Coded Robust Aggregation for Distributed Learning under Byzantine Attacks

Chengxi Li, *Member, IEEE,* Ming Xiao, *Senior Member, IEEE,* and Mikael Skoglund, *Fellow, IEEE*

***Abstract*—In this paper, we investigate the problem of distributed learning (DL) in the presence of Byzantine attacks. For this problem, various robust bounded aggregation (RBA) rules have been proposed at the central server to mitigate the impact of Byzantine attacks. However, current DL methods apply RBA rules for the local gradients from the honest devices and the disruptive information from Byzantine devices, and the learning performance degrades significantly when the local gradients of different devices vary considerably from each other. To overcome this limitation, we propose a new DL method to cope with Byzantine attacks based on coded robust aggregation (CRA-DL). Before training begins, the training data are allocated to the devices redundantly. During training, in each iteration, the honest devices transmit coded gradients to the server computed from the allocated training data, and the server then aggregates the information received from both honest and Byzantine devices using RBA rules. In this way, the global gradient can be approximately recovered at the server to update the global model. Compared with current DL methods applying RBA rules, the improvement of CRA-DL is attributed to the fact that the coded gradients sent by the honest devices are closer to each other. This closeness enhances the robustness of the aggregation against Byzantine attacks, since Byzantine messages tend to be significantly different from those of honest devices in this case. We theoretically analyze the convergence performance of CRA-DL. Finally, we present numerical results to verify the superiority of the proposed method over existing baselines, showing its enhanced learning performance under Byzantine attacks.**

***Index Terms*—Byzantine attacks, convergence analysis, distributed learning, gradient coding, robust aggregation.**

I. Introduction

DISTRIBUTED learning (DL) has recently attracted significant attention [1], [2]. In DL, the central server acts as a central processor with access to a very large dataset. To accelerate training on this large dataset, the server distributes the computational workload to multiple devices, which function as worker nodes. This is a common practice in distributed computing for machine learning applications. Compared to training on a single device, DL leverages the computational resources of various edge devices, thereby increasing training efficiency [3], [4]. Typically, the training process of DL involves multiple iterations. In each iteration, the server first sends the global model to the devices. After receiving the global model, each device computes the local gradient based on its local dataset and transmits the local gradient to the server. The server aggregates the local gradients from all devices to obtain the global gradient to update the global model [5].

Due to the distributed nature of the DL system, it is unrealistic for the server to continuously monitor the operational status of the devices to ensure they function properly at all times. Issues such as computation errors, crashes, and stalled processes may arise during training [6]. Besides, some external attackers may compromise the devices before the deployment of the system by injecting malicious firmware [7]. These malfunctioning or compromised devices send incorrect messages during the training, which are known as Byzantine devices [8]–[12]. DL systems affected by them are said to be under Byzantine attacks. To deal with Byzantine attacks, current DL approaches can be classified into two categories as follows.

In the first category, various aggregation rules are designed at the server, which are robust to the Byzantine attacks. For instance, in [13], coordinate-wise median and trimmed mean are adopted to aggregate the information from the devices, and the error rates for strongly convex, non-strongly convex, and smooth non-convex functions are analyzed. In [14], a robust iterative clipping aggregation rule is proposed, where momentum is incorporated to deal with time-coupled Byzantine attacks. In [15], the geometric median is used to aggregate the messages from the devices, resulting in a variant of the typical gradient descent method. In [16], a fast aggregation method is proposed that removes outliers in the messages uploaded by the devices to obtain local gradients closer to the true ones. In [17], a trimmed mean-based approach that is also dimensionally Byzantine-resilient is proposed, which is demonstrated to have nearly linear time complexity. In [6], the messages from the devices are aggregated using majority-based and squared-distance-based methods, where the vectors that minimize the distances to their closest vectors are selected as trustworthy. In [18], a new aggregator at the server is designed based on minimization of Huber loss, which attains enhanced robustness under a certain ratio of Byzantine devices under the independent and identically distributed (i.i.d) assumption. In [19], it is shown that most of the above state-of-the-art robust aggregation rules are all robust bounded aggregation (RBA) rules, where the bias between the aggregation result and the average of the messages from the honest devices is bounded by the the largest deviation of the messages from the honest devices. Although current DL methods with RBA rules at the server can achieve satisfactory Byzantine resilience under certain conditions, they have a significant shortcoming: degradation of learning performance when the local gradients of different

