Distributed Distributionally Robust Optimization with Non-Convex Objectives

Yang Jiao
Tongji University
yangjiao@tongji.edu.cn

Kai Yang*
Tongji University
kaiyang@tongji.edu.cn

Dongjin Song
University of Connecticut
dongjin.song@uconn.edu

Abstract

Distributionally Robust Optimization (DRO), which aims to find an optimal decision that minimizes the worst case cost over the ambiguity set of probability distribution, has been widely applied in diverse applications, e.g., network behavior analysis, risk management, etc. However, existing DRO techniques face three key challenges: 1) how to deal with the asynchronous updating in a distributed environment; 2) how to leverage the prior distribution effectively; 3) how to properly adjust the degree of robustness according to different scenarios. To this end, we propose an asynchronous distributed algorithm, named Asynchronous Single-looP alternative Gradient projection (ASPIRE) algorithm with the Iterative Active SEt method (EASE) to tackle the distributed distributionally robust optimization (DDRO) problem. Furthermore, a new uncertainty set, i.e., constrained D-norm uncertainty set, is developed to effectively leverage the prior distribution and flexibly control the degree of robustness. Finally, our theoretical analysis elucidates that the proposed algorithm is guaranteed to converge and the iteration complexity is also analyzed. Extensive empirical studies on real-world datasets demonstrate that the proposed method can not only achieve fast convergence, and remain robust against data heterogeneity as well as malicious attacks, but also tradeoff robustness with performance.

1 Introduction

The past decade has witnessed the proliferation of smartphones and Internet of Things (IoT) devices, which generate a plethora of data everyday. Centralized machine learning requires gathering the data to a particular server to train models which incurs high communication overhead [46] and suffers privacy risks [43]. As a remedy, distributed machine learning methods have been proposed. Considering a distributed system composed of $N$ workers (devices), we denote the dataset of these workers as $\{D_1, \cdots, D_N\}$. For the $j^{th}$ ($1 \leq j \leq N$) worker, the labeled dataset is given as $D_j = \{x_j^1, y_j^1\}$, where $x_j^i \in \mathbb{R}^d$ and $y_j^i \in \{1, \cdots, c\}$ denote the $i^{th}$ data sample and the corresponding label, respectively. The distributed learning tasks can be formulated as the following optimization problem,

$$\min_{w \in \mathcal{W}} F(w) \quad \text{with} \quad F(w) := \sum_{j} f_j(w),$$

where $w \in \mathbb{R}^p$ is the model parameter to be learned and $\mathcal{W} \subseteq \mathbb{R}^p$ is a nonempty closed convex set, $f_j(\cdot)$ is the empirical risk over the $j^{th}$ worker involving only the local data:

$$f_j(w) = \frac{1}{|D_j|} \sum_{i; x_j^i \in D_j} \mathcal{L}(x_j^i, y_j^i; w),$$

*Corresponding author.
where $L_j$ is the local objective function over the $j^{th}$ worker. Problem in Eq. (1) arises in numerous areas, such as distributed signal processing [19], multi-agent optimization [36], etc. However, such problem does not consider the data heterogeneity [57] among different workers (i.e., data distribution of workers could be substantially different from each other [44]). Indeed, it has been shown that traditional federated approaches, such as FedAvg [33], built for independent and identically distributed (IID) data may perform poorly when applied to Non-IID data [27]. This issue can be mitigated via learning a robust model that aims to achieve uniformly good performance over all workers by solving the following distributionally robust optimization (DRO) problem in a distributed manner:

$$
\min_{w \in \mathcal{W}} \max_{p \in \Omega \subseteq \Delta_N} F(w, p) := \sum_{j} p_j f_j(w),
$$

where $p = [p_1, \cdots, p_N] \in \mathbb{R}^N$ is the adversarial distribution in $N$ workers, the $j^{th}$ entry in this vector, i.e., $p_j$ represents the adversarial distribution value for the $j^{th}$ worker. $\Delta_N = \{p \in \mathbb{R}^N : \mathbf{1}^\top p = 1\}$ and $\Omega$ is a subset of $\Delta_N$. Agnostic federated learning (AFL) [35] firstly introduces the distributionally robust (agnostic) loss in federated learning and provides the convergence rate for (strongly) convex functions. However, AFL does not discuss the setting of $\Omega$. DRFA-Prox [16] considers $\Omega = \Delta_N$ and imposes a regularizer on adversarial distribution to leverage the prior distribution. Nevertheless, three key challenges have not yet been addressed by prior works. First, whether it is possible to construct an uncertainty framework that can not only flexibly maintain the trade-off between the model robustness and performance but also effectively leverage the prior distribution? Second, how to design asynchronous algorithms with guaranteed convergence? Compared to synchronous algorithms, the master in asynchronous algorithms can update its parameters after receiving updates from only a small subset of workers [58, 10]. Asynchronous algorithms are particularly desirable in practice since they can relax strict data dependencies and ensure convergence even in the presence of device failures [58]. Finally, whether it is possible to flexibly adjust the degree of robustness? Moreover, it is necessary to provide convergence guarantee when the objectives (i.e., $f_j(w_j)$) are non-convex.

To this end, we propose ASPIRE-EASE to effectively address the aforementioned challenges. Firstly, different from existing works, the prior distribution is incorporated within the constraint in our formulation, which can not only leverage the prior distribution more effectively but also achieve guaranteed feasibility for any adversarial distribution within the uncertainty set. The prior distribution can be obtained from side information or uniform distribution [41], which is necessary to construct the uncertainty (ambiguity) set and obtain a more robust model [16]. Specifically, we formulate the prior distribution informed distributionally robust optimization (PD-DRO) problem as:

$$
\min_{z \in \mathcal{Z}, \{w_j \in \mathcal{W}\}} \max_{p \in \mathcal{P}} \sum_j p_j f_j(w_j)
$$

s.t. $z = w_j, \; j = 1, \cdots, N$, 

var. $z, w_1, w_2, \cdots, w_N$,

where $z \in \mathbb{R}^p$ is the global consensus variable, $w_j \in \mathbb{R}^p$ is the local variable (local model parameter) of $j^{th}$ worker and $\mathcal{Z} \subseteq \mathbb{R}^p$ is a nonempty closed convex set. $\mathcal{P} \subseteq \mathbb{R}^N_+$ is the uncertainty (ambiguity) set of adversarial distribution $p$, which is set based on the prior distribution. To solve the PD-DRO problem in an asynchronous distributed manner, we first propose Asynchronous Single-loop Alternating Gradient Projection (ASPIRE), which employs simple gradient projection steps for the update of primal and dual variables at every iteration, thus is computationally efficient. Next, the itErative Active SEt method (EASE) is employed to replace the traditional cutting plane method to improve the computational efficiency and speed up the convergence. We further provide the convergence guarantee for the proposed algorithm. Furthermore, a new uncertainty set, i.e., constrained $D$-norm ($CD$-norm), is proposed in this paper and its advantages include: 1) it can flexibly control the degree of robustness; 2) the resulting subproblem is computationally simple; 3) it can effectively leverage the prior distribution and flexibly set the bounds for every $p_j$.

