IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 1 June 2023, accepted 29 June 2023, date of publication 3 July 2023, date of current version 11 July 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3291677

RESEARCH ARTICLE

Online Decentralized Multi-Agents Meta-Learning With Byzantine Resiliency

OLUSOLA T. ODEYOMI, (Member, IEEE), BASSEY UDE, (Student Member, IEEE), AND KAUSHIK ROY

Department of Computer Science, North Carolina Agricultural and Technical State University, Greensboro, NC 27411, USA

Corresponding author: Olusola T. Odeyomi (otodeyomi@ncat.edu)

This work was supported by the Department of Computer Science, North Carolina Agricultural and Technical State University.

ABSTRACT Meta-learning is a learning-to-learn paradigm that leverages past learning experiences for quick adaptation to new learning tasks. It has a wide application, such as in few-shot learning, reinforcement learning, neural architecture search, federated learning, etc. It has been extended to the online learning setting where task data distribution arrives sequentially. This provides continuous lifelong learning. However, in the online meta-learning setting, a single agent has to learn many varieties of related tasks. Yet, a single agent is limited to its local task data and must collaborate with neighboring agents to improve its learning performance. Therefore, online decentralized meta-learning algorithms are designed to allow an agent to collaborate with neighboring agents in order to improve learning performance. Despite their advantages, online decentralized meta-learning algorithms are susceptible to Byzantine attacks caused by the diffusion of poisonous information from unidentifiable Byzantine agents in the network. This is a serious problem where normal agents are unable to learn and convergence to the global meta-initializer is thwarted. State-of-the-art algorithms, such as BRIDGE, designed to provide robustness against Byzantine attacks are slow and cannot work in online learning settings. Therefore, we propose an online decentralized meta-learning algorithm that works with two Byzantine-resilient aggregation techniques, which are modified coordinate-wise screening and centerpoint aggregation. The proposed algorithm provides faster convergence speed and guarantees both resiliency and continuous lifelong learning. Our simulation results show that the proposed algorithm performs better than state-of-the-art algorithms.

INDEX TERMS Byzantine attacks, decentralized networks, diffusion learning, meta-learning, online learning, regret.

I. INTRODUCTION

Meta-learning is a process of extracting experiences from multiple related tasks over a series of learning episodes to improve learning performance on new tasks. This is similar to human brains capable of leveraging past experiences to learn a new task. Thus, meta-learning is a learning-to-learn paradigm that leads to improvement in data and computational efficiency. Meta-learning overcomes the weaknesses of conventional deep learning algorithms that do not leverage past experiences for quick adaptation [1]. During meta-learning, task-agnostic knowledge extracted in multi-task scenarios is applied to other related tasks in the same domain [2], [3]. Meta-learning can also be applied in single-task scenarios where learning occurs on a single task over multiple learning episodes [4], [5]. Meta-learning has practical applications in few-shot learning [6], unsupervised learning [7], reinforcement learning [8], hyperparameter optimization [5], and neural architecture search [9], [10], [11].

Model Agnostic Meta-learning (MAML) proposed by Finn et al. [12] is a seminal work on the application of stochastic gradient descent in meta-learning. The work aims at learning a suitable meta-initialization model at training time that can generalize to multiple related tasks during test

The associate editor coordinating the review of this manuscript and approving it for publication was Ramakrishnan Srinivasan.

time. From this work, meta-learning has grown to become a hot and interesting research area in machine learning. Refer to [1], [13], [14], and [15] for in-depth survey work on meta-learning. The majority of published work in meta-learning focuses on settings where the task distribution is fixed. This does not model the lifelong learning capabilities of humans.

Online meta-learning (OML) was developed to enable lifelong learning capabilities similar to humans. It connects two distinct fields of machine learning - meta-learning and online learning. In online learning, the task data arrive sequentially and it is drawn from a time-varying task data distribution [3], [16]. Thus, an agent learns a good meta-initialization in a sequential manner that can quickly adapt to new tasks [17]. Most variants of OML algorithm are based on online convex optimization approaches, such as online mirror descent and online gradient descent [18], [19]. Although these algorithms can support continual lifelong learning, a single agent has to learn so many tasks leading to a cold-start problem [19].

Decentralized meta-learning allows multi-agents spatially distributed within a task environment to collaborate among themselves. Each agent only needs to learn from its local task data and share its local model with other agents in its neighborhood [20], [21], [22], [23], [24]. This way, learning is fast and the cold-start problem is eliminated [19]. Convergence is better in multi-agent meta-learning than in single-agent meta-learning. Similar to decentralized networks are distributed networks based on a server-client architecture. Decentralized learning avoids the problem of congestion and a single point of failure that occur in distributed (or centralized) learning where there is a central coordinator coordinating the interactions among the agents [25]. There are few works on distributed meta-learning, some of which are applied to federated learning [26], [27], [28]. Since these works do not incorporate online learning, they are not capable of continuous lifelong learning even though they address the cold-start problem.

Online decentralized meta-learning algorithms can allow multiple agents to cooperatively learn a meta-initializer in a sequential manner and can also address the cold-start problem [19]. However, these algorithms do not provide resiliency against malicious attacks from Byzantine agents in the network [29]. When each agent receives task-related information from its neighbors, the agent does not screen the information before using it to update its local model. Thus, the network becomes susceptible to malicious attacks. Therefore, the existing online decentralized meta-learning approach does not accurately model the human brain, which acts like a natural filter, screening out fake information received from the environment while engaging in continuous lifelong learning.

One of the interesting areas of research in distributed and decentralized networks is providing resiliency against Byzantine attacks [30], [31], [32], [33], [34], [35]. Non-Byzantine agents are said to be normal or non-faulty. A single Byzantine agent can hazardously disrupt a network and preclude convergence. Byzantine agents are hard to decipher because they act uncertainly. Their behavior is difficult to predict [25]. Some state-of-the-art Byzantine-resilient algorithms have been designed to provide resiliency against Byzantine attacks in decentralized learning. These algorithms are ByRDiE [36] and BRIDGE [37]. They work by screening each coordinate of the received local model vectors for outliers during model aggregation. Thus, these algorithms are adaptations of the scalar-based coordinate-wise screening technique [38]. ByRDiE and BRIDGE are not computationally efficient for high-dimensional data, since they screen a coordinate at a time. However, in BRIDGE, the computational efficiency can be improved by screening every coordinate simultaneously with parallel computing [39]. It is worth noting that, ByRDiE and BRIDGE are not meta-learning algorithms, and cannot adapt quickly to new tasks with a small-sized test dataset. Moreover, ByRDiE and BRIDGE algorithms cannot work when agents' data distributions are both heterogeneous and time-varying. To the best of our knowledge, there is no existing online meta-learning algorithm that can overcome Byzantine attacks in time-varying environments, such as autonomous vehicles, where the model must be continuously updated with fewer sample sizes as data keeps arriving.

A. OBJECTIVE AND MOTIVATION FOR THIS RESEARCH

In this research work, our overarching research objective is to design a Byzantine-resilient online meta-learning algorithm for multi-agent systems with better performance than state-of-the-art online meta-learning algorithms. The motivation for this research is discussed as follows. Distributed and decentralized learning algorithms give a good training model when agents have independent and identically distributed (i.i.d) data distributions [40]. However, in many real-world applications, such as autonomous vehicles, intelligent robots, etc., agents have non-i.i.d data distributions and must collaborate in a time-varying environment. Also, some of these agents may be malicious. Therefore, we ask the question *can we develop a robust decentralized learning algorithm that converges quickly to the optimal model for real-time applications with time-varying and heterogeneous data distribution in the presence of Byzantine agents?* This is an important question in machine learning, faced with the urgent need to provide fast adaptation, continuous lifelong learning, and security. We answer in the affirmative by proposing a Byzantine-resilient online decentralized meta-learning algorithm that works with two Byzantine-resilient aggregation techniques.

B. NOVELTY OF THIS RESEARCH

This work aims to overcome the limitation of the coordinate-wise screening used in state-of-the-art Byzantine resilient algorithms, such as ByRDiE [36] and BRIDGE [37]. Therefore, we propose a Byzantine-resilient online decentralized meta-learning algorithm that can work with two superior Byzantine-resilient aggregation techniques. The first Byzantine-resilient aggregation technique is a modification

of the coordinate-wise screening technique. It improves performance by allowing every normal agent in a directed graph network topology to scale its local model with its self-loop weight rather than scaling the local model with a uniform weight. The intuition behind our approach is that since a normal agent cannot identify a Byzantine neighbor, then it should rely more on its local model and less on the received local models from its neighbors, even after the removal of outliers. This will minimize the effects of undetected Byzantine infiltration that escapes the screening process. The second Byzantine-resilient aggregation technique is the centerpoint aggregation. Unlike coordinate-wise screening, centerpoint aggregation is vector-based. It can completely remove the vectorial contributions of Byzantine agents. It has proven to be an effective method for removing outliers in discrete mathematics [41], [42], [43], yet it is still unknown in machine learning. The centerpoint theorem is a generalization of the median to data in higher dimensions. We have successfully applied centerpoint aggregation in our recent work to provide Byzantine resiliency in online federated learning, but with only empirical results provided [44]. However, the performance of the centerpoint aggregation in online decentralized meta-learning settings is yet to be investigated.

Our proposed algorithm does not need to be aware of the identities of the Byzantine agents or the exact number of Byzantine agents in the network, and there is no restriction on the behavior of the Byzantine agents. The only available information, similar to other works in this research area, is to know the upper bound on the number of Byzantine agents in the network [25]. Succinctly, the contributions of this paper are the following

- This work proposes a Byzantine-resilient online decentralized meta-learning algorithm that works with two Byzantine-resilient aggregation techniques to provide fast adaptation, continuous lifelong learning, and security against Byzantine attacks for real-time applications with time-varying data distributions.
- The first Byzantine-resilient aggregation technique is a modification of the coordinate-wise screening used in state-of-the-art decentralized learning algorithms, such as BRIDGE [37]. This modification scales the local model of a normal agent with its self-loop weight during the update process. By utilizing the self-loop weight, each normal agent is allowed to trust its own local model more than what it receives from its screened neighbors. This reduces the effects of undetected Byzantine infiltration and improves convergence.
- The second proposed Byzantine-resilient aggregation technique is the centerpoint aggregation based on the centerpoint theorem in discrete mathematics. Centerpoint aggregation can completely remove Byzantine local model vectors, unlike scalar-based coordinate-wise screening. Thus, the centerpoint aggregation is computationally efficient and performs better than coordinate-wise screening.
- This work provides both theoretical and simulation results to show that the proposed Byzantine-resilient online meta-learning algorithm performs better than existing algorithms.

The remainder of the paper is organized as follows: Section II discusses related works; Section III discusses the preliminaries; Section IV presents the problem formulation; Section V discusses the proposed algorithm; Section VI presents the theoretical results; Section VII discusses the simulation setup and results; Section VIII provides an elaborate discussion on other urgent problems faced in decentralized learning; and Section IX concludes the findings in the paper.