C. Li, M. Xiao and M. Skoglund are with the Division of Information Science and Engineering, School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, 10044 Stockholm, Sweden. (e-mail: chengxli@kth.se; mingx@kth.se; skoglund@kth.se). Corresponding author: Chengxi Li.

devices vary considerably. This degradation occurs because the input to the RBA rules includes both the local gradients from honest devices and the disruptive information from Byzantine devices. When the local gradients of different devices vary significantly due to heterogeneity among subsets in the training data, the disruptive information from the Byzantine devices can more easily mislead and manipulate the output of the aggregator.

In the second category, gradient coding methods are developed, where the training dataset is divided into subsets, and these subsets are redundantly assigned to devices before training. In this way, each device obtains multiple subsets. Leveraging this redundancy, in each training iteration, each device computes local gradients corresponding to its assigned subsets and encodes them into a coded gradient. Honest devices transmit the coded gradients to the server, while Byzantine devices send incorrect messages. Based on the received messages from all devices, the server can fully identify the Byzantine devices, enabling the accurate recovery of the true global gradient as if no Byzantine devices were present. For instance, in [20], coding techniques are combined with group-wise verification to deal with Byzantine attacks, specifically designed for matrix multiplication tasks in DL. In [21], a method specifically designed for matrix-vector multiplication in DL is proposed based on error correction with real numbers under Byzantine attacks, which is proved to be information-theoretically optimal with deterministic guarantees. For more general DL problems, in [22], based on fractional repetition allocation of the training data, erroneous messages from the Byzantine devices are detected and transformed into erasures at the cost of additional local computations of the server and additional communication in each iteration. In [23], the encoding of the local gradients is designed by using the fractional repetition code and cyclic repetition code, and the decoders are proposed based on majority vote and Fourier technique. However, these gradient coding techniques still require a very high level of redundancy in the allocation of training data among the devices to fully recover the true global gradient in each iteration, leading to significant computation and storage burdens on the devices.

In addition to dealing with Byzantine devices in DL, gradient coding techniques have also been explored to address other problems, such as the non-responsive devices in DL commonly referred to as stragglers [24]–[30]. Before training begins, the training data are allocated redundantly across devices. During training, non-straggler devices transmit coded gradients to the server in each iteration based on the local training data, while stragglers do not transmit anything. The server can decode and recover the true global gradient using the received coded gradients from the non-stragglers. Depending on whether the true global gradient is recovered exactly or approximately, current gradient coding techniques for handling stragglers can be classified into exact gradient coding techniques [24]–[26] and approximate gradient coding techniques [27]–[30]. Given that machine learning algorithms are inherently robust to noise, it may not be necessary to fully recover the true global gradient in each iteration. Obtaining an approximate version of the true global gradient may suffice for training machine learning models. As a result, approximate gradient coding techniques have gained significant attention recently, which require only a modest level of redundancy in the allocation of training data and induce lower computational and storage burdens on the devices. Among the existing approximate gradient coding techniques, stochastic gradient coding (SGC), originally proposed in [29], requires very simple encoding and decoding techniques while achieving satisfactory learning performance. In SGC, the training data are allocated to devices in a pair-wise balanced manner, and this method has been applied in various DL scenarios with stragglers [30], [31]. Although SGC was originally proposed to cope with the stragglers in DL, its advantages could also be leveraged to combat Byzantine attacks in DL. Nonetheless, leveraging the strengths of SGC to improve Byzantine resilience remains a significant challenge and is yet to be thoroughly investigated.