**Contributions.** Our contributions can be summarized as follows:

1. We formulate a PD-DRO problem with $CD$-norm uncertainty set. PD-DRO incorporates the prior distribution as constraints which can leverage prior distribution more effectively and guarantee robustness. In addition, $CD$-norm is developed to model the ambiguity set around the prior distribution and it provides a flexible way to control the trade-off between model robustness and performance.

2. We develop a single-loop asynchronous algorithm, namely ASPIRE-EASE, to optimize PD-DRO in an asynchronous distributed manner. ASPIRE employs simple gradient projection steps to
update the variables at every iteration, which is computationally efficient. And EASE is proposed to replace cutting plane method to enhance the computational efficiency and speed up the convergence. We demonstrate that even if the objectives $f_j(w_j), \forall j$ are non-convex, the proposed algorithm is guaranteed to converge. We also theoretically derive the iteration complexity of ASPIRE-EASE.

3. Extensive empirical studies on four different real world datasets demonstrate the superior performance of the proposed algorithm. It is seen that ASPIRE-EASE can not only ensure the model’s robustness against data heterogeneity but also mitigate malicious attacks.

2 Preliminaries

2.1 Distributionally Robust Optimization

Optimization problems often contain uncertain parameters. A small perturbation of the parameters could render the optimal solution of the original optimization problem infeasible or completely meaningless [5]. Distributionally robust optimization (DRO) [28, 17, 7] assumes that the probability distributions of uncertain parameters are unknown but remain in an ambiguity (uncertainty) set and aims to find a decision that minimizes the worst case expected cost over the ambiguity set, whose general form can be expressed as,

$$\min_{x \in X} \max_{P \in P} E_P[r(x, \xi)],$$

where $x \in X$ represents the decision variable, $P$ is the ambiguity set of probability distributions $P$ of uncertain parameters $\xi$. Existing methods for solving DRO can be broadly grouped into two widely-used categories [42]: 1) Dual methods [15, 50, 18] reformulate the primal DRO problems as deterministic optimization problems through duality theory. Ben-Tal et al. [2] reformulate the robust linear optimization (RLO) problem with an ellipsoidal uncertainty set as a second-order cone optimization problem (SOCP). 2) Cutting plane methods [34, 6] (also called adversarial approaches [21]) continuously solve an approximate problem with a finite number of constraints of the primal optimization problem (SOCP). Recently, several new methods [41, 29, 23] have been developed to solve DRO, which need to solve DRO problem, and subsequently check whether new constraints are needed to refine the feasible set.

In this section, we introduce the cutting plane method for PD-DRO in Eq. (4). We first reformulate the inner maximization problem at every iteration.

3.1 Extensive empirical studies on four different real world datasets demonstrate the superior performance of the proposed algorithm. It is seen that ASPIRE-EASE can not only ensure the model’s robustness against data heterogeneity but also mitigate malicious attacks.
3 ASPIRE

Distributed optimization is an attractive approach for large-scale learning tasks [54][8] since it does not require data aggregation, which protects data privacy while also reducing bandwidth requirements [45]. When the neural network models (i.e., \( f_j(w_j), \forall j \)) are non-convex functions, is used, solving problem in Eq. (8) in a distributed manner facing two challenges. 1) Computing the optimal solution to a non-convex subproblem requires a large number of iterations and therefore is highly computationally intensive if not possible. Thus, the traditional Alternating Direction Method of Multipliers (ADMM) is ineffective. 2) The communication delays of workers may differ significantly [11], thus, asynchronous algorithms are strongly preferred.

To this end, we propose the Asynchronous Single-look Alternating Gradient Projection (ASPIRE). The advantages of the proposed algorithm include: 1) ASPIRE uses simple gradient projection steps to update variables in each iteration and therefore it is computationally more efficient than the traditional ADMM method, which seeks to find the optimal solution in non-convex (for \( w_j, \forall j \)) and convex (for \( z \) and \( h \)) optimization subproblems every iteration, 2) the proposed asynchronous algorithm does not need strict synchronization among different workers. Therefore, ASPIRE remains resilient against communication delays and potential hardware failures from workers. Details of the algorithm are given below. Firstly, we define the node as master which is responsible for updating the global variables \( z \), and we define the node which is responsible for updating the local variable \( w_j \), as worker \( j \). In each iteration, the master updates its variables once it receives updates from at least \( S \) workers, i.e., active workers, satisfying \( 1 \leq S \leq N \). \( Q_t^{t+1} \) denotes the index subset of workers from which the master receives updates during \( (t+1)^{th} \) iteration. We also assume the master will receive updated variables from every worker at least once after each \( \tau \) iterations. The augmented Lagrangian function of Eq. (8) can be written as:

\[
L_p = h + \sum_i \lambda_i (\sum_j (\beta_i + a_{i,j}) f_j(w_j) - h) + \sum_j \phi_j(z - w_j) + \sum_j \frac{\kappa_j}{2} ||z - w_j||^2, \tag{9}
\]

where \( L_p = L_p(\{w_j\}, z, h, \{\lambda_i\}, \{\phi_j\}) \), \( \lambda_i \in \Lambda, \forall i \) and \( \phi_j \in \Phi, \forall j \) represent the dual variables of inequality and equality constraints in Eq. (8), respectively. \( \Lambda \subseteq R^1 \) and \( \Phi \subseteq R^p \) are nonempty closed convex sets, constant \( \kappa_j > 0 \) is a penalty parameter. Note that Eq. (9) does not consider the second-order penalty term for inequality constraint since it will invalidate the distributed optimization. Following [52], the regularized version of Eq. (9) is employed to update all variables as follows,

\[
\tilde{L}_p(\{w_j\}, z, h, \{\lambda_i\}, \{\phi_j\}) = L_p - \sum_i \frac{c_i^1}{2} ||\lambda_i||^2 - \sum_j \frac{c_j^2}{2} ||\phi_j||^2, \tag{10}
\]

where \( c_i^1 \) and \( c_j^2 \) denote the regularization terms in \( (t+1)^{th} \) iteration. To avoid enumerating the whole dataset, the mini-batch loss could be used. A batch of instances with size \( m \) can be randomly sampled from each worker during each iteration. The loss function of these instances from \( j^{th} \) worker is given by \( \hat{f}_j(w_j) = \frac{1}{m} \sum_{i=1}^{m} \ell_j(x_i, y_i; w_j) \). It is evident that \( \mathbb{E}[\hat{f}_j(w_j)] = f_j(w_j) \) and \( \mathbb{E}[\nabla \hat{f}_j(w_j)] = \nabla f_j(w_j) \). In \( (t+1)^{th} \) master iteration, the proposed algorithm proceeds as follows.