II. RELATED WORKS

Decentralized meta-learning was proposed to allow multiple agents with local data scattered in different locations to collaboratively learn a good meta-initializer. Learning performance improves over a single agent with access to only its local data. Due to a diffusion of the learning parameters across the agents, the cold-start problem is eliminated [19]. Some of the work on decentralized meta-learning include [22], [23]. Decentralized meta-learning was extended to the online decentralized meta-learning setting to also provide continuous lifelong learning [19]. However, both the offline and online decentralized meta-learning algorithms are vulnerable to security threats common to any decentralized network. These algorithms are not designed to withstand security threats. This is a serious concern because cyberattacks impact the performance of machine learning algorithms [52].

There are some techniques proposed in the literature to provide resilience against Byzantine attacks in machine learning. These techniques include Krum, Multi-Krum [32], geometric mean [31], geometric median [45], coordinate-wise screening [38], Zeno [46], Zeno++ [47], etc. These techniques were originally developed for distributed learning based on a worker-master architecture. The worker-master architecture is liable to congestion and a single point of failure at the coordinator [53]. Aside from coordinate-wise screening, the other techniques have not been extended to decentralized multi-agent settings, where multiple agents communicate in a peer-to-peer fashion. This is because these other techniques do not guarantee good resilience or provide good convergence analysis in decentralized learning [25]. In coordinate-wise screening, each normal agent screens the local models it receives from its neighbors and trims out small and large values per coordinate.

Recently, two coordinate-wise screening-based algorithms, ByRDiE [36] and BRIDGE [37] were proposed to guarantee Byzantine resiliency in decentralized learning. In the BRIDGE algorithm, the screened operation occurs in parallel, which results in better computational efficiency than the ByRDiE algorithm. However, both ByRDiE and BRIDGE algorithms use scalar-wise operations per coordinate. A Byzantine contribution can be trimmed in a coordinate but passes the screening test in other coordinates.

TABLE 1. Comparison of byzantine-resilient aggregation techniques for distributed and decentralized learning.

Algorithm	Structure	Data Distribution	Structure	Quick Adaptation	bound on Byzantine agents	Convergence rate
Krum & MultiKrum [32]	distributed	i.i.d	offline	no	$K \geq 2b+3$	$O(n^2d), n = \max_k \lvert\mathcal{N}_k\rvert$
Geometric mean [31]	distributed	i.i.d	offline	no	$K \geq 2b+1$	$O(nd\log^3\frac{n}{\gamma}), \gamma \geq 0$
Geometric median [45]	distributed	i.i.d	offline	no	$K \geq 2b+1$	$O(nd+bd\log^3\frac{1}{\gamma}), \gamma \geq 0$
Coordinate-wise trimmed mean [38]	distributed	i.i.d	offline	no	$K \geq 2b+1$	$O(nd)$
Zeno [46]	distributed	i.i.d	offline	no	$K \geq b+1$	$O(nd)$
Zeno++ [47]	distributed	i.i.d	offline	no	$K \geq b+1$	$O(nd)$
ByRDiE [36]	decentralized	non-i.i.d	offline	no	$K \geq b+3$	$O(n^2d)$
BRIDGE [37]	decentralized	non-i.i.d	offline	no	$K \geq b+3$	$O(n^2d)$
BREDA [48]	decentralized	non-i.i.d	offline	no	$b \geq \frac{n}{3}$	$O(n^2d)$
TV-Norm [49]	decentralized	non-i.i.d	offline	no	$M \geq b+1$	N/A
UBAR [50]	decentralized	non-i.i.d	offline	no	arbitrary	$O(nd)$
BASIL+ [51]	decentralized	non-i.i.d	offline	no	$z \geq b+1, z = \frac{K}{G}, G \geq 0$	$O(nd)$
Proposed Algorithm (Centerpoint)	decentralized	non-i.i.d	online	yes	$b < \lfloor\frac{n}{d+1}\rfloor$	$O(1/t)$ (regret bound)

Thus, both ByRDiE and BRIDGE cannot guarantee the total removal of Byzantine contributions in every coordinate. Hence, they cannot guarantee better convergence towards the global optimal model. Moreover, ByRDiE and BRIDGE are not computationally efficient with high dimensional data. BREDA was also proposed using coordinate-wise screening for Byzantine-resilient resource allocation in decentralized networks [48]. It inherits the limitations of coordinate-wise screening. Aside from coordinate-wise screening, a total variation norm-penalized (TV-norm) approximation method was proposed to handle Byzantine attacks in [49]. However, the limitation is that the optimal local models of the agents are not necessarily consensual.

The Byzantine-resilient algorithms discussed in the previous two paragraphs are distance-based because they identify Byzantine contributions by finding outliers with large distances from a reference point. These algorithms are mostly designed for i.i.d data distribution. Although, BRIDGE and ByRDiE have shown good performance with non-i.i.d data distribution. In order to handle non-i.i.d data distribution while guaranteeing Byzantine resiliency in decentralized networks, UBAR was proposed in [50]. UBAR is a performance-based approach that screens out Byzantine local model vectors based on their performance on a validation dataset. Similarly, BASIL was proposed for non-i.i.d data distribution and achieved a better performance than UBAR [51]. However, BASIL trains in a sequential manner over a logical ring, so it suffers from high latency that scales with the number of agents in the network.

It is noteworthy that none of the aforementioned Byzantine-resilient algorithms can work in a time-varying environment. Hence, this limits their application to many real-world applications, such as autonomous vehicles and traffic monitoring systems. Moreover, none of these algorithms can quickly adapt to a new task with a small dataset. Therefore, we propose a novel Byzantine-resilient algorithm that can work with two superior Byzantine-resilient techniques in a time-varying environment with non-i.i.d data distribution. The first Byzantine-resilient technique is a modification of the coordinate-wise screening but with better performance. The second algorithm uses a superior screening technique called centerpoint aggregation. Centerpoint aggregation is a vector-wise operation, that can screen out Byzantine vectorial infiltration. The centerpoint aggregation is based on the centerpoint theorem in discrete mathematics known for a long time [41], [42], [43]. However, its application in machine learning to provide resiliency against Byzantine attacks is new. Recently, it was applied to provide Byzantine resiliency in robotics [54], [55]. Similarly, we applied centerpoint aggregation in online federated learning for the first time to provide Byzantine resiliency, though with no rigorous experimental and theoretical results [44]. Table 1 shows a comparison of our proposed algorithm with other existing algorithms with K as the total number of agents, b as a known upper bound on the number of Byzantine agents, d as the dimension of the decision set, and n as the unknown number of normal agents. It can be seen from Table 1 that our centerpoint-based proposed meta-learning algorithm is the only Byzantine-resilient algorithm that works in the online setting and can also guarantee quick adaptation when the size of the dataset is small.

III. PRELIMINARIES

A. NOTATIONS AND DEFINITIONS

Table 2 defines the notation used in the paper.

Definition 1: (strong convexity) A function f is said to be $\lambda-$strongly convex in its first argument if $\forall x, y \in \mathbb{R}^d$, we have

$$f(x) \geq f(y) + \langle \nabla f(y), x-y\rangle + \frac{\lambda}{2}||x-y||^2. \tag{1}$$

Definition 2: (Lipschitz continuity) A function f is said to be $L-$Lipschitz continuous $\forall x, y \in \mathbb{R}^d$, if

$$|f(x) - f(y)| \leq L||x-y||. \tag{2}$$

This implies that the gradient is bounded, i.e., $||\nabla f(\omega)|| \leq L$ and $L > 0$.

TABLE 2. Notations.

Symbols	Definitions
$\mathbb{R}^d$	$d-$dimensional real vector space
$\|\|\cdot\|\|$	norm
$\|\|\cdot\|\|_2$	Euclidean norm
$\mathcal{G}$	graph network
A	adjacency matrix
$\mathcal{N}_k$	neighborhood of agent k
f	convex function
∇f	gradient
$\nabla^2 f$	Hessian
$x \in \mathbb{R}^d$	vector
$\mathbb{E}[\cdot]$	expectation of a random variable
$x^{\mathbb{T}}$	transpose of vector x
$\langle \cdot, \cdot \rangle$	inner product
$\|\cdot\|$	cardinality of a set

Assumption 1: (smoothness) The gradient ∇f is $L'-$smooth $\forall x, y \in \mathbb{R}^d$ if

$$||\nabla f(x) - \nabla f(y)|| \leq L'||x - y||, \tag{3}$$

where $L' > 0$.

Definition 3: (co-coercivity) A convex and continuously differentiable function f is said to undergo the co-coercivity condition if

$$\langle \nabla f(x) - \nabla f(y), x - y \rangle \geq \frac{1}{L}||\nabla f(x) - \nabla f(y)||^2. \tag{4}$$

Definition 4: (general position) A set containing at least $d + 1$ points in $d-$dimensional affine space is said to be in a general position if no hyperplane contains more than d points, i.e., the points do not satisfy any more linear relations than they must.

Definition 5: (interior point) A point $x \in \mathbb{R}^d$ is a interior point of the set $\mathcal{S} \subset \mathbb{R}^d$, if there exist an open ball centered at x which is completely contained in $\mathcal{S}$.

Definition 6: (half-space) A half-space is either of the two parts into which a hyperplane divides an affine space. A closed half-space is a set of the form $\{x \in \mathbb{R}^d : a^{\mathbb{T}}x \geq b\}$, where $a \in \mathbb{R}^d \backslash \{0\}$.

Definition 1, Definition 2, Assumption 1, and Definition 3 are standard for convex optimization and are necessary for the proof of convergence in this work. For a better understanding of convex optimization, we refer readers to [39] and [56]. Definition 4, Definition 5, and Definition 6 are well known in computational geometry but are necessary here to analyze centerpoint aggregation [54].

B. OFFLINE DECENTRALIZED META LEARNING

Assume a decentralized multi-agent setting where the agents are connected to form a directed graph network $\mathcal{G} = (\mathcal{K}, \mathcal{E})$ with $\mathcal{K} = \{1, \ldots, K\}$ representing the set of agents and $\mathcal{E}$ representing the set of edges. The edge $(j, k) \in \mathcal{E}$ if agent j can send one-way information directly to agent k and the edge $(k, j) \in \mathcal{E}$ if agent k can send one-way information directly to agent j. The adjacency matrix A for the graph network is formed from the edge weights between any two agents. That is, $[A]_{jk}$ is the weight of edge (j, k) and $[A]_{kk}$ is the self-loop weight of agent k. The matrix is column-stochastic, which means that the sum of its column must be one. The neighbors of agent k are members of the set $\mathcal{N}_k := \{j \in \mathcal{K} : (j, k) \in \mathcal{E}\}$. The objective of the agents is to collaboratively learn the best model as follows [57]

$$\min_{\omega \in \mathbb{R}^d} \sum_{k=1}^{K} f_k(\omega), \tag{5}$$

where $f_k(\omega) := \mathbb{E}_{d_k \sim D_k} F_k(\omega; d_k)$ is the local stochastic risk of agent k. The loss $F_k(\omega; d_k)$ is the penalization of the model ω on data sample d_k drawn randomly from data distribution D_k for agent k. The expectation is over the randomness of d_k.