To overcome the shortcomings of existing techniques designed for DL under Byzantine attacks, we propose a new DL method based on coded robust aggregation (CRA-DL), by simultaneously exploiting the advantages of SGC and RBA rules. In CRA-DL, before training begins, the training data are divided into subsets and allocated redundantly to the devices in a pair-wise balanced manner, motivated by the SGC scheme. In each training iteration, the server transmits the global model to all devices, and each device computes the local gradients based on its local training data subsets. Subsequently, the local gradients corresponding to different subsets are encoded to generate a single vector on each device, known as the coded gradient. Each honest device transmits its coded gradient to the server, while the Byzantine devices send arbitrarily incorrect messages. The server then receives the vectors from all devices and applies an RBA rule to these messages, which yields a final global update that approximates the global gradient. Finally, the global model is updated at the server with the global update. We analyze the convergence performance of CRA-DL. Additionally, we present ample numerical results to verify that CRA-DL outperforms existing baselines. Our contributions are listed as follows:

1) We propose a new method, i.e., CRA-DL, to deal with the DL problem under Byzantine attacks. In CRA-DL, each device computes local gradients on its data subsets and encodes them into a single coded vector. This redundancy and encoding ensure that the coded vectors from honest devices are more similar, which guarantees that the RBA rules at the server derive a more accurate approximation of the global gradient under Byzantine attacks. This is a meta algorithm that can be employed with any RBA rules proposed in the literature with enhanced learning performance.
2) We analyze the convergence performance of CRA-DL for non-convex loss functions and show that the asymptotic learning error diminishes with greater redundancy in data allocation.
3) Our numerical results verify that CRA-DL significantly improves learning performance and enhances robustness to Byzantine attacks in DL in various scenarios.

The novelty of this work is summarized as follows:

1) Compared to existing methods that directly apply RBA rules at the server to aggregate local gradients [6], [13]–[19], the proposed method aggregates coded vectors at the server using RBA rules to generate the global model update. Since the coded vectors from honest devices are more similar, the robustness of the aggregation at the server is enhanced, leading to a more accurate approximation of the global gradient under Byzantine attacks, thereby improving learning performance.
2) In traditional gradient coding approaches dealing with Byzantine attacks [20]–[23], the primary objective is to fully identify Byzantine devices at the server in order to accurately recover the true global gradient as if no Byzantine devices were present. In contrast, the key idea in our method is to apply RBA rules at the server to aggregate coded gradients, allowing for an approximate recovery of the global gradient, which is then used to update the global model. Compared to current gradient coding methods, the proposed method attains Byzantine robustness with a significantly lower level of redundancy in the allocation of training data among devices. In other words, the proposed method imposes lower computational and storage burdens on the devices.

The structure of this paper is as follows. In Section II, we introduce the problem model. In Section III, we propose our method and describe the implementation procedure. In Section IV, we present the performance analysis from a theoretical perspective. The numerical results are shown in Section V to demonstrate the superior performance of the proposed method. Finally, concluding remarks are provided in Section VI.

II. Problem Model

The considered problem is introduced as follows. There are N devices and a central server in the DL system, whose goal is to train a model by solving the optimization problem [1], [23], [32]:

$$\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathbb{R}^D} F(\mathbf{x}), \tag{1}$$

where $\mathbf{x}$ represents the model parameter vector, and $F(\mathbf{x})$ is the overall training loss defined as

$$F(\mathbf{x}) = \sum_{\varrho\in\mathcal{D}} l(\mathbf{x}, \varrho), \tag{2}$$

where $l(\mathbf{x}, \varrho) : \mathbb{R}^D \to \mathbb{R}$ represents the training loss based on the training data sample ϱ in the training dataset $\mathcal{D}$. Without specification, all vectors in this paper are column vectors.