1) Active workers update the local variables \( w_j \) as follows,

\[
w_j^{t+1} = \arg \min \{ \mathcal{P}_\mathcal{W}(w_j^{t+1} - \eta_w \nabla w_j \tilde{L}_p(\{w_j^{t+1}\}, z^{t+1}, h^{t+1}, \{\lambda_i\}, \{\phi_j\})) \}, \forall j \in Q_t^{t+1}, \tag{11}
\]

where \( \eta_w \) represents the step-size and \( \mathcal{P}_\mathcal{W} \) represents the projection onto the closed convex set \( \mathcal{W} \) and we set \( \mathcal{W} = \{w_j ||w_j||_\infty \leq \alpha_1 \} \), \( \alpha_1 \) is a positive constant. And \( l_j \) is the last iteration during which worker \( j \) was active. It is seen that \( w_j^{t+1} = w_j^{l_j} \) and \( \phi_j^{t+1} = \phi_j^{l_j} \), \( \forall j \in Q_t^{t+1} \). Then, the active workers (\( j \in Q_t^{t+1} \)) transmit their local model parameters \( w_j^{t+1} \) and loss \( f_j(w_j) \) to the master.

2) After receiving the updates from active workers, the master updates the global consensus variable \( z \), additional variable \( h \) and dual variables \( \lambda_j \) as follows,

\[
z^{t+1} = \mathcal{P}_z(z^t - \eta_z \nabla_z \tilde{L}_p(\{w_j^{t+1}\}, z^{t+1}, h^{t+1}, \{\lambda_i\}, \{\phi_j\})), \tag{12}
\]

\[
h^{t+1} = \mathcal{P}_h(h^t - \eta_h \nabla_h \tilde{L}_p(\{w_j^{t+1}\}, z^{t+1}, h^{t+1}, \{\lambda_i\}, \{\phi_j\})), \tag{13}
\]
\[ \lambda_t^{l+1} = \mathcal{P}_\Lambda (\lambda_t^l + \rho_1 \nabla \lambda_t \bar{L}_p (\{w_j^{t+1}\}, z^{t+1}, h^{t+1}, \{\lambda_t^l\}, \{\phi_j^l\}), t = 1, \cdots, |A|, \] (14)

where \( \eta_\varepsilon, \eta_h \) and \( \rho_1 \) represent the step-size. \( \mathcal{P}_Z, \mathcal{P}_H \) and \( \mathcal{P}_A \) respectively represent the projection onto the closed convex sets \( Z, H \) and \( A \). We set \( Z = \{z | ||z||_\infty \leq \alpha_1\}, H = \{h | 0 \leq h \leq \alpha_2\} \) and \( A = \{|\lambda_t| 0 \leq \lambda_t \leq \alpha_3\} \), where \( \alpha_2 \) and \( \alpha_3 \) are positive constants. \( |A| \) denotes the number of cutting planes. Then, master broadcasts \( z^{t+1}, h^{t+1}, \{\lambda_t^{l+1}\} \) to the active workers.

3) **Active workers** update the local dual variables \( \phi_j \) as follows,

\[
\phi_j^{l+1} = \left\{ \begin{array}{ll}
\mathcal{P}_\Phi (\phi_j^l + \rho_2 \nabla \phi_j \bar{L}_p (\{w_j^{t+1}\}, z^{t+1}, h^{t+1}, \{\lambda_t^{l+1}\}, \{\phi_j^l\}), j \in Q^{t+1}, \\
\phi_j^l, j \notin Q^{t+1},
\end{array} \right.
\] (15)

where \( \rho_2 \) represents the step-size and \( \mathcal{P}_\Phi \) represents the projection onto the closed convex set \( \Phi \) and we set \( \Phi = \{\phi_j | ||\phi_j||_\infty \leq \alpha_4\}, \alpha_4 \) is a positive constant. And master can also obtain \( \{\phi_j^{l+1}\} \) according to Eq. (15). It is seen that the projection operation in each step is computationally simple since the closed convex sets have simple structures [4].

### 4 Iterative Active Set Method

Cutting plane methods may give rise to numerous linear constraints and lots of extra message passing [55]. Moreover, more iterations are required to obtain the \( \varepsilon \)-stationary point when the size of a set containing cutting planes increases (which corresponds to a larger \( M \), which can be seen in Theorem 4). To improve the computational efficiency and speed up the convergence, we consider removing the inactive cutting planes. The proposed itErative Active SEt method (EASE) can be divided into the two steps: during \( T_1 \) iterations, 1) solving the cutting plane generation subproblem to generate cutting plane, and 2) removing the inactive cutting plane every \( k \) iterations, where \( k > 0 \) is a pre-set constant and can be controlled flexibly.

The cutting planes are generated according to the uncertainty set. For example, if we employ ellipsoid uncertainty set, the cutting plane is generated via solving a SOCP. In this paper, we propose \( CD \)-norm uncertainty set, which can be expressed as follows,

\[ \mathcal{P} = \{p : -\bar{p}_j \leq p_j - q_j \leq \bar{p}_j, \sum_j \left| \frac{p_j - q_j}{p_j} \right| \leq \Gamma, 1^T p = 1\}, \] (16)

where \( \Gamma \in \mathbb{R}^1 \) can flexibly control the level of robustness, \( q = [q_1, \cdots, q_N] \in \mathbb{R}^N \) represents the prior distribution, \( -\bar{p}_j \) and \( \bar{p}_j (\bar{p}_j \geq 0) \) represent the lower and upper bounds for \( p_j - q_j \), respectively. The setting of \( q \) and \( \bar{p}_j, \forall j \) are based on the prior knowledge. \( D \)-norm is a classical uncertainty set (which is also called as budget uncertainty set) [5]. We call Eq. (16) \( CD \)-norm uncertainty set since \( p \) is a probability vector so all the entries of this vector are non-negative and add up to exactly one, i.e., \( 1^T p = 1 \). Due to the special structure of \( CD \)-norm, the cutting plane generation subproblem is easy to solve and the level of robustness in terms of the outage probability, i.e., probabilistic bounds of the violations of constraints can be flexibly adjusted via a single parameter \( \Gamma \). We claim that \( l_1 \)-norm (or twice total variation distance) uncertainty set is closely related to \( CD \)-norm uncertainty set. Nevertheless, there are two differences: 1) \( CD \)-norm uncertainty set could be regarded as a weighted \( l_1 \)-norm with additional constraints. 2) \( CD \)-norm uncertainty set can flexibly set the lower and upper bounds for every \( p_j \) (i.e., \( q_j - \bar{p}_j \leq p_j \leq \bar{p}_j + \bar{p}_j \)) while \( 0 \leq p_j \leq 1, \forall j \) in \( l_1 \)-norm uncertainty set. Based on the \( CD \)-norm uncertainty set, the cutting plane can be derived as follows,

