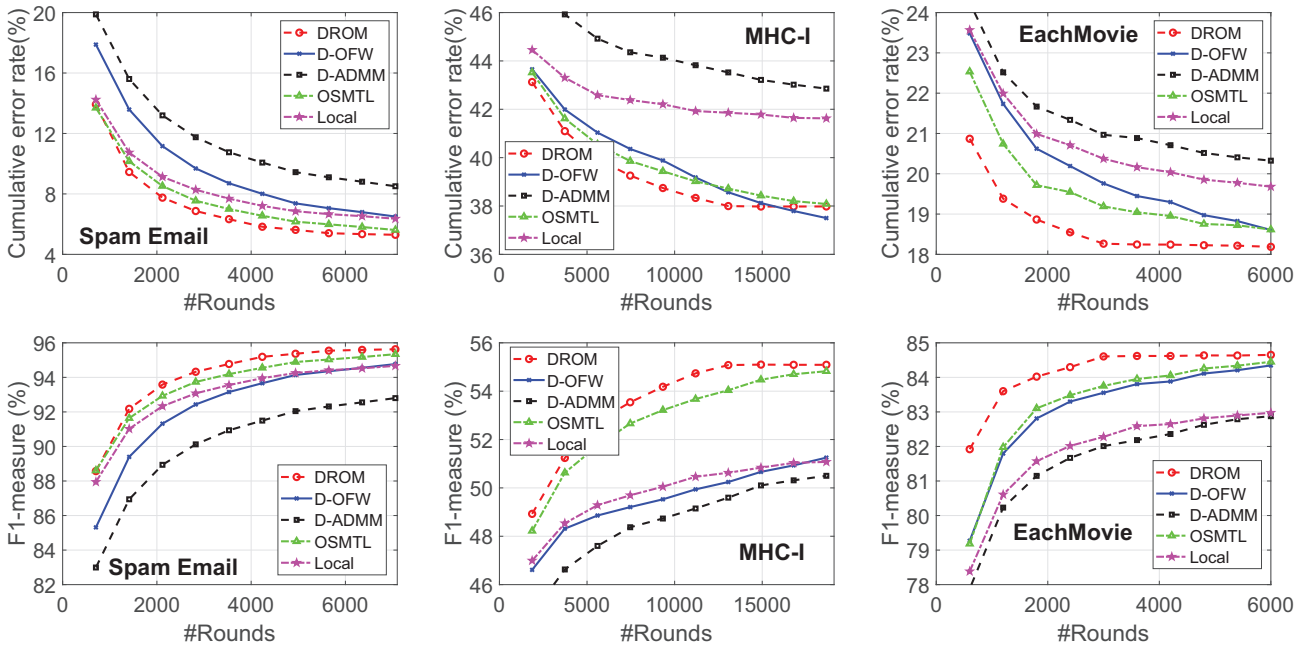

Figure 3: Cumulative error rate and F1-measure along online learning process

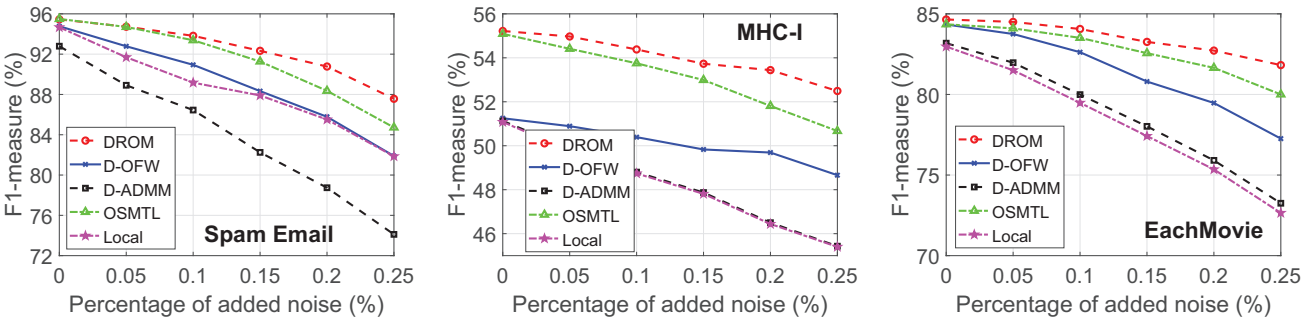

Figure 4: Classification accuracy performance under adversarial noisy data

Experiments

Empirical experiments are conducted to evaluate the algorithms on three datasets used in previous work (Zhang et al. 2018). Table 2 summarizes the statistics of the datasets.