Under the typical DL framework [33], before the training starts, the training dataset $\mathcal{D}$ is divided into N non-overlapping subsets and allocated to N devices so that each device obtains one subset. During the training, in iteration t, the current global model $\mathbf{x}^t$ is transmitted from the server to the devices. Then, each device computes the local gradient corresponding to its subset and sends the local gradient to the server. After receiving the local gradients from the devices, the server aggregates them to form the global gradient to update the global model and to obtain $\mathbf{x}^{t+1}$ [23].

In the above system, some devices may be under Byzantine attacks due to malicious attacks or malfunction of the devices [8]–[12]. Let us define $\mathcal{B}^t$ as the set containing all indices of the Byzantine devices in iteration t, and use $\mathcal{H}^t$ as the set containing all indices of the honest devices in iteration t. During each iteration, the honest devices transmit truth-worthy messages to the server as expected, while the Byzantine devices transmit arbitrarily incorrect information to the server. Without prior knowledge of the Byzantine devices, it is assumed that in each iteration, a certain fraction α of the devices are Byzantine, while the rest are honest [19]. The server does not know the identities of the devices beforehand and only knows the value of α [19]. The identities of the devices are considered to be random and independent across iterations [23], which implies that the identities of the devices are non-persistent across iterations. If the identities were persistent, the server could potentially identify the Byzantine devices by accumulating information over time. From this perspective, the non-persistent case represents the strongest form of Byzantine attacks.

For the above problem, our aim is to enhance the learning performance under Byzantine attacks.

III. The Proposed Method: CRA-DL

In this section, we describe the implementation details of the proposed CRA-DL method.

Before the training starts, the training dataset $\mathcal{D}$ is divided into M subsets, represented as $\mathcal{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_M\}$. These subsets are allocated to N devices in a pair-wise balanced manner, motivated by the advantages of the SGC scheme [29]. Specifically, each device i holds r subsets from the total set of subsets[1]. The number of subsets held by both device i and device j is $\frac{r^2}{M}$, for $i \neq j$. Let us denote the number of devices that hold subset $\mathcal{D}_k$ as d_k, $\forall k$. We define a data allocation matrix $\mathbf{S}$, where $s(i,k)$ is the (i,k)-th element. If $s(i,k) = 1$, subset $\mathcal{D}_k$ is allocated to device i; otherwise, it is not. Based on the above setting, the problem in (1) can be equivalently expressed as:

$$\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathbb{R}^D} F(\mathbf{x}) \triangleq \arg\min_{\mathbf{x}\in\mathbb{R}^D} \sum_{k=1}^{M} f_k(\mathbf{x}), \tag{3}$$

where $f_k(\mathbf{x}) : \mathbb{R}^D \to \mathbb{R}$ denotes the training loss associated with subset $\mathcal{D}_k$:

$$f_k(\mathbf{x}) = \sum_{\varrho\in\mathcal{D}_k} l(\mathbf{x}, \varrho). \tag{4}$$

Next, during the training process, in iteration t, the server sends the current global model $\mathbf{x}^t$ to all devices. After that, each device i computes the local gradients associated with its local subsets and obtains $\{\nabla f_k(\mathbf{x}^t) \,|k \in \{1, ..., M\}, s(i,k) \neq 0\}$. Based on that, device i encodes the local gradients into a single vector as

$$\mathbf{g}_i^t = \sum_{k\in\{\,k|s(i,k)\neq 0\}} \frac{1}{d_k} \nabla f_k(\mathbf{x}^t). \tag{5}$$

[1]We focus on the case where the same number of subsets is allocated to each device. It is worth noting that the analysis in this paper can be easily extended to the case where different numbers of subsets are assigned to each device.