1) Solve the following problem,

\[
\begin{align*}
p^{t+1} &= \arg \max_{p_1, \cdots, p_N} \left( \sum_j (p_j - p)f_j(w_j) \right) \\
\text{s.t.} \quad & \sum_j \left| \frac{p_j - q_j}{p_j} \right| \leq \Gamma, \quad -\bar{p}_j \leq p_j - q_j \leq \bar{p}_j, \forall j, \quad \sum_j p_j = 1 \\
& \text{var.} \quad p_1, \cdots, p_N
\end{align*}
\] (17)

where \( p^{t+1} = [p_1^{t+1}, \cdots, p_N^{t+1}] \in \mathbb{R}^N \). Let \( \bar{a}^{t+1} = p^{t+1} - \bar{p} \), where \( \bar{p} = [\bar{p}_1, \cdots, \bar{p}_N] \in \mathbb{R}^N \). This first step aims to obtain the distribution \( \bar{a}^{t+1} \) by solving problem in Eq. (17). This problem can be effectively solved through combining merge sort [13] (for sorting \( \bar{p}_j f_j(w_j), j = 1, \cdots, N \)) with few basic arithmetic operations (for obtaining \( p_j^{t+1}, j = 1, \cdots, N \)). Since \( N \) is relatively large in
After the cutting plane subproblem is solved, the inactive cutting plane will be removed, that is: when a new cutting plane is added, its corresponding dual variable

Definition 2

(ε-stationary point) \( (\{w_j^t\}, z^t, h^t, \{\lambda_l^t\}, \{\phi_j^t\}) \) is an \( ε \)-stationary point \( (ε \geq 0) \) of a differentiable function \( L_p \), if \( ||\nabla G^t|| \leq ε \). \( T(ε) \) is the first iteration index such that \( ||\nabla G^t|| \leq ε \), i.e.,

Assumption 1 (Smoothness/Gradient Lipschitz) \( L_p \) has Lipschitz continuous gradients. We assume that there exists \( L > 0 \) satisfying
We compare the proposed ASPIRE-EASE with baseline methods based on SHL\cite{20}, Person Activity\cite{26}, Single Chest-Mounted Accelerometer (SM-AC)\cite{9} and Fashion MNIST\cite{51} datasets. The baseline methods include Ind\cite{108}, Mix\cite{108}, DRFA-Prox\cite{16}. The detailed descriptions of datasets and baselines are given in Appendix C.

In our empirical studies, since the downstream tasks are multi-class classification, the cross entropy loss is used on each worker\( (i.e., L_c(\cdot), \forall j)\). For SHL, Person Activity, and SM-AC datasets, we adopt the deep multilayer perceptron\cite{49} as the base model. And we use the same logistic regression model as in\cite{35,16} for Fashion MNIST\cite{51} dataset. The base models are trained with SGD. More details are given in Appendix C.

In our empirical studies, since the downstream tasks are multi-class classification, the cross entropy loss is used on each worker\( (i.e., L_c(\cdot), \forall j)\). For SHL, Person Activity, and SM-AC datasets, we adopt the deep multilayer perceptron\cite{49} as the base model. And we use the same logistic regression model as in\cite{35,16} for Fashion MNIST\cite{51} dataset. The base models are trained with SGD. More details are given in Appendix C.

In our empirical studies, since the downstream tasks are multi-class classification, the cross entropy loss is used on each worker\( (i.e., L_c(\cdot), \forall j)\). For SHL, Person Activity, and SM-AC datasets, we adopt the deep multilayer perceptron\cite{49} as the base model. And we use the same logistic regression model as in\cite{35,16} for Fashion MNIST\cite{51} dataset. The base models are trained with SGD. More details are given in Appendix C.

In our empirical studies, since the downstream tasks are multi-class classification, the cross entropy loss is used on each worker\( (i.e., L_c(\cdot), \forall j)\). For SHL, Person Activity, and SM-AC datasets, we adopt the deep multilayer perceptron\cite{49} as the base model. And we use the same logistic regression model as in\cite{35,16} for Fashion MNIST\cite{51} dataset. The base models are trained with SGD. More details are given in Appendix C.

In our empirical studies, since the downstream tasks are multi-class classification, the cross entropy loss is used on each worker\( (i.e., L_c(\cdot), \forall j)\). For SHL, Person Activity, and SM-AC datasets, we adopt the deep multilayer perceptron\cite{49} as the base model. And we use the same logistic regression model as in\cite{35,16} for Fashion MNIST\cite{51} dataset. The base models are trained with SGD. More details are given in Appendix C.

In our empirical studies, since the downstream tasks are multi-class classification, the cross entropy loss is used on each worker\( (i.e., L_c(\cdot), \forall j)\). For SHL, Person Activity, and SM-AC datasets, we adopt the deep multilayer perceptron\cite{49} as the base model. And we use the same logistic regression model as in\cite{35,16} for Fashion MNIST\cite{51} dataset. The base models are trained with SGD. More details are given in Appendix C.

In our empirical studies, since the downstream tasks are multi-class classification, the cross entropy loss is used on each worker\( (i.e., L_c(\cdot), \forall j)\). For SHL, Person Activity, and SM-AC datasets, we adopt the deep multilayer perceptron\cite{49} as the base model. And we use the same logistic regression model as in\cite{35,16} for Fashion MNIST\cite{51} dataset. The base models are trained with SGD. More details are given in Appendix C.

In our empirical studies, since the downstream tasks are multi-class classification, the cross entropy loss is used on each worker\( (i.e., L_c(\cdot), \forall j)\). For SHL, Person Activity, and SM-AC datasets, we adopt the deep multilayer perceptron\cite{49} as the base model. And we use the same logistic regression model as in\cite{35,16} for Fashion MNIST\cite{51} dataset. The base models are trained with SGD. More details are given in Appendix C.