In offline decentralized MAML, the goal is not to find the best model for the agents. Rather, MAML exploits the relatedness among the underlying task data distributions of the agents to find a meta-initializer that can quickly adapt to new tasks arriving for each agent after one or more gradient descent updates [12]. We will restrict our analysis to a single gradient descent update, although, this can be extended easily to multiple gradient descent updates. Using a single gradient descent update, the agents' objective is formulated as follows [27]:

$$\min_{\omega \in \mathbb{R}^d} \sum_{k=1}^{K} J_k(\omega) := \min_{\omega \in \mathbb{R}^d} \sum_{k=1}^{K} f_k(\omega - \alpha \nabla f_k(\omega)), \tag{6}$$

where meta function $J_k(\omega) := f_k(\omega - \alpha \nabla f_k(\omega))$ and $\alpha > 0$ is the step-size. Any agent can take the solution of Equation (6) as an initial point and updates it with respect to its own task data. Equation (6) can be seen as the sum of the meta functions $J_1, \ldots, J_K$. The gradient $\nabla J_k(\omega)$ is given as follows

$$\nabla J_k(\omega) = (I - \alpha \nabla^2 f_k(\omega)) \nabla f_k(\omega - \alpha \nabla f_k(\omega)). \tag{7}$$

IV. PROBLEM FORMULATION

In this section, we will start by formulating the problem of online decentralized meta-learning, which differs from the offline decentralized meta-learning discussed in the preliminaries. Then, we will extend our formulation to the case where there are Byzantine agents in the graph network.

A. ONLINE DECENTRALIZED META LEARNING

In online learning, an agent k is faced with a sequence of global stochastic risk functions $f_1, \ldots, f_T$ over a period of T iterations. The risk function sequence is chosen adversarially from an unknown distribution. Each agent k must choose the local models $\omega_{k,1}, \ldots, \omega_{k,T}$ that perform well on this risk function sequence. In essence, the goal of the agent is to minimize a performance metric called regret. A single agent's regret is defined as the difference between the cumulative risk of the agent from time $t = 1$ to $t = T$ and the cumulative risk of an oracle with hindsight knowledge of the minimizer (or best model) from time $t = 1$ to $t = T$. Formally, a single

agent's regret is represented as [58]

$$Reg_T = \sum_{t=1}^{T} f_t(\omega_{k,t}) - \min_{\omega \in \mathbb{R}^d} \sum_{t=1}^{T} f_t(\omega). \tag{8}$$

The regret definition in (8) does not solve the online optimization problem in a decentralized fashion for the multi-agents $\mathcal{K} = \{1, \ldots, K\}$. The agents can leverage their graph connections to collaboratively learn the minimizer as follows

$$Reg_T^K = \sum_{t=1}^{T} \sum_{k=1}^{K} f_{k,t}(\omega_{k,t}) - \min_{\omega \in \mathbb{R}^d} \sum_{t=1}^{T} \sum_{k=1}^{K} f_{k,t}(\omega), \tag{9}$$

where $f_{k,t}(\omega) := \mathbb{E}_{d_{k,t} \sim D_{k,t}} F_{k,t}(\omega; d_{k,t})$ is the time-varying local risk function of the agent k, and the expectation is over a time-varying data sample $d_{k,t}$ drawn from a time-varying data distribution $D_{k,t}$. It is often better to compute the expected regret of the agents instead of the instantaneous regret of the agents. Therefore, Equation (9) can be reformulated as follows

$$Reg_T^K = \mathbb{E}\left[\sum_{t=1}^{T} \sum_{k=1}^{K} f_{k,t}(\omega_{k,t}) - \min_{\omega \in \mathbb{R}^d} \sum_{t=1}^{T} \sum_{k=1}^{K} f_{k,t}(\omega) \right]. \tag{10}$$

The expectation is over the randomness of the local risk function $f_{k,t}$. We can consider an online decentralized meta-learning setting where each agent performs a task-specific update to its local model before it is used for evaluation in each iteration. The goal is no longer to learn the minimizer but to learn a meta-initializer. This update is done using an update function such that $U_{k,t} : \omega \to \hat{\omega}$. The update function takes ω and produces $\hat{\omega}$ which performs better on $f_{k,t}$. The update function $U_{k,t}$ is a one step of stochastic gradient descent, i.e., $U_{k,t}(\omega) = \omega - \alpha \nabla f_{k,t}(\omega)$ [3]. Therefore, equation (10) becomes

$$Reg_{T,meta}^K = \mathbb{E}\Bigg[\sum_{t=1}^{T} \sum_{k=1}^{K} f_{k,t}(U_{k,t}(\omega_{k,t})) - \min_{\omega \in \mathbb{R}^d} \sum_{t=1}^{T} \sum_{k=1}^{K} f_{k,t}(U_{k,t}(\omega)) \Bigg]. \tag{11}$$

The online meta function $J_{k,t}(\omega) := f_{k,t}(U_{k,t}(\omega))$ and its gradient $\nabla J_{k,t}(\omega) = (I - \alpha \nabla^2 f_{k,t}(\omega)) \nabla f_{k,t}(\omega - \alpha \nabla f_{k,t}(\omega))$.

B. BYZANTINE AGENTS AND ATTACKS

Cooperation among agents in the neighborhood is essential in multi-agent settings to converge to the minimizer. However, some of the agents in the neighborhood may deviate arbitrarily from the expected behavior during the iterative training process. Such agents are said to be Byzantine. Byzantine agents refer to any malfunctioning or malicious agents.

Definition 7 ([37]): An agent $j \in \mathcal{K}$ is said to be Byzantine at any iteration of the online decentralized meta-learning process, if it follows a different update rule, say $\tilde{g}_j(\cdot)$ instead of $g_j(\cdot)$, or broadcasts an arbitrary summary of its local information to agents in its neighborhood that is different from the intended summary of its local information.

For the rest of this paper, we denote $\mathcal{M} \subset \mathcal{K}$ as the set of normal agents and $\mathcal{B} \subset \mathcal{K}$ as the set of Byzantine agents, such that $\mathcal{M} \cap \mathcal{B} = \emptyset$ and $\mathcal{M} \cup \mathcal{B} = \mathcal{K}$. Let n represent the cardinality of $\mathcal{M}$. As common in literature on Byzantine resiliency, it is assumed that the Byzantine agents are unknown and hard to identify [37]. Hence, the cardinality and members of $\mathcal{B}$ are unknown. However, a known upper bound on the number of Byzantine agents is denoted as b, i.e., $0 \leq |\mathcal{B}| \leq b$ and $n \geq K - b$.

The presence of Byzantine agents makes (11) unsolvable. Hence, we will focus on training only the normal agents. Thus, (11) is reformulated as follows

$$Reg_{T,meta}^{Byz,K} = \mathbb{E}\Bigg[\sum_{t=1}^{T} \sum_{k=1}^{K-b} f_{k,t}(U_{k,t}(\omega_{k,t})) - \min_{\omega \in \mathbb{R}^d} \sum_{t=1}^{T} \sum_{k=1}^{K-b} f_{k,t}(U_{k,t}(\omega)) \Bigg], \tag{12}$$

where the global meta-initializer $\omega^* := \arg\min_{\omega \in \mathbb{R}^d} \mathbb{E}[\sum_{t=1}^{T} \sum_{k=1}^{K-b} f_{k,t}(U_{k,t}(\omega))]$. In order to develop an online decentralized meta-learning algorithm that can solve (12), the following assumptions will be made.

Definition 8 ([37]): (source component) A source component of a graph is a subset of a graph nodes such that every node in the source component has a directed path towards every other node in the graph

Definition 9 ([37]): (reduced graph) A subgraph $\mathcal{G}_{red}(b)$ of a graph G with parameter b is a reduced graph generated from (i) removing all Byzantine agents together with their incoming and outgoing edges, and (ii) removing b edges from each normal agents.

Assumption 2 ([37]): (sufficient network connectivity) The reduced graphs $\mathcal{G}_{red}(b)$ of the graph network $\mathcal{G} = (\mathcal{K}, \mathcal{E})$ give rise to a sufficiently connected decentralized network in the sense that there exists at least one source component among the reduced graphs of cardinality greater than $(b + 1)$.

Assumption 2 ensures that each normal node can still receive and send information to other normal nodes despite the removal of Byzantine nodes and edges from the graph network. Empirical findings show that this assumption is valid [37].

V. PROPOSED ALGORITHM

In practice, there is always a lack of information about the time-varying data distribution $D_{k,t}$ for any agent k in an online learning setting. This makes computing $f_{k,t}(\omega)$, and in essence computing both $\nabla f_{k,t}(\omega)$ and $\nabla J_{k,t}(\omega)$ computationally infeasible. Hence, data realizations are usually collected and an empirical risk minimization is computed. Thus, we can estimate the gradient $\nabla f_{k,t}(\omega)$ as follows

$$\nabla \bar{f}_{k,t}(\omega; S_{k,t}) := \frac{1}{|S_{k,t}|} \sum_{n=1}^{|S_{k,t}|} \nabla F_k(\omega; d_{k,t}^n), \tag{13}$$

where $S_{k,t}$ is a time-varying independent mini-batch of data containing realizations $\{d_{k,t}^n\}_{n=1}^{|S_{k,t}|}$. The gradient estimate $\nabla\bar{f}_{k,t}(\omega; S_{k,t})$ is an unbiased version of the true gradient $\nabla f_{k,t}(\omega)$. Similarly, the Hessian in $\nabla J_{k,t}$ can be approximated with its unbiased estimate $\nabla^2\bar{f}_{k,t}(\omega, S_{k,t})$. Therefore, the approximate online meta gradient $\nabla\bar{J}_{k,t}(\omega)$ is computed as follows

$$\nabla\bar{J}_{k,t}(\omega; S_{k,t}, S'_{k,t}) = (I - \alpha\nabla^2\bar{f}_{k,t}(\omega, S_{k,t}))\times \nabla\bar{f}_{k,t}(\omega - \alpha\nabla\bar{f}_{k,t}(\omega; S_{k,t}); S'_{k,t}), \quad (14)$$

where $S_{k,t}$ and $S'_{k,t}$ are two independent mini-batch data. Computing the Hessian every iteration can be computationally challenging. Therefore, the Hessian part in $\nabla\bar{J}_{k,t}(\omega; S_{k,t}, S'_{k,t})$ is often removed to obtain an approximate online meta gradient given as $\nabla\bar{f}_{k,t}(\omega - \alpha\nabla\bar{f}_{k,t}(\omega; S_{k,t}); S'_{k,t})$ [59]. It is shown in [12] that during the local model update, $\nabla\bar{f}_{k,t}(\omega - \alpha\nabla\bar{f}_{k,t}(\omega; S_{k,t}); S'_{k,t})$ can be implemented as two different stochastic gradient descent steps, i.e., $\omega_{k,t+1} \leftarrow \omega_{k,t} - \alpha\nabla\bar{f}_{k,t}(\omega - \alpha\nabla\bar{f}_{k,t}(\omega_{k,t}; S_{k,t}); S'_{k,t})$ can be updated in two steps as follows: $\tilde{\omega}_{k,t+1} \leftarrow \omega_{k,t} - \alpha\nabla\bar{f}_{k,t}(\omega_{k,t}; S_{k,t})$ and $\omega_{k,t+1} \leftarrow \omega_{k,t} - \alpha\nabla\bar{f}_{k,t}(\tilde{\omega}_{k,t+1}; S'_{k,t})$. However, updating the local model $\omega_{k,t+1}$ this way does not include collaborative learning with other agents in the neighborhood set $\mathcal{N}_k$.