Spam Email[2] contains 7,068 emails collected from mailboxes of 4 users (i.e., 4 tasks). Each mail entry is represented by a word document vector via the TF-IDF conversion technique. A classifier is proposed to classify each incoming email into two categories: *legitimate* or *spam* for each user.

MHC-I[3], a bio-marker dataset, contains 18,664 peptide sequences for 12 MHC-I molecules (i.e., 12 tasks). Each peptide sequence is converted to a 400 dimensional feature vector (Li et al. 2011). The learner aims to classify whether a peptide sequence is *binder* or *non-binder* for each MHC-I molecule. Recent work has demonstrated that the shared knowledge among related molecules (tasks) can be leveraged to improve the MHC-I binding prediction.

EachMovie[4] is a movie recommendation dataset where 72,916 users rate a subset of 1,628 movies. It randomly prioritizes 6,000 user-rating pairs spanning 30 users and 200 movies. The ratings (i.e. $[1, 6]$) are converted into *like* or *dislike*, based on the rating order. For each movie, we randomly select 1,783 users who viewed that movie and use their ratings as its features. Finally, we obtain 200 instances (1,783 features) for each of 30 tasks.

Baselines and Evaluation Metrics

We compare our method with four baselines: 1) **Local**, where each task learns a model locally on its own data. 2) **Smoothed OMTL** (OSMTL) (Murugesan et al. 2016) jointly learns the per-task hypothesis and the inter-task relationships in an

[2]http://labs-repos.iit.demokritos.gr/skel/i-config/

[3]http://web.cs.iastate.edu/ honavar/ailab/

[4]http://goldberg.berkeley.edu/jester-data/

Table 3: Sensitivity study on the parameters τ and ζ

Parameter Setting	Spam Email		MHC-I		EachMovie	
	Error Rate	F1-measure	Error Rate	F1-measure	Error Rate	F1-measure
$\tau = 1, \zeta = 0$	5.31 (2.14)	95.63 (1.75)	38.05 (0.31)	55.11 (0.49)	18.13 (6.54)	84.61 (8.36)
$\tau = 1, \zeta = 0.5$	5.52 (3.32)	95.32 (1.64)	38.61 (1.21)	54.51 (1.51)	18.90 (6.12)	83.22 (8.77)
$\tau = 1, \zeta = 0.9$	6.36 (0.64)	94.67 (0.54)	41.62 (3.95)	51.08 (6.23)	19.78 (7.39)	82.97 (9.35)
$\tau = 20, \zeta = 0$	5.32 (2.16)	95.63 (1.76)	38.13 (0.21)	55.05 (0.33)	18.27 (6.67)	84.51 (8.45)
$\tau = 20, \zeta = 0.5$	5.67 (3.12)	95.03 (1.98)	38.99 (0.54)	54.11 (1.35)	19.35 (5.39)	83.07 (8.99)
$\tau = 20, \zeta = 0.9$	6.53 (0.43)	94.29 (0.36)	41.93 (1.97)	50.81 (1.23)	19.86 (5.32)	82.91 (7.33)

online setting. 3) Two distributed optimization methods for regularized online multi-task learning: Online Alternating Direction Method of Multipliers (**D-ADMM**) (Matamoros 2017) and Online Frank-Wolfe (**D-OFW**) (Zhang et al. 2017). We adapt two algorithms into distributed multi-task setting, and provide corresponding implementations in Supporting Materials. To handle with online data, we modify the offline setting of ADMM by retaining online data after observing one example. All parameters of the baselines are tuned according to their recommended instructions. DROM and DORM-D are the proposed distributed algorithms. For both methods, we simply set $\lambda = 1, \rho = 1$ to avoid overfitting, and tune $p \in (0, 1)$ with $\xi = 1$ to deal with adversarial noise.

There are no good ways of unitizing network when prior knowledge of tasks is unknown. Generally speaking, there are three different types of networks: *full-connected* ($\zeta = 0$), *rid-connected* ($\zeta = 0.5$) and *ring-connected* ($\zeta = 0.9$) network, used to examine the impact of adjacency matrix $\mathbf{S}$, where $\zeta = \max(|\sigma_2(\mathbf{S})|, |\sigma_m(\mathbf{S})|)$ is the second largest absolute eigenvalue of $\mathbf{S}$. Specifically, $S_{ij} = 1$ indicates a connection between task i and task j; $S_{ij} = 0$ otherwise. Moreover, there are two types of synchronization, *fully synchronization* ($\tau = 1$) and *periodic synchronization* (e.g., $\tau = 20$) after every (e.g.,) 20 rounds.