In our empirical studies, since the downstream tasks are multi-class classification, the cross entropy loss is used on each worker\( (i.e., L_c(\cdot), \forall j)\). For SHL, Person Activity, and SM-AC datasets, we adopt the deep multilayer perceptron\cite{49} as the base model. And we use the same logistic regression model as in\cite{35,16} for Fashion MNIST\cite{51} dataset. The base models are trained with SGD. More details are given in Appendix C.

In our empirical studies, since the downstream tasks are multi-class classification, the cross entropy loss is used on each worker\( (i.e., L_c(\cdot), \forall j)\). For SHL, Person Activity, and SM-AC datasets, we adopt the deep multilayer perceptron\cite{49} as the base model. And we use the same logistic regression model as in\cite{35,16} for Fashion MNIST\cite{51} dataset. The base models are trained with SGD. More details are given in Appendix C.

In our empirical studies, since the downstream tasks are multi-class classification, the cross entropy loss is used on each worker\( (i.e., L_c(\cdot), \forall j)\). For SHL, Person Activity, and SM-AC datasets, we adopt the deep multilayer perceptron\cite{49} as the base model. And we use the same logistic regression model as in\cite{35,16} for Fashion MNIST\cite{51} dataset. The base models are trained with SGD. More details are given in Appendix C.

In our empirical studies, since the downstream tasks are multi-class classification, the cross entropy loss is used on each worker\( (i.e., L_c(\cdot), \forall j)\). For SHL, Person Activity, and SM-AC datasets, we adopt the deep multilayer perceptron\cite{49} as the base model. And we use the same logistic regression model as in\cite{35,16} for Fashion MNIST\cite{51} dataset. The base models are trained with SGD. More details are given in Appendix C.

In our empirical studies, since the downstream tasks are multi-class classification, the cross entropy loss is used on each worker\( (i.e., L_c(\cdot), \forall j)\). For SHL, Person Activity, and SM-AC datasets, we adopt the deep multilayer perceptron\cite{49} as the base model. And we use the same logistic regression model as in\cite{35,16} for Fashion MNIST\cite{51} dataset. The base models are trained with SGD. More details are given in Appendix C.

In our empirical studies, since the downstream tasks are multi-class classification, the cross entropy loss is used on each worker\( (i.e., L_c(\cdot), \forall j)\). For SHL, Person Activity, and SM-AC datasets, we adopt the deep multilayer perceptron\cite{49} as the base model. And we use the same logistic regression model as in\cite{35,16} for Fashion MNIST\cite{51} dataset. The base models are trained with SGD. More details are given in Appendix C.

In our empirical studies, since the downstream tasks are multi-class classification, the cross entropy loss is used on each worker\( (i.e., L_c(\cdot), \forall j)\). For SHL, Person Activity, and SM-AC datasets, we adopt the deep multilayer perceptron\cite{49} as the base model. And we use the same logistic regression model as in\cite{35,16} for Fashion MNIST\cite{51} dataset. The base models are trained with SGD. More details are given in Appendix C.

In our empirical studies, since the downstream tasks are multi-class classification, the cross entropy loss is used on each worker\( (i.e., L_c(\cdot), \forall j)\). For SHL, Person Activity, and SM-AC datasets, we adopt the deep multilayer perceptron\cite{49} as the base model. And we use the same logistic regression model as in\cite{35,16} for Fashion MNIST\cite{51} dataset. The base models are trained with SGD. More details are given in Appendix C.

In our empirical studies, since the downstream tasks are multi-class classification, the cross entropy loss is used on each worker\( (i.e., L_c(\cdot), \forall j)\). For SHL, Person Activity, and SM-AC datasets, we adopt the deep multilayer perceptron\cite{49} as the base model. And we use the same logistic regression model as in\cite{35,16} for Fashion MNIST\cite{51} dataset. The base models are trained with SGD. More details are given in Appendix C.

In our empirical studies, since the downstream tasks are multi-class classification, the cross entropy loss is used on each worker\( (i.e., L_c(\cdot), \forall j)\). For SHL, Person Activity, and SM-AC datasets, we adopt the deep multilayer perceptron\cite{49} as the base model. And we use the same logistic regression model as in\cite{35,16} for Fashion MNIST\cite{51} dataset. The base models are trained with SGD. More details are given in Appendix C.

In our empirical studies, since the downstream tasks are multi-class classification, the cross entropy loss is used on each worker\( (i.e., L_c(\cdot), \forall j)\). For SHL, Person Activity, and SM-AC datasets, we adopt the deep multilayer perceptron\cite{49} as the base model. And we use the same logistic regression model as in\cite{35,16} for Fashion MNIST\cite{51} dataset. The base models are trained with SGD. More details are given in Appendix C.

In our empirical studies, since the downstream tasks are multi-class classification, the cross entropy loss is used on each worker\( (i.e., L_c(\cdot), \forall j)\). For SHL, Person Activity, and SM-AC datasets, we adopt the deep multilayer perceptron\cite{49} as the base model. And we use the same logistic regression model as in\cite{35,16} for Fashion MNIST\cite{51} dataset. The base models are trained with SGD. More details are given in Appendix C.

In our empirical studies, since the downstream tasks are multi-class classification, the cross entropy loss is used on each worker\( (i.e., L_c(\cdot), \forall j)\). For SHL, Person Activity, and SM-AC datasets, we adopt the deep multilayer perceptron\cite{49} as the base model. And we use the same logistic regression model as in\cite{35,16} for Fashion MNIST\cite{51} dataset. The base models are trained with SGD. More details are given in Appendix C.

In our empirical studies, since the downstream tasks are multi-class classification, the cross entropy loss is used on each worker\( (i.e., L_c(\cdot), \forall j)\). For SHL, Person Activity, and SM-AC datasets, we adopt the deep multilayer perceptron\cite{49} as the base model. And we use the same logistic regression model as in\cite{35,16} for Fashion MNIST\cite{51} dataset. The base models are trained with SGD. More details are given in Appendix C.
Table 1: Performance comparisons based on $\text{Acc}_w$ ($\%$), $\text{Loss}_w$ ↓ and $\text{Std}$ ↓ (↑ and ↓ respectively denote higher scores represent better performance and lower scores represent better performance). The boldfaced digits represent the best results, “−” represents not available.