Table I
THE VALUES OF C_α^2 OF SOME COMMONLY USED RBA RULES [19]

RBA Rules	C_α^2
Coordinate-wise median [13]	$\frac{1}{2(1-\alpha)^2}\left[\min\left\{2\sqrt{N-N\alpha}, \sqrt{D}\right\}\right]^2$
Trimmed mean [13]	$\frac{2\alpha(1-\alpha)}{(1-2\alpha)^2}$
Geometric median [15]	$\left[\frac{2(1-\alpha)}{1-2\alpha}\right]^2$
Krum [6]	$2\left(1+\sqrt{\frac{1-\alpha}{1-2\alpha}}\right)^2$
Phocas [17]	$4+\frac{12\alpha(1-\alpha)}{(1-2\alpha)^2}$
FABA [16]	$4\left(\frac{N\alpha}{N-N\alpha}+\frac{N+1-N\alpha}{N-N\alpha}\frac{N\alpha}{N-3N\alpha}\right)$

If device i is honest, i.e., $i \in \mathcal{H}^t$, the coded gradient $\mathbf{g}_i^t$ is sent to the server, $\forall i$. If device j is a Byzantine device, i.e., $j \in \mathcal{B}^t$, it transmits incorrect information to the server, denoted by $\mathbf{b}_j^t$, which is the same size as $\mathbf{g}_j^t$ but contains different elements. With the received messages from the devices, denoted by $\{\{\mathbf{g}_i^t\}_{i\in\mathcal{H}^t}, \{\mathbf{b}_j^t\}_{j\in\mathcal{B}^t}\}$, the server adopts an RBA rule $A(\cdot)$ to yield the global model update as

$$\hat{\mathbf{g}}^t = A\left(\{\mathbf{g}_i^t\}_{i\in\mathcal{H}^t}, \{\mathbf{b}_j^t\}_{j\in\mathcal{B}^t}\right), \tag{6}$$

which is an approximate version of the global gradient. Here, RBA rules are utilized, considering that various state-of-the-art robust aggregation rules with recent advances fall within their scope [19]. RBA rules have been well-defined in [19], and the definition is provided below.

Definition 1 (RBA rules [19])**.** *Suppose there are N_1 messages $\mathbf{z}_1, ..., \mathbf{z}_{N_1} \in \mathbb{R}^D$ from N_1 honest devices and N_2 messages $\tilde{\mathbf{z}}_1, ..., \tilde{\mathbf{z}}_{N_2} \in \mathbb{R}^D$ from N_2 Byzantine devices. The fraction of Byzantine devices is $\alpha = \frac{N_2}{N_1+N_2}$. An aggregation rule $A(\cdot)$ is an RBA rule, if the difference between the aggregation result and the average of the messages from the honest devices is bounded by*

$$\left\| A\left(\{\mathbf{z}_i\}_{i\in\{1,...,N_1\}}, \{\tilde{\mathbf{z}}_j\}_{j\in\{1,...,N_2\}}\right) - \bar{\mathbf{z}}\right\|^2 \leqslant C_\alpha^2 \varsigma, \tag{7}$$

where $\bar{\mathbf{z}}$ is the average of the messages from the honest devices denoted by $\bar{\mathbf{z}} = \frac{1}{N_1}\sum_{i=1}^{N_1}\mathbf{z}_i$, ς is defined as $\varsigma = \max_{i\in\{1,...,N_1\}}\|\bar{\mathbf{z}} - \mathbf{z}_i\|^2$, and C_α^2 is a constant determined by the value of α. The values of C_α^2 for some commonly used RBA rules are provided in Table I [19], where $N = N_1 + N_2$. From (7), it can be seen that a more accurate aggregation result can be achieved when the messages sent by the honest devices are closer to each other.

Note that the proposed method is a meta-algorithm that can be adapted based on the choice of any particular RBA rule. Consequently, various state-of-the-art RBA rules, including those introduced in [6], [13]–[17], [19], can be applied within our proposed method.

At the end of iteration t, the global model is updated at the server as

$$\mathbf{x}^{t+1} = \mathbf{x}^t - \gamma^t \hat{\mathbf{g}}^t, \tag{8}$$

where γ^t is the learning rate. The paradigm of the proposed method is shown as Fig. 1, which is also presented as Algorithm 1.