We evaluate the performance using two measurements: 1) **cumulative error rate**, ratio of predicted errors over online data, reflecting the prediction accuracy of online learning; 2) **F1-measure**, the harmonic mean of precision and recall, evaluating the performance of classification model. Number of iteration (trial) is used to reflect the convergence of online algorithms (Zhang et al. 2018), which is different from offline setting with CPU time (Smith et al. 2017). For error rate, the smaller the measures, the better the performance of an algorithm; For F1-measure, a higher value means a better performance. To compare these algorithms fairly, we randomly shuffle the ordering of samples in each dataset. We repeat each experiment 10 times and report the averaged results.

Comparison Result

Evaluation measures versus running rounds of online learning is plotted in Figure 3. The results illustrate the following:

- Among all the baselines, DROM achieves a lower error rate and a higher F1 score on most measures.
- The improvement of our algorithm over the baselines is significant. As can be seen, our method converges faster than other baselines. This is expected as DROM achieves an optimal regret with an efficient runtime complexity.
- Although D-OFW has a higher order of regret, it practically obtains a better result than strong baselines.
- Nuclear norm regularization boosts the prediction performance over plain single task learning significantly, which infers the effectiveness of leveraging the shared knowledge in multi-task learning.

To evaluate the robustness of the algorithms, we randomly impose adversarial noisy labels with a probability from 5% to 25%. Figure 4 presents the evaluation measures of the algorithms on various noisy levels. We observe that DROM consistently outperforms other methods over various levels of noise data. This shows the clear advantage of developing robust loss functions on adversarial learning scenario.

Table 4: Run-time (sec) of each iteration for each algorithm

Algorithm	Spam Email	MHC-I	EachMovie
Local	0.53	0.76	1.14
D-OFW	1.16	1.50	2.33
D-ADMM	1.92	3.35	4.01
DROM	1.26	1.59	2.25

We evaluate these algorithms with runtime cost in Table 4. It can be observed that DROM runs faster than D-ADMM. The reason should be obvious as D-ADMM has to perform SVD in each round, while DROM computes only the leading singular vectors. DROM is relatively slower than Local, which is expected since DROM has to learn the structure of task relativeness. However, the extra computational cost is worth it as learning multiple tasks jointly can significantly improve the prediction performance.

Sensitivity study on the parameters τ and ζ

We conduct sensitivity analysis on the parameters τ and ζ. A high value of τ or ζ would reduce inter-worker communication, which gradually leads to independent learning on local tasks. Specifically, we set τ to $\{1, 20\}$ and ζ to $\{0, 0.5, 0.9\}$, and evaluate DROM-D in various $\mathcal{P}(\mathbf{S}, \tau)$. The comparison result is shown in Table 3. We observe that either increasing a value of τ or ζ would degrade the performance. In a fully-connected setting (ζ = 0), large synchronous interval ($\tau = 20$) is tolerant since the workers can interact with others to leverage the task relativeness. In a sparse-connected

network ($\zeta > 0$), frequent synchronization ($\tau = 1$) is preferable since it can accelerate propagation of information between the tasks. To achieve a balance, we choose $\tau = 20$ in fully-connected tasks in our experiment since the algorithm achieves a good accuracy with a low cost of communication.

Conclusion

This paper studies distributed primal-dual adaptive optimization for online multi-task learning. Specifically, we propose an adaptive projection-free algorithm with optimal regret and computational efficiency. Furthermore, the proposed algorithm is well-adapted in decentralized periodic-connected network with theoretical analysis based on task relatedness. We evaluate the efficacy of the proposed algorithm on three real-world datasets for multi-task classification and find out it runs significantly faster than the counterpart algorithms with projection. The theoretical results regarding the robust learning on adversarial noise have also been verified.

References

Amaldi, E., and Kann, V. 1998. On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems. *Theoretical Computer Science* 209(1-2):237–260.

Bauschke, H. H.; Burachik, R. S.; Combettes, P. L.; Elser, V.; Luke, D. R.; and Wolkowicz, H. 2011. *Fixed-point algorithms for inverse problems in science and engineering*, volume 49. Springer Science & Business Media.