| Model | SHL | Person Activity | SC-MA | Fashion MNIST |
|-------|-----|-----------------|-------|---------------|
|       | $\text{Acc}_w$ | $\text{Loss}_w$ | Std. | $\text{Acc}_w$ | $\text{Loss}_w$ | Std. | $\text{Acc}_w$ | $\text{Loss}_w$ | Std. |
| max[$\text{Ind}_j$] | 78.94±0.08 | 0.515±0.019 | 1.02 | 59.54±0.21 | 1.051±0.016 | 0.79 | 56.71±0.16 | 1.119±0.028 | 3.48 |
| MixE | 79.16±1.13 | 0.515±0.019 | 1.02 | 59.31±0.44 | 1.051±0.016 | 0.79 | 55.81±0.29 | 1.127±0.021 | 3.46 |
| Ind | 79.4±1.27 | 0.521±0.023 | 1.36 | 59.54±0.21 | 1.051±0.016 | 0.79 | 56.71±0.16 | 1.119±0.028 | 3.48 |
| FedAvg | 69.96±1.01 | 0.802±0.023 | 5.21 | 56.28±0.69 | 1.154±0.019 | 3.13 | 49.3±0.96 | 1.44±0.015 | 7.17 |
| APL | 78.11±0.99 | 0.582±0.021 | 1.87 | 58.39±0.79 | 1.172±0.018 | 3.50 | 77.95±0.51 | 0.702±0.007 | 1.86 |
| MixE | 78.34±1.46 | 0.532±0.034 | 1.85 | 58.62±0.16 | 1.096±0.037 | 1.26 | 54.61±0.76 | 1.151±0.039 | 4.49 |
|ÐRFa-Prox [16] | 78.31±1.33 | 0.532±0.034 | 1.85 | 58.62±0.16 | 1.096±0.037 | 1.26 | 54.61±0.76 | 1.151±0.039 | 4.49 |
| ASPIRE-EASE | 79.16±1.13 | 0.515±0.019 | 1.02 | 59.54±0.21 | 1.051±0.016 | 0.79 | 56.71±0.16 | 1.119±0.028 | 3.48 |
| ASPIRE-EASE$_{eg}$ | 79.4±1.27 | 0.521±0.023 | 1.36 | 59.54±0.21 | 1.051±0.016 | 0.79 | 56.71±0.16 | 1.119±0.028 | 3.48 |

the master will make an update once it receives a message. Each experiment is repeated 10 times, both mean and standard deviations are reported. We implement our model with PyTorch and conduct all the experiments on a server with two TITAN V GPUs.

6.2 Results

Robustness against Data Heterogeneity. We first assess the robustness of the proposed ASPIRE-EASE by comparing it with baseline methods when data are heterogeneously distributed across different workers. Specifically, we compare the $\text{Acc}_w$, $\text{Loss}_w$ and $\text{Std}$ of different methods on all datasets. The performance comparison results are shown in Table 1. In this table, we can observe that max[$\text{Ind}_j$], which represents the best performance of individual training over all workers, exhibits the worst robustness on SHL, Person Activity, and SC-MA. This is because individual training (max[$\text{Ind}_j$]) only learns from the data in its local worker and cannot generalize to other workers due to different data distributions. Note that max[$\text{Ind}_j$] is unavailable for Fashion MNIST since each worker only contains one class of data and cross entropy loss cannot be used in this case. max[$\text{Ind}_j$] also does not have $\text{Loss}_w$, since $\text{Ind}_j$ is trained only on individual worker $j$. The FedAvg and MixE$_{even}$ exhibit better performance than max[$\text{Ind}_j$] since they consider the data from all workers. Nevertheless, FedAvg and MixE$_{even}$ only assign the fixed weight for each worker. AFL is more robust than FedAvg and MixE$_{even}$ since it not only utilizes the data from all workers but also considers optimizing the weight of each worker. DRFA-Prox outperforms AFL since it also considers the prior distribution and regards it as a regularizer in the objective function. Finally, we can observe that the proposed ASPIRE-EASE shows excellent robustness, which can be attributed to two factors: 1) ASPIRE-EASE considers data from all workers and can optimize the weight of each worker; 2) compared with DRFA-Prox which uses prior distribution as a regularizer, the prior distribution is incorporated within the constraint in our formulation (Eq. 4), which can be leveraged more effectively. And it is seen that ASPIRE-EASE can perform periodic communication since ASPIRE-EASE$_{eg}$, which represents ASPIRE-EASE with periodic communication, also has excellent performance.

Within ASPIRE-EASE, the level of robustness can be controlled by adjusting $\Gamma$. Specially, when $\Gamma = 0$, we obtain a nominal optimization problem in which no adversarial distribution is considered. The size of the uncertainty set will increase with $\Gamma$ (when $\Gamma \leq N$), which enhances the adversarial robustness of the model. As shown in Figure 1, the robustness of ASPIRE-EASE can be gradually enhanced when $\Gamma$ increases. More results are available in Figure C2 of Appendix C.

Robustness against Malicious Attacks. To assess the model robustness against malicious attacks, malicious workers with backdoor attacks [43], which attempt to mislead the model training process, are added to the distributed system. Following [14], we report the success attack rate of backdoor attacks for comparison. It can be calculated by checking how many instances in the backdoor dataset can be misled and categorized into the target labels. Lower success attack rates indicate more robustness against backdoor attacks. The comparison results are summarized in Table 2 and more detailed settings of backdoor attacks are available in Appendix C. In Table 2, we observe that AFL can be attacked easily since it could assign higher weights to malicious workers. Compared to AFL, FedAvg and MixE$_{even}$ achieve relatively lower success attack rates since they assign equal weights to the malicious workers and other workers. DRFA-Prox can achieve even lower success attack rates since it can leverage the prior distribution to assign lower weights for malicious workers. The proposed ASPIRE-EASE achieves the lowest success attack rates since it can leverage the prior distribution more effectively. Specifically, it will assign lower weights to malicious workers with tight theoretical guarantees.
Figure 1: Control the degree of robustness (worst case performance in the problem) on (a) Person Activity, (b) SC-MA datasets.

Figure 2: Comparison of the convergence time on worst case worker on (a) Person Activity, (b) SC-MA datasets.

Figure 3: Comparison of ASPIRE-CP and ASPIRE-EASE regarding the number of cutting planes on (a) Person Activity, (b) SC-MA datasets.

**Efficiency.** In Figure 2, we compare the convergence speed of the proposed ASPIRE-EASE with AFL and DRFA-Prox by considering different communication and computation delays for each worker. The proposed ASPIRE-EASE has two variants, ASPIRE-CP (ASPIRE with cutting plane method), ASPIRE-EASE(-) (ASPIRE-EASE without asynchronous setting). More results are available in Figure C3 of Appendix C. Based on the comparison, we observe that the proposed ASPIRE-EASE generally converges faster than baseline methods and its two variants. This is because 1) compared with AFL, DRFA-Prox, and ASPIRE-EASE(-), ASPIRE-EASE is an asynchronous algorithm in which the master updates its parameters only after receiving the updates from active workers instead of all workers; 2) unlike DRFA-Prox, the master in ASPIRE-EASE only needs to communicate with active workers once per iteration; 3) compared with ASPIRE-CP, ASPIRE-EASE utilizes active set method instead of cutting plane method, which is more efficient. It is seen from Figure 2 that, the convergence speed of ASPIRE-EASE mainly benefits from the asynchronous setting.