Algorithm 1: CRA-DL

Input: Training data $\mathcal{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_M\}$, learning rate $\{\gamma^t\}$.
Output: Trained model $\mathbf{x}^{T+1}$.
Initialization: Initialize the model $\mathbf{x}^0$.
for $t = 0$ **to** T **do**
  Server sends the global model $\mathbf{x}^t$ to all devices.
  for *each device i in parallel* **do**
    Compute local gradients: $\{\nabla f_k(\mathbf{x}^t) | s(i,k) = 1\}$.
    Encode local gradients as (5).
    if $i \in \mathcal{H}^t$ *(honest)* **then**
      Transmit $\mathbf{g}_i^t$ to the server.
    else
      Transmit an incorrect vector $\mathbf{b}_i^t$ to the server.
    end
  end
  Server receives $\{\mathbf{g}_i^t\}_{i\in\mathcal{H}^t}$ and $\{\mathbf{b}_j^t\}_{j\in\mathcal{B}^t}$.
  Aggregate the messages using an RBA rule as (6).
  Update the model as (8).
end
return $\mathbf{x}^{T+1}$

From a high-level perspective, in the proposed method, the server aggregates the coded gradients and incorrect messages from the Byzantine devices using RBA rules, rather than directly aggregating the local gradients with the incorrect messages from the Byzantine devices. In this way, by leveraging the redundancy in data allocation, coded gradients are closer to each other compared to the original local gradients. By increasing the redundancy in data allocation, the coded gradients become increasingly similar. This will be analytically demonstrated in Section IV. According to the properties of RBA rules implied by Definition 1, the difference between the aggregation output and the average of the messages from the honest devices is reduced by increasing the redundancy in data allocation. In this way, a more accurate global gradient can be recovered at the server under Byzantine attacks, which is then used to update the global model. This process improves learning performance and enhances the robustness against Byzantine attacks.

IV. PERFORMANCE ANALYSIS

In this section, we analyze the convergence performance of CRA-DL. First, let us state the assumptions, which have been widely used in the related field.

Assumption 1. *The overall training loss F is L-smooth, which indicates the following inequality [34], [35]:*

$$F(\mathbf{x}) \leq F(\mathbf{y}) + \langle \nabla F(\mathbf{y}), \mathbf{x} - \mathbf{y}\rangle + \frac{L}{2}\|\mathbf{x} - \mathbf{y}\|^2, \forall \mathbf{x}, \mathbf{y}. \tag{9}$$