To allow for collaborative learning, updating $\omega_{k,t+1}$ follows a diffusion learning process [60]. The conventional diffusion learning process uses the adapt-then-combine (ATC) technique. However, this diffusion learning process is not robust against Byzantine attacks. Hence we propose an adapt-meta-adapt-then-robust-combine (AMATRC) diffusion learning process with time-varying step sizes described as follows:

$$\tilde{\omega}_{k,t+1} \leftarrow \omega_{k,t} - \alpha_t\nabla\bar{f}_{k,t}(\omega_{k,t}, S_{k,t}), \textit{ adapt}$$

$$\theta_{k,t+1} \leftarrow \omega_{k,t} - \alpha_t\nabla\bar{f}_{k,t}(\tilde{\omega}_{k,t+1}S'_{k,t}), \quad \textit{meta} - \textit{adapt}$$

$$\omega_{k,t+1} \leftarrow [A]_{kk}\theta_{k,t+1} + \textit{Byzascreen}(\{\theta_{j,t+1}, [A]_{jk}\}_{j\in\mathcal{N}_K}), \quad \textit{robust} - \textit{combine}.$$

Algorithm 1 Differentially Private Personalized Online Federated Learning Algorithm Run by Each Client

1: **Input**: Time duration T, time-varying step sizes $\alpha_t > 0$.
2: **Output:** $\omega_{k,T}$.
3: Initialization: $\omega_{k,1}$
4: **for** $t = 1, \ldots, T$ **do**
5: Obtain the estimated risk $\bar{f}_{k,t}(\omega_{k,t}; S_{k,t})$.
6: Compute $\tilde{\omega}_{k,t+1} = \omega_{k,t} - \alpha_t\nabla_{\omega}\bar{f}_{k,t}(\omega_{k,t}, S_{k,t})$.
7: Compute $\theta_{k,t+1} = \omega_{k,t} - \alpha_t\nabla_{\tilde{\omega}}\bar{f}_{k,t}(\tilde{\omega}_{k,t+1}, S'_{k,t})$.
8: Cooperate with neighbors to update the local meta-model $\omega_{k,t+1} = \theta_{k,t+1}[A]_{kk} + \textit{Byzascreen}(\{\theta_{j,t+1}, [A]_{jk}\}_{j\in\mathcal{N}_K})$.

The proposed Algorithm 1 works as follows: Step 1 shows the necessary inputs to the algorithm, which are the time duration T and time-varying step size α_t. The use of time-varying step size rather than a fixed step size is to provide faster convergence. Step 2 shows the expected output of the algorithm, which is the local model $\omega_{k,T}$ of agent k at time T. This local model $\omega_{k,T}$ is an estimate of the global meta-initializer ω^*. Step 3 initializes the local model $\omega_{k,1}$ at the start of the algorithm. From Steps 4 to 8, the algorithm iterates for round $t = 1, \ldots, T$. Step 5 computes the estimated risk. Step 6 is the *adapt* step which computes a stochastic gradient descent update on the local model. Step 7 is the *meta-adapt* step, which is another stochastic gradient descent update. Step 8 is the *robust-combine* step, which uses a *Byzascreen* protocol to screen out Byzantine contributions during the aggregation of the local model of client k with the contributions of its neighbors. Byzascreen protocol is Byzantine-resilient unlike the conventional ATC or the consensus approach in decentralized optimization, which is highly susceptible to Byzantine infiltration [25]. We propose two independent Byzascreen protocols. The first is the modified coordinate-wise screening, which is a modified version of the conventional coordinate-wise screening technique used for removing outliers in distributed and decentralized networks [38]. The second is the centerpoint aggregation based on the centerpoint theorem in discrete geometry but still unknown in machine learning [42].

A. MODIFIED COORDINATE-WISE BYZANTINE SCREENING

The conventional coordinate-wise screening trims the b largest and the b smallest values in each coordinate of the intermediate models $\{\theta_{j,t+1}\}_{j\in\mathcal{N}_k}$ received from neighbors of the normal agent k at time t. Then, it finds the average of the remaining values to update each coordinate of $\omega_{k,t}$ and forms a new local model $\omega_{k,t+1}$ [37]. In our work, we use a modified updating approach that allows each normal client to leverage its self-loop weight for faster convergence. Specifically, in each iteration, the proposed coordinate-wise screening technique computes the following in parallel for each coordinate point $i = \{1, \ldots, d\}$ of the normal agent k [37]:

$$\overline{\mathcal{N}}_{k,t}^i := \underset{\mathcal{X}:\mathcal{X}\subset\mathcal{N}_k, |\mathcal{X}|=b}{\arg\max} \sum_{j\in\mathcal{X}} [\theta_{j,t+1}]_i \quad (15)$$

$$\underline{\mathcal{N}}_{k,t}^i := \underset{\mathcal{X}:\mathcal{X}\subset\mathcal{N}_k, |\mathcal{X}|=b}{\arg\min} \sum_{j\in\mathcal{X}} [\theta_{j,t+1}]_i \quad (16)$$

$$\mathcal{C}_{k,t}^i := \mathcal{N}_k \backslash \{\overline{\mathcal{N}}_{k,t}^i \bigcup \underline{\mathcal{N}}_{k,t}^i\}. \quad (17)$$

It should be noted that the member of sets $\overline{\mathcal{N}}_{k,t}^i$ and $\underline{\mathcal{N}}_{k,t}^i$ are not fixed but are time-dependent due to the arbitrary behavior of Byzantine attacks. It is possible for a neighbor of agent k to belong to set $\overline{\mathcal{N}}_{k,t}^i$ or $\underline{\mathcal{N}}_{k,t}^i$ at coordinate i but does not belong to any of these sets in other coordinates at a given iteration. After the completion of (15), (16), and (17), the update process is done in parallel for $\forall i = \{1, \ldots, d\}$

as follows

$$[\omega_{k,t+1}]_i := [A]_{kk}[\theta_{k,t+1}]_i + \frac{(1-[A]_{kk})}{|\mathcal{N}_k|-2b}\sum_{j\in\mathcal{C}^i_{k,t}}[\theta_{j,t+1}]_i. \tag{18}$$

The normal agent k leverages its self-loop weight in (18) unlike in the conventional coordinate-wise screening technique. The edge weight for each member of the set $\mathcal{C}^i_{k,t}$ is $\frac{(1-[A]_{kk})}{|\mathcal{N}_k|-2b}$ to ensure $\sum_{j\in\mathcal{C}^i_{k,t}\cup\{k\}}[A]_{jk} = 1 \quad \forall i \in \{1,\ldots,d\}$. The modified coordinate-wise Byzantine screening process requires that each agent k has at least $2b+1$ neighbors. *Since the modified coordinate-wise screening process is not vector-wise, there is no guarantee that this screening process will completely screen out a Byzantine neighbor's local model contribution in all coordinates at any iteration.*

B. CENTERPOINT AGGREGATION

The centerpoint aggregation is a vector-wise screening technique for outliers, well known in discrete geometry. It is a generalization of the median to higher dimensions. It states that for any point set P of z points in $\mathbb{R}^d$, there exists a point c such that the halfspace containing c contains not less than $\frac{z}{d+1}$ points of P [61]. Recently, the centerpoint aggregation technique was applied to provide resilient aggregation against Byzantine agents in the consensus ATC algorithm [54]. Also, our recent work applied centerpoint aggregation technique to provide Byzantine resiliency in online federated learning for the first time [44]. However, we provided only empirical results without experimentation or convergence analysis. Moreover, our previous work does not discuss meta-learning. Hence, we provide rigorous analysis and experimentation for the online decentralized meta-learning setting discussed in this paper.

A *safe point*, which is an *interior centerpoint*, exists when the number of Byzantine agents in the neighborhood of any normal agent k is less than $b = \frac{|\mathcal{N}_k|}{d+1}$, where $|\mathcal{N}_k|$ is the size of neighborhood set $\mathcal{N}_k$, and d is the dimension of the local model [55]. In our work, we show that *the centerpoint aggregation technique ensures that a centerpoint lies in the convex hull of points corresponding to the received intermediate model vectors of the screened neighbors of agent k*. This does not require a knowledge of the location and behavior of the Byzantine agents. However, computing the centerpoint can be challenging in practice for large dimensions. There are approximate algorithms in the literature to compute a centerpoint that lies in the interior of the convex hull of the normal agents within computational time $O((rd)^d)$, given that there are at most $\frac{|\mathcal{N}_k|}{d^{r/r-1}}$ Byzantine agents, where r is an integer greater than 1. Increasing the value of r makes the approximation better [62]. A more feasible approximate algorithm proposed is the Tverberg partitioning that finds an approximate centerpoint [63], [64].

Definition 10 ([44]): Given a set P of z in $\mathbb{R}^d$ in general position, where $z \geq d+1$, a centerpoint c is a point, not necessarily a member of P, such that any closed half-space containing c will also contain at least $\lceil\frac{z}{d+1}\rceil$ points from P. The value of z is same as the cardinality $|\mathcal{N}_k|$.

Remark 1: The centerpoint divides the set P into roughly two equal halves, such that sufficient points exist on both sides of the centerpoint.

Theorem 1 ([44], [65]): There exists a centerpoint for any given point set in general positions in any arbitrary dimension. This is the centerpoint theorem.

Remark 2: There are more than one centerpoints. The centerpoint is not unique. The set of these centerpoints is called the centerpoint region, and it is closed and convex.

Lemma 1 ([44]): Given a set of z neighbors where each transmits $d-$ dimensional local model vector, out of which there are $|\mathcal{B}|$ unknown Byzantine agents, then a centerpoint exists that lies in the interior of the convex hull of the normal agents, if and only if $|\mathcal{B}| < \lfloor\frac{z}{d+1}\rfloor$.

Theorem 2 ([44]): For a set of z neighbors in general positions in $\mathbb{R}^d$, if the number of Byzantine agents is less than $\lfloor\frac{z}{d+1}\rfloor$, then there is always a safe point.

VI. THEORETICAL RESULTS

In this section, we will discuss the theoretical guarantee for the proposed Algorithm using the centerpoint aggregation scheme.

Lemma 2: Algorithm 1 is said to be resilient convergent if for every $t \in T$ and $k \in \mathcal{M}$ we have that $||\omega_{k,t+1} - \omega^*|| \leq ||\omega_{k,t} - \omega^*||$.