**Ablation Study.** For ASPIRE, compared with cutting plane method, EASE is more efficient since it considers removing the inactive cutting planes. To demonstrate the efficiency of EASE, we firstly compare ASPIRE-EASE with ASPIRE-CP concerning the number of cutting planes used during the training. In Figure 3, we can observe that ASPIRE-EASE uses fewer cutting planes than ASPIRE-CP, thus is more efficient. The convergence speed of ASPIRE-EASE and ASPIRE-CP in Figure 2 also suggests that ASPIRE-EASE converges much faster than ASPIRE-CP. More results are available in Figure C3 and C4 Appendix C.

**7 Conclusion**

In this paper, we present ASPIRE-EASE method to effectively solve the distributed distributionally robust optimization problem with non-convex objectives. In addition, CD-norm uncertainty set has been proposed to effectively incorporate the prior distribution into the problem formulation, which allows for flexible adjustment of the degree of robustness of DRO. Theoretical analysis has also been conducted to analyze the convergence properties and the iteration complexity of ASPIRE-EASE. ASPIRE-EASE exhibits strong empirical performance on multiple real-world datasets and is effective in tackling DRO problems in a fully distributed and asynchronous manner. In the future work, more uncertainty sets could be designed for our framework and more update rules for variables in ASPIRE could be considered.
Acknowledgments and Disclosure of Funding

The work of Yang Jiao and Kai Yang was supported in part by the Fundamental Research Funds for the Central Universities of China, in part by the Shenzhen Institute of Artificial Intelligence and Robotics for Society (AIRS), in part by the National Natural Science Foundation of China under Grant 61771013, and in part by the Fundamental Research Funds of Shanghai Jiading District.

References

[1] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov. How to backdoor federated learning. In *International Conference on Artificial Intelligence and Statistics*, pages 2938–2948. PMLR, 2020.

[2] A. Ben-Tal and A. Nemirovski. Robust solutions of uncertain linear programs. *Operations research letters*, 25(1):1–13, 1999.

[3] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski. *Robust optimization*. Princeton university press, 2009.

[4] D. P. Bertsekas. Nonlinear programming. *Journal of the Operational Research Society*, 48(3):334–334, 1997.

[5] D. Bertsimas and M. Sim. The price of robustness. *Operations research*, 52(1):35–53, 2004.

[6] D. Bertsimas, I. Dunning, and M. Lubin. Reformulation versus cutting-planes for robust optimization. *Computational Management Science*, 13(2):195–217, 2016.

[7] J. Blanchet and K. Murthy. Quantifying distributional model risk via optimal transport. *Mathematics of Operations Research*, 44(2):565–600, 2019.

[8] L. Bottou, F. E. Curtis, and J. Nocedal. Optimization methods for large-scale machine learning. *Siam Review*, 60(2):223–311, 2018.

[9] P. Casale, O. Pujol, and P. Radeva. Personalization and user verification in wearable systems using biometric walking patterns. *Personal and Ubiquitous Computing*, 16(5):563–580, 2012.

[10] T.-H. Chang, M. Hong, W.-C. Liao, and X. Wang. Asynchronous distributed ADMM for large-scale optimization—Part I: Algorithm and convergence analysis. *IEEE Transactions on Signal Processing*, 64(12):3118–3130, 2016.

[11] Y. Chen, Y. Ning, M. Slawski, and H. Rangwala. Asynchronous online federated learning for edge devices with Non-IID data. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 15–24. IEEE, 2020.

[12] A. Cohen, A. Daniely, Y. Drori, T. Koren, and M. Schain. Asynchronous stochastic optimization robust to arbitrary delays. *Advances in Neural Information Processing Systems*, 34:9024–9035, 2021.

[13] R. Cole. Parallel merge sort. *SIAM Journal on Computing*, 17(4):770–785, 1988.

[14] J. Dai, C. Chen, and Y. Li. A backdoor attack against LSTM-based text classification systems. *IEEE Access*, 7:138872–138878, 2019.

[15] E. Delage and Y. Ye. Distributionally robust optimization under moment uncertainty with application to data-driven problems. *Operations research*, 58(3):595–612, 2010.

[16] Y. Deng, M. M. Kamani, and M. Mahdavi. Distributionally robust federated averaging. *arXiv preprint arXiv:2102.12660*, 2021.

[17] J. C. Duchi and H. Namkoong. Learning models with uniform performance via distributionally robust optimization. *The Annals of Statistics*, 49(3):1378–1406, 2021.

[18] R. Gao and A. J. Kleywegt. Distributionally robust stochastic optimization with wasserstein distance. *arXiv preprint arXiv:1604.02199*, 2016.
[19] G. Geraci, M. Wildemeersch, and T. Q. Quek. Energy efficiency of distributed signal processing in wireless networks: A cross-layer analysis. *IEEE Transactions on Signal Processing*, 64(4): 1034–1047, 2015.

[20] H. Gjoreski, M. Ciliberto, L. Wang, F. J. O. Morales, S. Mekki, S. Valentin, and D. Roggen. The university of sussex-huawei locomotion and transportation dataset for multimodal analytics with mobile devices. *IEEE Access*, 6:42592–42604, 2018.

[21] B. L. Gorissen, İ. Yarıkoğlu, and D. den Hertog. A practical guide to robust optimization. *Omega*, 53:124–137, 2015.

[22] F. Haddadpour, M. M. Kamani, M. Mahdavi, and A. Karbasi. Learning distributionally robust models at scale via composite optimization. *arXiv preprint arXiv:2203.09607*, 2022.

[23] Y. Hu, X. Chen, and N. He. On the bias-variance-cost tradeoff of stochastic optimization. *Advances in Neural Information Processing Systems*, 34, 2021.

[24] J. Jiang, W. Zhang, J. Gu, and W. Zhu. Asynchronous decentralized online learning. *Advances in Neural Information Processing Systems*, 34:20185–20196, 2021.

[25] C. Jin, P. Netrapalli, and M. Jordan. What is local optimality in nonconvex-nonconcave minimax optimization? In *International Conference on Machine Learning*, pages 4880–4889. PMLR, 2020.