Baytas, I. M.; Yan, M.; Jain, A. K.; and Zhou, J. 2016. Asynchronous multi-task learning. In *IEEE 16th International Conference on Data Mining (ICDM)*, 11–20.

Boyd, S.; Parikh, N.; Chu, E.; Peleato, B.; and Eckstein, J. 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning* 3(1):1–122.

Caruana, R. 1997. Multitask learning. *Machine learning* 28(1):41–75.

Dekel, O.; Long, P. M.; and Singer, Y. 2006. Online multitask learning. In *Proceedings of the 19th Annual Conference on Learning Theory (COLT)*, 453–467.

Duchi, J. C.; Shalev-Shwartz, S.; Singer, Y.; and Tewari, A. 2010. Composite objective mirror descent. In *Proceedings of the 23rd Conference on Learning Theory (COLT)*, 14–26.

Gong, P.; Zhou, J.; Fan, W.; and Ye, J. 2014. Efficient multi-task feature learning with calibration. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 761–770.

Hazan, E., and Kale, S. 2012. Projection-free online learning. *arXiv preprint arXiv:1206.4657*.

Lapin, M.; Schiele, B.; and Hein, M. 2014. Scalable multitask representation learning for scene classification. In *2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1434–1441.

Li, G.; Chang, K.; Hoi, S. C. H.; Liu, W.; and Jain, R. C. 2011. Collaborative online learning of user generated content. In *Proceedings of the 20th ACM Conference on Information and Knowledge Management (CIKM)*, 285–290.

Liu, S.; Pan, S. J.; and Ho, Q. 2017. Distributed multi-task relationship learning. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 937–946.

Liu, H.; Wang, L.; and Zhao, T. 2014. Multivariate regression with calibration. In *Advances in Neural Information Processing Systems (NIPS)*, 127–135.

Matamoros, J. 2017. Asynchronous online ADMM for consensus problems. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 5875–5879.

Murugesan, K.; Liu, H.; Carbonell, J. G.; and Yang, Y. 2016. Adaptive smoothed online multi-task learning. In *Advances in Neural Information Processing Systems (NIPS)*, 4296–4304.

Rockafellar, R. T. 1970. *Convex analysis*, volume 28. Princeton University Press.

Saha, A.; Rai, P.; III, H. D.; and Venkatasubramanian, S. 2011. Online learning of multiple tasks and their relationships. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, 643–651.

Seltzer, M. L., and Droppo, J. 2013. Multi-task learning in deep neural networks for improved phoneme recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 6965–6969.

Shalev-Shwartz, S. 2012. Online learning and online convex optimization. *Foundations and Trends in Machine Learning* 4(2):107–194.

Smith, V.; Chiang, C.; Sanjabi, M.; and Talwalkar, A. S. 2017. Federated multi-task learning. In *Advances in Neural Information Processing Systems (NIPS)*, 4424–4434.

Wang, J.; Kolar, M.; and Srebro, N. 2016. Distributed multi-task learning with shared representation. *arXiv preprint arXiv:1603.02185*.

Xiao, Y.; Li, Z.; Yang, T.; and Zhang, L. 2017. Svd-free convex-concave approaches for nuclear norm regularization. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI)*, 3126–3132.

Xie, L.; Baytas, I. M.; Lin, K.; and Zhou, J. 2017. Privacy-preserving distributed multi-task learning with asynchronous updates. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 1195–1204.

Yang, P.; Zhao, P.; Zhou, J.; and Gao, X. 2019. Confidence weighted multitask learning. In *The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI)*, 5636–5643.

Yang, P.; Zhao, P.; and Gao, X. 2017. Robust online multi-task learning with correlative and personalized structures. *IEEE Trans. Knowl. Data Eng.* 29(11):2510–2521.

Zhang, W.; Zhao, P.; Zhu, W.; Hoi, S. C. H.; and Zhang, T. 2017. Projection-free distributed online learning in networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 4054–4062.

Zhang, C.; Zhao, P.; Hao, S.; Soh, Y. C.; Lee, B. S.; Miao, C.; and Hoi, S. C. 2018. Distributed multi-task classification: a decentralized online learning approach. *Machine Learning* 107(4):727–747.

Zheng, W.; Bellet, A.; and Gallinari, P. 2018. A distributed frank–wolfe framework for learning low-rank matrices with the trace norm. *Machine Learning* 107(8-10):1457–1475.

Zhou, J.; Liu, J.; Narayan, V. A.; and Ye, J. 2013. Modeling disease progression via multi-task learning. *NeuroImage* 78:233–248.