Proof: Let the set of normal clients in the neighborhood of agent k with k inclusive, be given as $\mathcal{N}^+_k \subseteq \mathcal{N}_k \cup \{k\}$. It holds from the Algorithm 1 that

$$||\omega_{k,t+1} - \omega^*|| \leq \max_{j\in\mathcal{N}^+_k} ||\theta_{j,t+1} - \omega^*|| \quad \forall t. \tag{19}$$

Let us simplify $||\theta_{j,t+1} - \omega^*||$ before returning back to (19). From step 7 of Algorithm 1

$$||\theta_{j,t+1} - \omega^*|| = ||\omega_{j,t} - \alpha_t \nabla_{\tilde{\omega}} \bar{f}_{j,t}(\tilde{\omega}_{j,t}, S'_{j,t}) - \omega^*||, \tag{20}$$

substituting step 6 of Algorithm 1 in (20) gives

$$= ||\omega_{j,t} - \alpha_t \nabla_{\omega} \bar{f}_{j,t}(\omega_{j,t} - \alpha_t \nabla_{\omega} \bar{f}_{j,t}(\omega_{j,t}, S_{j,t}), S'_{j,t}) - \vec{\omega}^*||,$$

let $U_{j,t}(\omega_{j,t}, S_{j,t}) := \omega_{j,t} - \alpha_t \nabla_{\omega} \bar{f}_{j,t}(\omega_{j,t}, S_{j,t})$, then from (20) we have

$$||\theta_{j,t+1} - \omega^*|| = ||\omega_{j,t} - \alpha_t \nabla_{\omega} \bar{f}_{j,t}(U_{j,t}(\omega_{j,t}, S_{j,t}), S'_{j,t}) - \omega^*||,$$

the empirical online meta function $\bar{f}_{j,t}(U_{j,t}(\omega_{j,t}, S_{j,t}), S'_{j,t})$ is an approximation of stochastic online meta function $J_{k,t}(\omega_{k,t}) := f_{j,t}(U_{j,t}(\omega_{j,t}))$. Thus, in approximation, $||\theta_{j,t+1} - \omega^*||$ can be written as

$$||\theta_{j,t+1} - \omega^*|| \approx ||\omega_{j,t} - \alpha_t \nabla_{\omega} f_{j,t}(U_{j,t}(\omega_{j,t})) - \omega^*||.$$

We can express $||\omega_{j,t} - \alpha_t \nabla_{\omega} f_{j,t}(U_{j,t}(\omega_{j,t})) - \omega^*||^2$ as

$$\begin{aligned} &||\omega_{j,t} - \alpha_t \nabla_{\omega} f_{j,t}(U_{j,t}(\omega_{j,t})) - \omega^*||^2 \\ &= ||\omega_{j,t} - \omega^*||^2 - 2\alpha_t \nabla_{\omega} f_{j,t}(U_{j,t}(\omega_{j,t}))^{\mathbb{T}}(\omega_{j,t} - \omega^*) + \\ &\alpha_t^2 ||\nabla_{\omega} f_{j,t}(U_{j,t}(\omega_{j,t}))||^2. \end{aligned} \tag{21}$$

From the co-coercivity of $f_{j,t}(\cdot)$ in Definition 3, we have

$$(\nabla_\omega f_{j,t}(U_{j,t}(\omega_{j,t})) - \nabla_\omega f_{j,t}(U_{j,t}(\omega^*)))^{\mathbb{T}}(\omega_{j,t} - \omega^*) \geq \frac{1}{L}||\nabla_\omega f_{j,t}(U_{j,t}(\omega_{j,t})) - \nabla_\omega f_{j,t}(U_{j,t}(\omega^*))||^2. \quad (22)$$

However, $\nabla_\omega f_{j,t}(U_{j,t}(\omega^*)) = 0$, hence (22) becomes

$$(\nabla_\omega f_{j,t}(U_{j,t}(\omega_{j,t})))^{\mathbb{T}}(\omega_{j,t} - \omega^*) \geq \frac{1}{L}||\nabla_\omega f_{j,t}(U_{j,t}(\omega_{j,t}))||^2. \quad (23)$$

Substitute (23) into (21). This becomes

$$\begin{aligned} &||\omega_{j,t} - \alpha_t \nabla_\omega f_{j,t}(U_{j,t}(\omega_{j,t})) - \omega^*||^2 \\ &\leq ||\omega_{j,t} - \omega^*||^2 - \left(\frac{2\alpha_t}{L} - \alpha_t^2\right)||\nabla_\omega f_{j,t}(U_{j,t}(\omega_{j,t}))||^2 \\ &\leq ||\omega_{j,t} - \omega^*||^2. \end{aligned} \quad (24)$$

Now, substitute (24) into (19) to obtain

$$||\omega_{k,t+1} - \omega^*||^2 \leq \max_{j\in\mathcal{N}_k^+} ||\omega_{j,t} - \omega^*||^2. \quad (25)$$

Since $k \in \mathcal{N}_k^+$, let $\omega_{k,t} := \text{argmax}_{\omega_{j,t}, j\in\mathcal{N}_k^+} ||\omega_{j,t} - \omega^*||$, then from (25), $||\omega_{k,t+1} - \omega^*||^2 \leq ||\omega_{k,t} - \omega^*||^2$, which implies that $||\omega_{k,t+1} - \omega^*|| \leq ||\omega_{k,t} - \omega^*||$.

Remark 3: Lemma 2 shows that every round of iteration of Algorithm 1 improves convergence accuracy by reducing the distance to the global meta-initializer.

Assumption 3: For any normal agent $k \in \mathcal{M}$, there exists constant $c \geq 1$, such that for all $t \in \{1, .., T\}$, $\mathbb{E}[||\nabla \bar{f}_{k,t}(\omega_{k,t}, S_{k,t})||_2^2] \leq \sigma_k^2 + c||\nabla f_{k,t}(\omega_{k,t})||_2^2$.

Remark 4: Assumption 3 bounds the variance of the estimated loss gradient using the variance of the true loss gradient [66], [67]. In expectation, the estimated loss gradient returns the stochastic loss gradient, i.e., $\mathbb{E}[\nabla \bar{f}_{k,t}(\omega_{k,t}, S_{k,t})] = \nabla f_{k,t}(\omega_{k,t})$.

Assumption 4: There exists scalar $\mu > 0$, such that, for all $t \in \{1, \ldots, T\}$, we have $\nabla f_{k,t}(\omega_{k,t})^{\mathbb{T}}\mathbb{E}[\nabla \bar{f}_{k,t}(\omega_{k,t}, S_{k,t})] \geq \mu||\nabla f(\omega_{k,t})||_2^2$.

Remark 5: Assumptions 3 and 4 are used in Theorem 3 to bound the regret [67].

Theorem 3: The expected regret incurred by a normal agent $k \in \mathcal{M}$ running Algorithm 1, with $f_{k,t}$ as a $\lambda-$strongly convex function and the stepsize α_t, such that $\alpha_t = \frac{\beta}{\gamma+t}$ for some $\beta > \frac{1}{\lambda\mu}$, $\gamma > 0$ and $\alpha_1 = \frac{\mu}{L\sigma_k^2}$. Then, the regret is given as

$$\mathbb{E}\left[\sum_{t=1}^{T} f_{k,t}(U_{k,t}(\omega_{k,t})) - \sum_{t=1}^{T} f_{k,t}(U_{k,t}(\omega^*))\right] \leq \sum_{t=1}^{T}\left(\frac{\nu}{t+\gamma}\right).$$

where $\nu := \max\left\{\frac{\beta^2 Lc}{2(\beta\lambda\mu-1)}, (\gamma + t)(f_{k,t}(U_{k,t}(\omega_{k,1})) - f_{k,t}(U_{k,t}(\omega^*))\right\}$.

Proof: It can be obtained from Theorem 4.7 in [67] that

$$\begin{aligned} &\mathbb{E}\left[f_{k,t}(U_{k,t}(\omega_{k,t}) - f_{k,t}(U_{k,t}(\omega^*))\right] \leq \\ &\left(1 - \frac{\beta\lambda\mu}{\gamma+t}\right)\frac{\nu}{\gamma+t} + \frac{\beta^2 Lc}{2(\gamma+t)^2}. \\ &= \left(\frac{\gamma+t-\beta\lambda\mu}{(\gamma+t)^2}\right)\nu + \frac{\beta^2 Lc}{2(\gamma+t)^2} \\ &= \left(\frac{(\gamma+t)-1}{(\gamma+t)^2}\right)\nu - \left(\frac{\beta\lambda\mu-1}{(\gamma+t)^2}\right)\nu \\ &\quad + \frac{\beta^2 Lc}{2(\gamma+t)^2} \leq^{(a)} \frac{\nu}{\gamma+t}, \end{aligned}$$

where in (a), the sum of the second and third terms in the left-hand side of the above equation is negative from the definition of ν and can be removed. Moreover, in the first term $(\gamma + t)^2 \geq ((\gamma + t) - 1)((\gamma + t) + 1)$. Now, summing from $t = 1$ to T gives

$$\mathbb{E}\left[\sum_{t=1}^{T} f_{k,t}(U_{k,t}(\omega_{k,t}) - \sum_{t=1}^{T} f_{k,t}(U_{k,t}(\omega^*))\right] \leq \sum_{t=1}^{T}\left(\frac{\nu}{\gamma+t}\right) \quad (26)$$

Remark 5: The regret bound of Algorithm 1 per round of iteration is of order $O(\frac{1}{t})$ as long as the stepsize parameter $\beta > \frac{1}{\lambda\mu}$.

VII. SIMULATION

A. SIMULATION OVERVIEW

For the simulation, we will use a synthetic dataset similar to existing work [48]. We will show first that the conventional ATC aggregation technique used in decentralized algorithms will help agents converge when there is no Byzantine attack in the network. Then, we will show that the ATC aggregation will cause an oscillation but no convergence in the presence of one Byzantine agent. By increasing the number of Byzantine agents to two, we will show that the ATC aggregation technique will give irrational behavior and never converge. This indicates the need for Byzantine-resilient aggregation techniques. Next, we will compare the convergence of the coordinate-wise aggregation used in state-of-the-art BRIDGE and ByRDiE algorithms with both the proposed modified coordinate-wise screening and centerpoint aggregation for both the case of one and two Byzantine agents in the network. Then, we determine the effect of the learning rate on the speed of convergence. Finally, we compare the regret bound of our proposed meta-learning algorithm with existing online meta-learning algorithms.

B. SIMULATION SETUP

We consider online linear regression where the loss function is $F_{k,t}(\omega_{k,t}; d_{k,t}) := \frac{1}{2}(\langle\omega_{k,t}, d_{k,t}\rangle - y_{k,t})^2 + \frac{\lambda}{2}||\omega_{k,t}||^2$, where $d_{k,t} \in \mathbb{R}^d$ is a data sample and $y_{k,t} \in \mathbb{R}$ is its label. The time-varying mini-batch data $S_{k,t} = (d_{k,t}^n, y_{k,t}^n)_{n=1}^N$, where N is the cardinality of $S_{k,t}$. The element of each coordinate of the data

TABLE 3. Simulation parameters.

Parameters	description
loss function	online linear regression
data type	synthetic
Time duration T	100
learning rate α_t	$\frac{1}{t}$
dimension d	3
regularization λ	1
Gaussian noise Z	$Z \sim \mathcal{N}(0,1)$
number of Byzantine agents b	2
number of agents K	10

sample $d_{k,t}$ is drawn from the interval $(-1, 1)$ and the label is computed as follows

$$y_{u,t} := \langle \omega_0, d_{k,t} \rangle + Z \tag{27}$$

where $[\omega_0]_i = 1$ for $1 \leq i \leq \lfloor \frac{d}{2} \rfloor$ and 0 otherwise; $Z \sim \mathcal{N}(0, 1)$ is Gaussian noise drawn from a Gaussian noise distribution with a mean of 0 and variance of 1. We use a fully connected directed graph network with 10 agents for the simulation. Let the self-loop weight $[A]_{kk} = \frac{1}{2}\ \forall k \in \mathcal{K}$ and the edge weight $[A]_{jk} = \frac{1}{18} \quad \forall j \in \mathcal{N}_k$. The graph network is strongly connected and the adjacency matrix is column stochastic. The adjacency matrix is shown below.

$$A = \begin{bmatrix} \frac{1}{2} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} \\ \frac{1}{18} & \frac{1}{2} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} \\ \frac{1}{18} & \frac{1}{18} & \frac{1}{2} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} \\ \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{2} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} \\ \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{2} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} \\ \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{2} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} \\ \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{2} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} \\ \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{2} & \frac{1}{18} & \frac{1}{18} \\ \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{2} & \frac{1}{18} \\ \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{18} & \frac{1}{2} \\ . & & & & & & & & & \end{bmatrix}$$

The parameters of the simulation are as follows: the step sizes $\alpha_t = \frac{1}{t}$, dimension $d = 3$, and the time duration $T = 100$. This is also shown in Table 3.