[26] B. Kaluža, V. Mirchevska, E. Dovgan, M. Luštrek, and M. Gams. An agent-based approach to care in independent living. In *International joint conference on ambient intelligence*, pages 177–186. Springer, 2010.

[27] S. P. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, and A. T. Suresh. SCAFFOLD: Stochastic Controlled Averaging for On-Device Federated Learning. 2019.

[28] D. Kuhn, P. M. Esfahani, V. A. Nguyen, and S. Shafieezadeh-Abadeh. Wasserstein distributionally robust optimization: Theory and applications in machine learning. In *Operations Research & Management Science in the Age of Analytics*, pages 130–166. INFORMS, 2019.

[29] D. Levy, Y. Carmon, J. C. Duchi, and A. Sidford. Large-scale methods for distributionally robust optimization. *Advances in Neural Information Processing Systems*, 33:8847–8860, 2020.

[30] W.-H. Liao and Y.-T. Huang. Investigation of DNN model robustness using heterogeneous datasets. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 4393–4397. IEEE, 2021.

[31] T. Lin, C. Jin, and M. Jordan. On gradient descent ascent for nonconvex-concave minimax problems. In *International Conference on Machine Learning*, pages 6083–6093. PMLR, 2020.

[32] S. Lu, I. Tsaknakis, M. Hong, and Y. Chen. Hybrid block successive approximation for one-sided non-convex min-max problems: algorithms and applications. *IEEE Transactions on Signal Processing*, 68:3676–3691, 2020.

[33] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[34] S. Mehrotra and D. Papp. A cutting surface algorithm for semi-infinite convex programming with an application to moment robust optimization. *SIAM Journal on Optimization*, 24(4): 1670–1697, 2014.

[35] M. Mohri, G. Sivek, and A. T. Suresh. Agnostic federated learning. In *International Conference on Machine Learning*, pages 4615–4625. PMLR, 2019.

[36] A. Nedic and A. Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, 2009.

[37] Y. Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2003.
[38] Q. Qi, Z. Guo, Y. Xu, R. Jin, and T. Yang. An online method for a class of distributionally robust optimization with non-convex objectives. *Advances in Neural Information Processing Systems*, 34:10067–10080, 2021.

[39] J. Qian, X. Fafoutis, and L. K. Hansen. Towards federated learning: Robustness analytics to data heterogeneity. *arXiv preprint arXiv:2002.05038*, 2020.

[40] J. Qian, L. K. Hansen, X. Fafoutis, P. Tiwari, and H. M. Pandey. Robustness analytics to data heterogeneity in edge computing. *Computer Communications*, 164:229–239, 2020.

[41] Q. Qian, S. Zhu, J. Tang, R. Jin, B. Sun, and H. Li. Robust optimization over multiple domains. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4739–4746, 2019.

[42] H. Rahimian and S. Mehrrotra. Distributionally robust optimization: A review. *arXiv preprint arXiv:1908.05659*, 2019.

[43] S. Sicari, A. Rizzardi, L. A. Greico, and A. Coen-Porisini. Security, privacy and trust in Internet of Things: The road ahead. *Computer networks*, 76:146–164, 2015.

[44] K. Singhal, H. Sidahmed, Z. Garrett, S. Wu, J. Rush, and S. Prakash. Federated reconstruction: Partially local federated learning. *Advances in Neural Information Processing Systems*, 34, 2021.

[45] T. Subramanya and R. Riggio. Centralized and federated learning for predictive VNF autoscaling in multi-domain 5G networks and beyond. *IEEE Transactions on Network and Service Management*, 18(1):63–78, 2021.

[46] J. Sun, T. Chen, G. B. Giannakis, and Z. Yang. Communication-efficient distributed learning via lazily aggregated quantized gradients. *arXiv preprint arXiv:1909.07588*, 2019.

[47] K. K. Thekumparampil, P. Jain, P. Netrapalli, and S. Oh. Efficient algorithms for smooth minimax optimization. *Advances in Neural Information Processing Systems*, 32, 2019.

[48] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B. Y. Zhao. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 707–723. IEEE, 2019.

[49] Z. Wang, W. Yan, and T. Oates. Time series classification from scratch with deep neural networks: A strong baseline. In *2017 International joint conference on neural networks (IJCNN)*, pages 1578–1585. IEEE, 2017.

[50] W. Wiesemann, D. Kuhn, and B. Rustem. Robust Markov decision processes. *Mathematics of Operations Research*, 38(1):153–183, 2013.

[51] H. Xiao, K. Rasul, and R. Vollgraf. Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[52] Z. Xu, H. Zhang, Y. Xu, and G. Lan. A unified single-loop alternating gradient projection algorithm for nonconvex-concave and convex-nonconcave minimax problems. *arXiv preprint arXiv:2006.02032*, 2020.

[53] Z. Xu, J. Shen, Z. Wang, and Y. Dai. Zeroth-order alternating randomized gradient projection algorithms for general nonconvex-concave minimax problems. *arXiv preprint arXiv:2108.00473*, 2021.

[54] K. Yang, Y. Wu, J. Huang, X. Wang, and S. Verdú. Distributed robust optimization for communication networks. In *IEEE INFOCOM 2008-The 27th Conference on Computer Communications*, pages 1157–1165. IEEE, 2008.

[55] K. Yang, J. Huang, Y. Wu, X. Wang, and M. Chiang. Distributed robust optimization (DRO), part I: Framework and example. *Optimization and Engineering*, 15(1):35–67, 2014.

[56] I. Yanköglu, B. L. Gorissen, and D. den Hertog. A survey of adjustable robust optimization. *European Journal of Operational Research*, 277(3):799–813, 2019.
[57] S. Zawad, A. Ali, P.-Y. Chen, A. Anwar, Y. Zhou, N. Baracaldo, Y. Tian, and F. Yan. Curse or redemption? how data heterogeneity affects the robustness of federated learning. *arXiv preprint arXiv:2102.00655*, 2021.

[58] R. Zhang and J. Kwok. Asynchronous distributed ADMM for consensus optimization. In *International conference on machine learning*, pages 1701–1709. PMLR, 2014.

[59] X. Zhou. On the fenchel duality between strong convexity and lipschitz continuous gradient. *arXiv preprint arXiv:1803.06573*, 2018.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper’s contributions and scope? [Yes] See Section 1.
   (b) Did you describe the limitations of your work? [Yes] See Section 7.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] There is no potential negative societal impact of our work.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Section 5.
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix A and B.

3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The references of the data used in this paper are added in Section 6.1.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section C.2.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Section 6.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Section 6.1.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you’re using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]