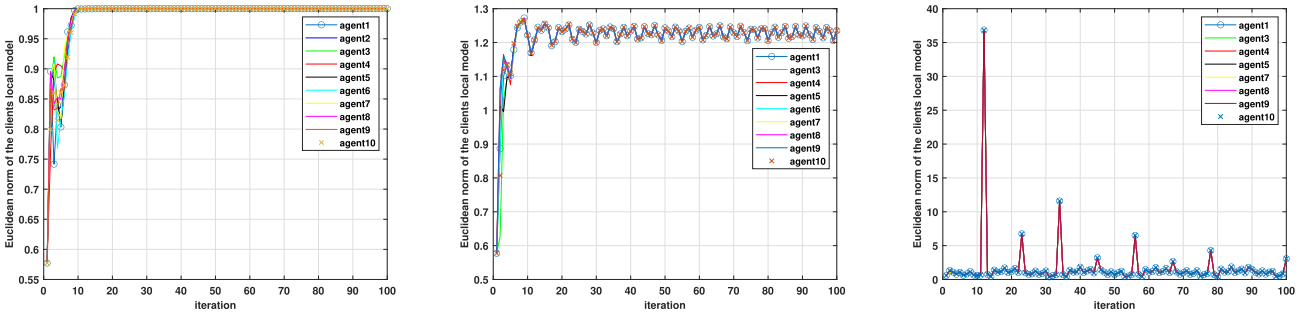
C. SIMULATION RESULTS

First, we will show that the conventional ATC aggregation technique [23], will converge when there is no Byzantine agent in the graph network. To achieve this, we will replace Step 8 in Algorithm 1 with the ATC aggregation technique to enable its application for the online decentralized meta-learning problem. In the presence of Byzantine agents, the ATC online diffusion meta-learning algorithm is expected not to converge. Figure 1(a)-(c) shows the plot of the Euclidean norm of $\omega_{k,t} \quad \forall k \in \mathcal{M}$ with the iteration time t using the ATC online diffusion meta-learning algorithm. It can be seen from Figure 1(a) that when there is no Byzantine agent in the network, the agents converge at iteration time $t = 9$. In Figure 1(b), we let an agent become Byzantine, say agent 2, such that it uses a different update equation to produce its intermediate local model $\theta_{2,t+1}$, i.e., $\theta_{2,t+1} = [5 \sin t, 3 \cos t, 5 \tanh t]^{\mathbb{T}}$, where $\mathbb{T}$ denote transpose operation. It can be seen from Figure 1(b) that there is an oscillation but no convergence. In Figure 1(c), we let two agents become Byzantine, say agents 2 and 6. Agent 6 update its intermediate local model $\theta_{6,t+1}$ using $[\cot t, 5 \sin t, 3 \tan t]^{\mathbb{T}}$. It can be seen from Figure 1(c) that the presence of two Byzantine agents invades the graph network leading to irrational behavior that never converges.

Now, we will compare the performance of the conventional coordinate-wise screening technique [36], [37] when applied in Algorithm 1 with our proposed modified coordinate-wise screening and the centerpoint techniques. It is of note that existing algorithms such as BRIDGE and ByRDiE cannot work directly in online settings. Therefore, for the sake of comparison, we apply the conventional coordinate-wise screening used in BRDIGE and ByRDiE in Step 8 of our proposed algorithm. The conventional coordinate-wise screening, modified coordinate-wise screening, and centerpoint aggregation techniques can all guarantee convergence in the presence of some number of Byzantine clients but with different speeds of convergence. In both the conventional and modified coordinate-wise screening techniques, each normal agent must have at least $2b + 1$ neighbors. From the fully connected graph network, each normal agent has 9 neighbors, which means that the number of Byzantine agents cannot exceed four to provide resiliency and guarantee convergence. For the centerpoint aggregation $b < \lfloor \frac{\lceil \mathcal{N}_k \rceil}{d+1} \rfloor$, which means that the centerpoint aggregation can provide resiliency against at most two Byzantine agents. We will compare the speed of convergence of the conventional coordinate-wise aggregation technique, the modified coordinate-wise aggregation technique, and the centerpoint aggregation technique for only one and two Byzantine agents.

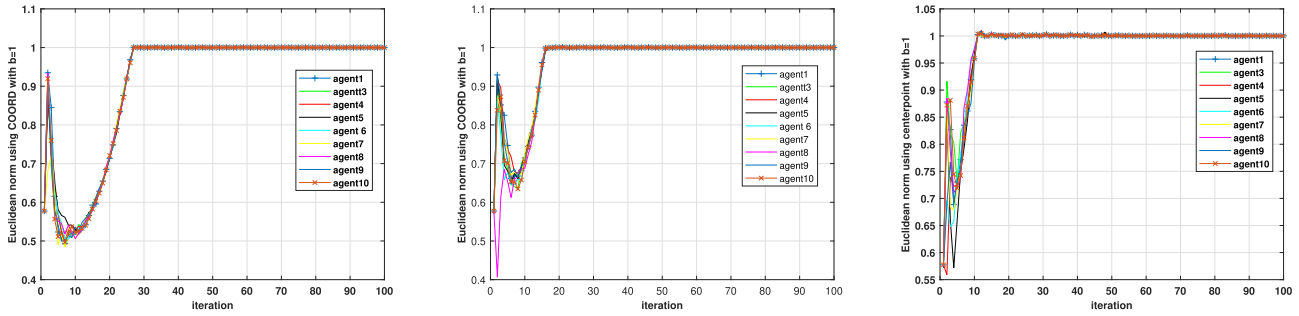
For the case of one Byzantine agent in Figure 2(a)-(c), the centerpoint aggregation technique converges the fastest, while the modified coordinate-wise screening technique converges faster than the conventional coordinate-wise screening technique. The convergence time for the conventional coordinate-wise screening technique in Figure 2(a) is 27, the convergence time for the modified coordinate-wise screening technique in Figure 2(b) is 16, and the convergence time for the centerpoint aggregation technique in Figure 2(c) is 11 in Figure 2(c). It should be noted that a uniform weight is used instead of the self-loop weight in the conventional coordinate-wise aggregation technique, which is responsible for its slow convergence.

We repeat the simulation with two Byzantine agents. Again, the centerpoint aggregation technique converges faster than both the conventional and modified coordinate-wise



(a) no Byzantine agent (b) one Byzantine agent (c) two Byzantine agents

FIGURE 1. Convergence plots using diffusion meta learning Algorithm for 10 agents in the presence of 0, 1, and 2 Byzantine agents with $\alpha_t = \frac{1}{t}$.



(a) coordinate-wise (b) modified coordinate-wise (c) centerpoint

FIGURE 2. Convergence plot using conventional coordinate-wise screening, modified coordinate-wise screening, and centerpoint aggregation for 10 agents in a fully connected graph network which includes 1 Byzantine agent with $\alpha_t = \frac{1}{t}$.

aggregation techniques. This is shown in Figure 3(a)-(c), where the convergence time for centerpoint aggregation is 12 while the convergence time for the conventional and modified coordinate-wise screening techniques are 16 and 28 respectively. As the number of Byzantine agents increased from 1 to 2, it can be seen that it has a negligible effect on the speed of convergence for all three techniques.

We adjust the learning rates of the simulation from $\alpha_t = \frac{1}{t}$ to $\alpha_t = \frac{2}{t}$ to observe its influence on the speed of convergence. This result is shown in Figure 4(a)-(c), where the convergence time for the conventional coordinate-wise screening is 42, the convergence for the modified coordinate-wise screening is 29, and the convergence time for the centerpoint aggregation is 20. It is surprising that increasing the learning rate slows down convergence instead of speeding it up. We deduce that finding the optimal hyperparameters is not straightfoward.

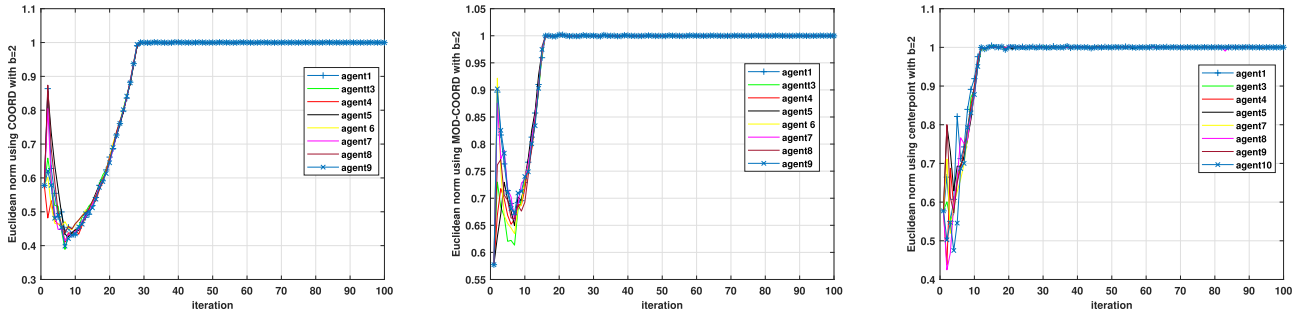
We compare the sublinearity of our regret bound with the online decentralized meta-learning algorithm proposed with regret bound $O(\frac{1}{\sqrt{t}})$ for each agent [19], and the OML algorithm proposed with a regret bound of $\tilde{O}(\ln t)$ [3], where $\tilde{O}(\cdot)$ hides constant parameters. Sublinearity is defined as $\lim_{t\to\infty} \frac{regret}{t} = 0$. This is shown in Figure 4. It can be seen from Figure 4 that our proposed algorithm performs better,

VIII. DISCUSSION

In this work, we developed a Byzantine-resilient online decentralized meta-learning algorithm that can mitigate Byzantine attacks, while coping with the data (or statistical) heterogeneity of the agents, and providing quick adaptation of the trained model to some specific tasks in a time-varying environment. Yet, there are many other challenges in distributed and decentralized learning not covered in this work but require urgent attention. We discuss some of them:

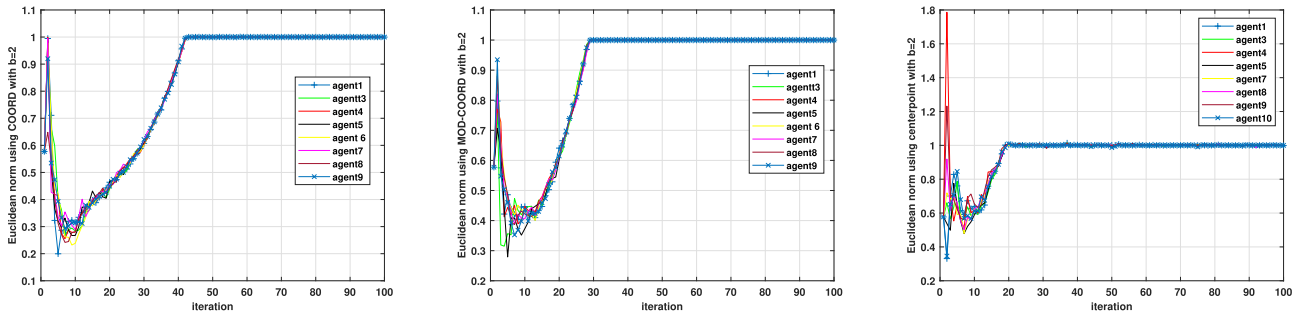
A. THE CHALLENGE OF SYNCHRONIZATION

Synchronization is the de-facto communication mode in distributed and decentralized learning. It has given rise to many synchronization strategies, such as diffusion learning, consensus, adapt-then-combine, etc. However, it may be challenging to achieve synchronous aggregation due to the spatial distribution of agents, and due to system and statistical heterogeneity of the agents. Agents may have different power limitations, bandwidth, and processing speed which are referred to as system heterogeneity, and different data distributions referred to as statistical heterogeneity. This can introduce straggling effects. Therefore, an asynchronous mode of communication can be an effective way to address straggling, although, the use of stale updates defect convergence [68], [69]. Other techniques such as knowledge distillation [70] and quantization [71] can help to conserve



(a) coordinate-wise (b) modified coordinate-wise (c) centerpoint

FIGURE 3. Convergence plot using conventional coordinate-wise screening, modified coordinate-wise screening, and centerpoint aggregation for 10 agents in a fully connected graph network which includes 2 Byzantine agents with $\alpha_t = \frac{1}{t}$.



(a) coordinate-wise (b) modified coordinate-wise (c) centerpoint

FIGURE 4. Convergence plot using conventional coordinate-wise screening, modified coordinate-wise screening, and centerpoint aggregation for 10 agents in a fully connected graph network which includes 2 Byzantine agents with $\alpha_t = \frac{2}{t}$.

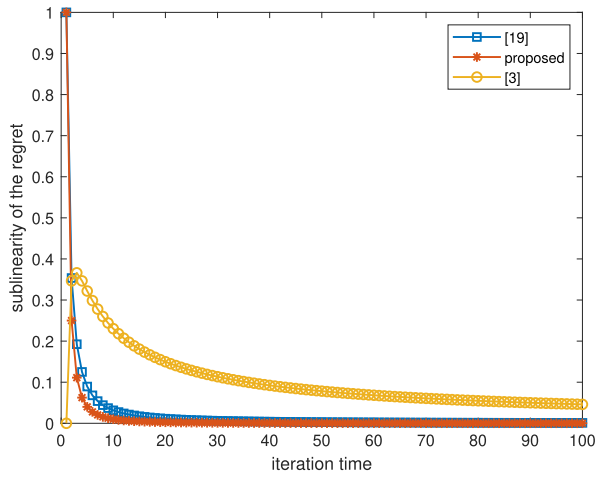


FIGURE 5. Comparison of the sublinearity of the regret bounds.

network resources during a synchronous mode of communication by scaling down the size of the training models.

B. CONFLICTING OBJECTIVES

In many real-world problems, agents may have conflicting objectives, and seeking a common global optimal model or global meta-initializer may be infeasible. Therefore, agents must aim to achieve Pareto-optimality [72]. Algorithms for achieving Pareto-optimality in distributed and decentralized learning are not fully developed.

C. INCOMPLETE INFORMATION

In reality, agents may be faced with bad communication networks that may corrupt the data and limit the information accessible. Also, there may be a delay in transmitting and receiving information. Most distributed and decentralized learning algorithms are not designed to handle such cases. A potential solution would be to develop bandit convex and non-convex optimization algorithms to provide a worst-case regret. We refer interested readers to our published paper that addressed this problem in the non-convex setting [24]. However, obtaining a good regret bound, that matches convex optimization algorithms for the non-convex settings, is still a challenge.

D. PRIVACY

It is possible for some of the normal agents to have a particular interest in the global model or the local models of some other agents. Such normal agents are not malicious but can act as spies. Therefore, it is important to protect the local and global models before and after the aggregation process. Differential privacy is a rigorous mathematical definition of privacy that is well-developed in distributed learning but still underdeveloped in decentralized optimization. However,

we take an early step to address this problem in our previous work [44], [53].

E. DEFENSE AGAINST AN ARBITRARY NUMBER OF BYZANTINE AGENTS

Currently, many Byzantine-resilient algorithms have convergence bounds, beyond which these algorithms fail to provide resiliency. As cyberattacks are increasing and attacks are becoming more sophisticated, there is an urgent need to develop algorithms that are robust against an indefinite number of Byzantine agents in the network.

F. FAIRNESS OF THE GLOBAL OBJECTIVES

Due to statistical and system heterogeneity, the trained global model may be fair to some agents but biased against some other agents. This calls for approaches, such as meta-learning, multi-task learning, constrained optimization, etc., to address this pressing issue [68]. This can be more concerning when the bias is against some protected groups, such as race and gender.

G. COMPUTATIONAL SPEED AND COMPLEXITY

Although meta-learning algorithms are able to quickly adapt to a specific task with a small dataset, training meta-learning algorithms can be computationally intensive. This is because the number of stochastic gradient descent steps is more compared to traditional deep learning algorithms. Therefore, with a series of multiple iterations occurring in training, there will be increased computational time complexity [73]. However, the justification for meta-learning is realized during meta-test time.

IX. CONCLUSION

Meta-learning is a process of distilling task-agnostic knowledge from multiple related tasks over a series of learning episodes to improve learning performance on new tasks from the same task family. Meta-learning has been extended to the online learning setting to provide a continuous lifelong learning experience. However, in a decentralized setting, a single agent is limited to its local task data and must collaborate with other agents in its neighborhood to improve its learning performance. Therefore, online decentralized meta-learning algorithms are designed to allow multiple agents to learn faster and better from shared experiences. On the other hand, online decentralized meta-learning algorithms are susceptible to Byzantine attacks and failure, which lead to poor performance. Although there are some state-of-the-art Byzantine-resilient techniques applied in offline decentralized settings to withstand Byzantine attacks, these techniques are not applicable in online settings and their performance for meta-learning is unknown. Therefore, this paper developed an online decentralized meta-learning algorithm that works with two Byzantine-resilient aggregation techniques to provide better Byzantine resiliency and guarantee higher convergence speed and accuracy. The first proposed Byzantine-resilient aggregation technique is a modification of the coordinate-wise screening used in state-of-the-art Byzantine-resilient algorithms. The second proposed Byzantine-resilient aggregation technique is the centerpoint aggregation based on the centerpoint theorem in discrete geometry. The centerpoint aggregation is both novel and superior to the coordinate-wise screening technique. Simulation results showed that the proposed algorithm is indeed superior to existing algorithms.

REFERENCES

[1] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey, "Meta-learning in neural networks: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 9, pp. 5149–5169, Sep. 2022.
[2] S. Thrun and L. Pratt, "Learning to learn: Introduction and overview," in *Learning to Learn*. New York, NY, USA: Springer, 1998, pp. 3–17.
[3] C. Finn, A. Rajeswaran, S. Kakade, and S. Levine, "Online meta-learning," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 1920–1930.
[4] M. Andrychowicz, M. Denil, S. Gomez, M. W. Hoffman, D. Pfau, T. Schaul, B. Shillingford, and N. De Freitas, "Learning to learn by gradient descent by gradient descent," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016, pp. 1–9.
[5] L. Franceschi, P. Frasconi, S. Salzo, R. Grazzi, and M. Pontil, "Bilevel programming for hyperparameter optimization and meta-learning," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1568–1577.
[6] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11.
[7] L. Metz, N. Maheswaranathan, B. Cheung, and J. Sohl-Dickstein, "Meta-learning update rules for unsupervised representation learning," 2018, *arXiv:1804.00222*.
[8] F. Alet, M. F. Schneider, T. Lozano-Perez, and L. P. Kaelbling, "Meta-learning curiosity algorithms," 2020, *arXiv:2003.05325*.
[9] H. Liu, K. Simonyan, and Y. Yang, "DARTS: Differentiable architecture search," 2018, *arXiv:1806.09055*.
[10] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le, "Regularized evolution for image classifier architecture search," in *Proc. AAAI Conf. Artif. Intell.*, 2019, vol. 33, no. 1, pp. 4780–4789.
[11] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," 2016, *arXiv:1611.01578*.
[12] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1126–1135.
[13] R. Vilalta and Y. Drissi, "A perspective view and survey of meta-learning," *Artif. Intell. Rev.*, vol. 18, no. 2, pp. 77–95, Jun. 2002.
[14] J. Vanschoren, "Meta-learning: A survey," 2018, *arXiv:1810.03548*.
[15] M. Huisman, J. N. van Rijn, and A. Plaat, "A survey of deep meta-learning," *Artif. Intell. Rev.*, vol. 54, no. 6, pp. 4483–4541, Aug. 2021.
[16] T. Yu, X. Geng, C. Finn, and S. Levine, "Variable-shot adaptation for online meta-learning," 2020, *arXiv:2012.07769*.
[17] J. Harrison, A. Sharma, C. Finn, and M. Pavone, "Continuous meta-learning without tasks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 17571–17581.
[18] P. Nazari and E. Khorram, "Dynamic regret analysis for online meta-learning," 2021, *arXiv:2109.14375*.
[19] S. Lin, M. Dedeoglu, and J. Zhang, "Accelerating distributed online meta-learning via multi-agent collaboration under limited communication," in *Proc. 22nd Int. Symp. Theory, Algorithmic Found., Protocol Design Mobile Netw. Mobile Comput.*, Jul. 2021, pp. 261–270.
[20] A. Prodromidis, P. Chan, and S. Stolfo, "Meta-learning in distributed data mining systems: Issues and approaches," *Adv. Distrib. Parallel Knowl. Discovery*, vol. 3, pp. 81–114, Aug. 2000.
[21] Y. Hu, M. Chen, W. Saad, H. V. Poor, and S. Cui, "Distributed multi-agent meta learning for trajectory design in wireless drone networks," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 10, pp. 3177–3192, Oct. 2021.
[22] X. Zhang, C. Hu, B. He, and Z. Han, "Distributed reptile algorithm for meta-learning over multi-agent systems," *IEEE Trans. Signal Process.*, vol. 70, pp. 5443–5456, 2022.

[23] M. Kayaalp, S. Vlaski, and A. H. Sayed, "Dif-MAML: Decentralized multi-agent meta-learning," *IEEE Open J. Signal Process.*, vol. 3, pp. 71–93, 2022.

[24] O. T. Odeyomi, "Online learning of time-varying unbalanced networks in non-convex environments: A multi-armed bandit approach," *IEEE Access*, vol. 11, pp. 15567–15577, 2023.

[25] Z. Yang, A. Gang, and W. U. Bajwa, "Adversary-resilient distributed and decentralized statistical inference and machine learning: An overview of recent advances under the Byzantine threat model," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 146–159, May 2020.

[26] F. Chen, M. Luo, Z. Dong, Z. Li, and X. He, "Federated meta-learning with fast convergence and efficient communication," 2018, *arXiv:1802.07876*.

[27] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 3557–3568.

[28] Y. Jiang, J. Konečnỳ, K. Rush, and S. Kannan, "Improving federated learning personalization via model agnostic meta learning," 2019, *arXiv:1909.12488*.

[29] C. Xie, O. Koyejo, and I. Gupta, "Fall of empires: Breaking Byzantine-tolerant SGD by inner product manipulation," in *Proc. Uncertainty Artif. Intell.*, 2020, pp. 261–270.

[30] M. Kim, L. Lima, F. Zhao, J. Barros, M. Medard, R. Koetter, T. Kalker, and K. J. Han, "On counteracting Byzantine attacks in network coded peer-to-peer networks," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 5, pp. 692–702, Jun. 2010.

[31] Y. Chen, L. Su, and J. Xu, "Distributed statistical machine learning in adversarial settings: Byzantine gradient descent," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 1, no. 2, pp. 1–25, 2017.

[32] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11.

[33] Q. Xia, Z. Tao, Z. Hao, and Q. Li, "FABA: An algorithm for fast aggregation against Byzantine attacks in distributed neural networks," in *Proc. IJCAI*, Aug. 2019, pp. 4824–4830.

[34] A. S. Rawat, P. Anand, H. Chen, and P. K. Varshney, "Countering Byzantine attacks in cognitive radio networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Mar. 2010, pp. 3098–3101.

[35] M. Abdelhakim, L. E. Lightfoot, J. Ren, and T. Li, "Distributed detection in mobile access wireless sensor networks under Byzantine attacks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 4, pp. 950–959, Apr. 2014.

[36] Z. Yang and W. U. Bajwa, "ByRDiE: Byzantine-resilient distributed coordinate descent for decentralized learning," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 5, no. 4, pp. 611–627, Dec. 2019.

[37] C. Fang, Z. Yang, and W. U. Bajwa, "BRIDGE: Byzantine-resilient decentralized gradient descent," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 8, pp. 610–626, 2022.

[38] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 5650–5659.

[39] C. Fang, Z. Yang, and W. U. Bajwa, "BRIDGE: Byzantine-resilient decentralized gradient descent," 2019, *arXiv:1908.08098*.

[40] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Artif. Intell. Statist.*, 2017, pp. 1273–1282.

[41] H. Edelsbrunner, *Algorithms in Combinatorial Geometry*, vol. 10. New York, NY, USA: Springer, 1987.

[42] S. Jadhav and A. Mukhopadhyay, "Computing a centerpoint of a finite planar set of points in linear time," in *Proc. 9th Annu. Symp. Comput. Geometry (SCG)*, 1993, pp. 83–90.

[43] J. Eckhoff, "Helly, radon, and carathéodory type theorems," in *Handbook of Convex Geometry*. Amsterdam, The Netherlands: Elsevier, 1993, pp. 389–448.

[44] O. T. Odeyomi and G. Zaruba, "Byzantine-resilient federated learning with differential privacy using online mirror descent," in *Proc. Int. Conf. Comput., Netw. Commun. (ICNC)*, Feb. 2023, pp. 66–70.

[45] M. B. Cohen, Y. T. Lee, G. Miller, J. Pachocki, and A. Sidford, "Geometric median in nearly linear time," in *Proc. 48th Annu. ACM Symp. Theory Comput.*, Jun. 2016, pp. 9–21.

[46] C. Xie, O. Koyejo, and I. Gupta, "Zeno: Byzantine-suspicious stochastic gradient descent," vol. 24, 2018, *arXiv:1805.10032*.

[47] C. Xie, O. Koyejo, and I. Gupta, "Zeno++: Robust asynchronous SGD with arbitrary number of Byzantine workers," 2019, *arXiv:1903.07020*.

[48] R. Wang, Y. Liu, and Q. Ling, "Byzantine-resilient resource allocation over decentralized networks," *IEEE Trans. Signal Process.*, vol. 70, pp. 4711–4726, 2022.

[49] J. Peng and Q. Ling, "Byzantine-robust decentralized stochastic optimization," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2020, pp. 5935–5939.

[50] S. Guo, T. Zhang, H. Yu, X. Xie, L. Ma, T. Xiang, and Y. Liu, "Byzantine-resilient decentralized stochastic gradient descent," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 6, pp. 4096–4106, Jun. 2022.

[51] A. R. Elkordy, S. Prakash, and S. Avestimehr, "Basil: A fast and Byzantine-resilient approach for decentralized training," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 9, pp. 2694–2716, Sep. 2022.

[52] N. Papernot, P. McDaniel, A. Sinha, and M. P. Wellman, "SoK: Security and privacy in machine learning," in *Proc. IEEE Eur. Symp. Secur. Privacy (EuroS&P)*, Apr. 2018, pp. 399–414.

[53] O. Odeyomi and G. Zaruba, "Differentially-private federated learning with long-term constraints using online mirror descent," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2021, pp. 1308–1313.

[54] W. Abbas, M. Shabbir, J. Li, and X. Koutsoukos, "Resilient distributed vector consensus using centerpoint," *Automatica*, vol. 136, Feb. 2022, Art. no. 110046.

[55] J. Li, W. Abbas, M. Shabbir, and X. Koutsoukos, "Byzantine resilient distributed learning in multirobot systems," *IEEE Trans. Robot.*, vol. 38, no. 6, pp. 3550–3563, Dec. 2022.

[56] S. Bubeck, "Convex optimization: Algorithms and complexity," *Found. Trends Mach. Learn.*, vol. 8, nos. 3–4, pp. 231–357, 2015.

[57] A. Nedić, A. Olshevsky, and M. G. Rabbat, "Network topology and communication-computation tradeoffs in decentralized optimization," *Proc. IEEE*, vol. 106, no. 5, pp. 953–976, May 2018.

[58] S. Shalev-Shwartz, "Online learning and online convex optimization," *Found. Trends Mach. Learn.*, vol. 4, no. 2, pp. 107–194, 2012.

[59] A. Nichol, J. Achiam, and J. Schulman, "On first-order meta-learning algorithms," 2018, *arXiv:1803.02999*.

[60] A. H. Sayed, "Adaptation, learning, and optimization over networks," *Found. Trends Mach. Learn.*, vol. 7, nos. 4–5, pp. 311–801, 2014.

[61] J. Matousek, *Lectures on Discrete Geometry*, vol. 212. New York, NY, USA: Springer, 2013.

[62] G. L. Miller and D. R. Sheehy, "Approximate center points with proofs," in *Proc. 25th Annu. Symp. Comput. Geometry*, Jun. 2009, pp. 153–158.

[63] W. Mulzer and D. Werner, "Approximating tverberg points in linear time for any fixed dimension," in *Proc. 28th Annu. Symp. Comput. Geometry*, Jun. 2012, pp. 303–310.

[64] H. Tverberg, "A generalization of Radon's theorem," *J. London Math. Soc.*, vol. 1, no. 1, pp. 123–128, 1966.

[65] J. Li, W. Abbas, M. Shabbir, and X. D. Koutsoukos, "Resilient distributed diffusion for multi-robot systems using centerpoint," in *Proc. Robot., Sci. Syst.*, 2020, pp. 1–9.

[66] J. Li, W. Abbas, and X. Koutsoukos, "Byzantine resilient distributed multi-task learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 18215–18225.

[67] L. Bottou, F. E. Curtis, and J. Nocedal, "Optimization methods for large-scale machine learning," *SIAM Rev.*, vol. 60, no. 2, pp. 223–311, Jan. 2018.

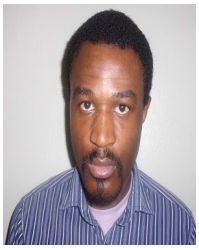
[68] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11.

[69] A. Reisizadeh, H. Taheri, A. Mokhtari, H. Hassani, and R. Pedarsani, "Robust and communication-efficient collaborative learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–12.

[70] C. Wu, F. Wu, L. Lyu, Y. Huang, and X. Xie, "Communication-efficient federated learning via knowledge distillation," *Nature Commun.*, vol. 13, no. 1, p. 2032, Apr. 2022.

[71] A. Reisizadeh, A. Mokhtari, H. Hassani, A. Jadbabaie, and R. Pedarsani, "FedPAQ: A communication-efficient federated learning method with periodic averaging and quantization," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2020, pp. 2021–2031.

[72] J. Tan, R. Khalili, H. Karl, and A. Hecker, "Multi-agent distributed reinforcement learning for making decentralized offloading decisions," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, May 2022, pp. 2098–2107.
[73] A. Yang, C. Lu, J. Li, X. Huang, T. Ji, X. Li, and Y. Sheng, "Application of meta-learning in cyberspace security: A survey," *Digit. Commun. Netw.*, vol. 9, no. 1, pp. 67–78, Feb. 2023.



OLUSOLA T. ODEYOMI (Member, IEEE) received the B.Sc. and M.Sc. degrees in electronic and electrical engineering from Obafemi Awolowo University, Nigeria, in 2011 and 2016, respectively, and the Ph.D. degree in electrical engineering and computer science from Wichita State University, Wichita, KS, USA, in 2021. He was a Postdoctoral Researcher with Howard University and a former Lecturer with Obafemi Awolowo University. He is currently an Assistant Professor of computer science with North Carolina Agricultural and Technical State University, Greensboro, NC, USA. His research interests include machine learning, deep learning, social networks, wireless communication, natural language processing, and bioinformatics. He is a member of the IEEE Information Theory Society and the IEEE Computer Science Society. He won the IEEE International Symposium on Information Theory (ISIT) Video Contest Award, in 2021, and the Outstanding Graduate Student Award from Wichita State University, in 2021. He was also a recipient of the Petroleum Technology Development Fund (PTDF) overseas Ph.D. Scholarship in Nigeria. He is a reviewer of IEEE and ACM journals and conferences.

BASSEY UDE (Student Member, IEEE) received the bachelor's degree in computer science from Coventry University, U.K., in 2015. He is currently teaching and a Research Graduate Student of computer science with North Carolina A&T, USA. He has more than four years of experience as the Product Manager and a Business Analyst. His research interests include cloud computing security, cybersecurity, federated learning, machine learning, and deep learning. He is currently researching on fair resource allocation in federated learning. Looking ahead, he is committed to continuing his research, as well as mentoring the next generation of scholars in his field. He is also passionate about promoting diversity and inclusion in academia and using his work to make a positive impact on society.

KAUSHIK ROY is currently a Professor and the Interim Chair with the Department of Computer Science, North Carolina Agricultural and Technical State University. His research is funded by the National Science Foundation (NSF), the Department of Defense (DoD), and the Department of Energy (DoE). He has more than 150 publications, including 40 journal articles and a book. His current research interests include cybersecurity, cyber identity, biometrics, machine learning (deep learning), data science, cyber-physical systems, and big data analytics. He is the Director of the Center for Cyber Defense (CCD) and directs the Cyber Defense and AI Laboratory.