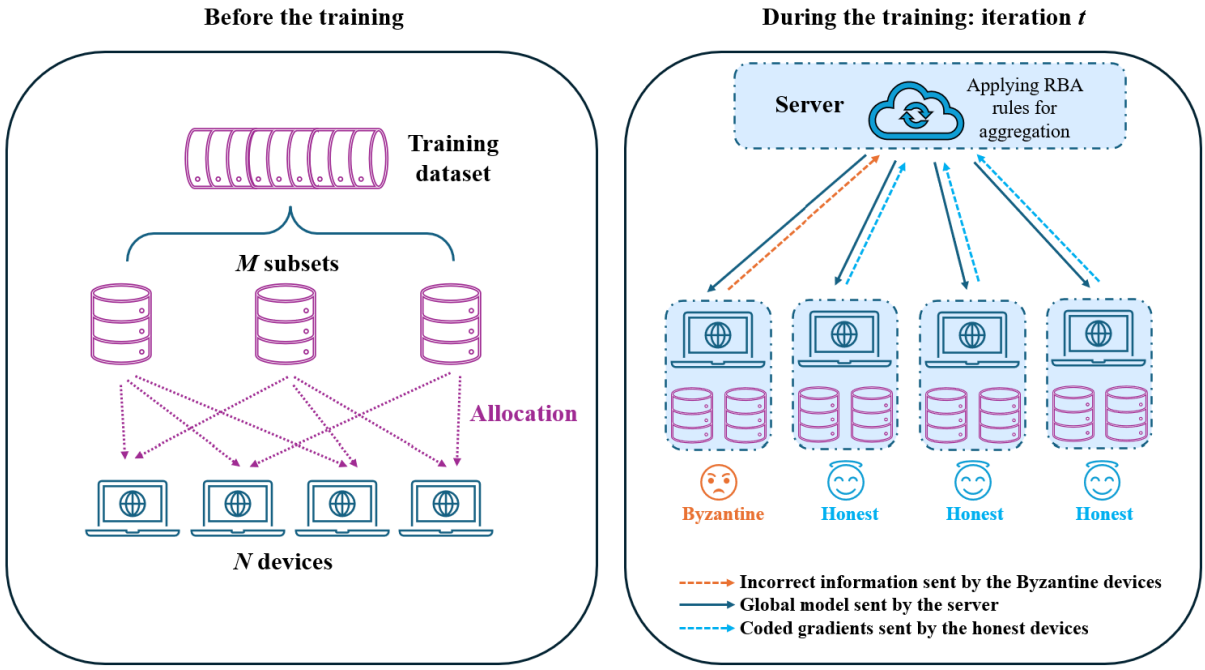


Figure 1. The paradigm of the proposed method.

Assumption 2. *The heterogeneity among the subsets* $\{\mathcal{D}_1, ..., \mathcal{D}_M\}$ *is bounded, indicating [36]*

$$\left\|\nabla f_i(\mathbf{x}) - \frac{1}{M}\nabla F(\mathbf{x})\right\|^2 \leqslant \beta^2, \forall i, \forall \mathbf{x}. \tag{10}$$

Assumption 3. *For some constant* F^**, it holds that [37]*

$$F(\mathbf{x}) \geqslant F^*, \forall \mathbf{x}, \tag{11}$$

which implies the overall training loss is lower bounded by F^*.

Let us present two lemmas which aid the derivation of the main theorem.

Lemma 1. *The maximum difference between any two coded gradients can be bounded as*

$$\begin{aligned}&\max_{i,j\in\{1,...,N\}}\left\|\mathbf{g}_i^t - \mathbf{g}_j^t\right\|^2\\ \leqslant& 8\frac{1}{d_{\min}^2}\left(r - \frac{r^2}{M}\right)^2\left(\beta^2 + \frac{1}{M^2}\left\|\nabla F(\mathbf{x}^t)\right\|^2\right),\end{aligned} \tag{12}$$

where $d_{\min} \triangleq \min\{d_1, ..., d_M\}$.

Proof. Please see Appendix A. □

Remark 1. *When the values of* $d_1, \ldots, d_M$ *do not vary significantly from each other, it holds that* $d_{\min}M \approx Nr$. *Under this condition, we can rewrite (12) as*

$$\begin{aligned}&\max_{i,j\in\{1,...,N\}}\left\|\mathbf{g}_i^t - \mathbf{g}_j^t\right\|^2\\ \leqslant& 8\frac{(M-r)^2}{N^2}\left(\beta^2 + \frac{1}{M^2}\left\|\nabla F(\mathbf{x}^t)\right\|^2\right).\end{aligned} \tag{13}$$

From (13), it can be observed that as the value of r increases, meaning greater redundancy in data allocation, the maximum difference between any two coded gradients is bounded by a smaller value, which implies that the coded gradients are closer to each other. As a special case, when $r = M$, the maximum difference between any two coded gradients becomes zero. In this case, all coded gradients sent by the honest devices are identical. In contrast, in existing methods based on RBA rules, there is no data allocation redundancy, and the maximum difference between any two local gradients sent by honest devices is determined by β, as defined in Assumption 2. Without loss of generality, in the case where $M = N$, device i transmits $\nabla f_i(\mathbf{x}^t)$ to the server, and device j transmits $\nabla f_j(\mathbf{x}^t)$ to the server, for all i, j. In this case, the maximum difference between any two local gradients sent by the honest devices can be bounded as follows:

$$\begin{aligned}&\max_{i,j}\left\|\nabla f_i(\mathbf{x}^t) - \nabla f_j(\mathbf{x}^t)\right\|^2\\ \leqslant&\max_{i,j}\left\|\nabla f_i(\mathbf{x}^t) - \frac{1}{M}\nabla F(\mathbf{x}^t) + \frac{1}{M}\nabla F(\mathbf{x}^t) - \nabla f_j(\mathbf{x}^t)\right\|^2\\ \leqslant&2\max_i\left\|\nabla f_i(\mathbf{x}^t) - \frac{1}{M}\nabla F(\mathbf{x}^t)\right\|^2\\ &+ 2\max_j\left\|f_j(\mathbf{x}^t) - \frac{1}{M}\nabla F(\mathbf{x}^t)\right\|^2 \leqslant 4\beta^2.\end{aligned} \tag{14}$$

By comparing (13) and (14), it can be seen that the maximum difference among the messages sent by the honest devices can be reduced in the proposed method by increasing the level of redundancy in the training data allocation, potentially approaching zero. In contrast, this reduction is not possible in existing DL methods with RBA rules. Note from (7) that a more accurate version of the global gradient can be obtained at the server when the messages sent by honest devices are closer to each other, when applying RBA rules in both the

proposed method and the existing DL methods. Based on that, the proposed method is more likely to achieve better learning performance by adopting a global model update that more accurately approximates the true global gradient, while the learning performance of existing DL methods with RBA rules deteriorates as the variation among local gradients increases.

Lemma 2. *Let us denote the average of the messages from the honest devices in iteration t as*

$$\bar{\mathbf{g}}^t = \frac{1}{|\mathcal{H}^t|}\sum_{i\in\mathcal{H}^t}\mathbf{g}_i^t. \tag{15}$$

We can bound $\bar{\mathbf{g}}^t$ conditioned on the previous iterations as

$$\begin{aligned}&\mathbb{E}\left(\left\|\bar{\mathbf{g}}^t\right\|^2\middle|\mathcal{F}^t\right)\\ \leqslant&\frac{(\phi_1-\phi_2)\,2r^2}{(1-\alpha)^2Nd_{\min}^2}\left[\beta^2+\frac{1}{M^2}\left\|\nabla F\left(\mathbf{x}^t\right)\right\|^2\right]\\ &+\frac{\phi_2}{(1-\alpha)^2N^2}\left\|\nabla F\left(\mathbf{x}^t\right)\right\|^2,\end{aligned} \tag{16}$$

where $\mathbb{E}\left(\cdot\middle|\mathcal{F}^t\right)$ denotes the expectation conditioned on the previous iterations $0,...,t-1$, and ϕ_1 and ϕ_2 are defined as the following constants:

$$\phi_1 \triangleq 1-\alpha, \phi_2 \triangleq \frac{(1-\alpha)\left(N-N\alpha-1\right)}{N-1}. \tag{17}$$

Proof. Please see Appendix B. □

Next, based on Lemma 1 and Lemma 2, we characterize the convergence performance of the proposed method in the following theorem.

Theorem 1 (Convergence performance of CRA-DL with fixed learning rates)**.** *Based on Assumptions 1-3, if $C_\alpha < \frac{d_{\min}M}{2\sqrt{2}N\left(r-\frac{r^2}{M}\right)}$, with fixed learning rates $\gamma^t=\gamma=\frac{\lambda}{\sqrt{T+1}}$, $\lambda>0$, for $T>\left(\frac{\lambda\rho_2}{\rho_1}\right)^2-1$, CRA-DL converges as*

$$\begin{aligned}&\frac{1}{T+1}\sum_{t=0}^{T}\mathbb{E}\left[\left\|\nabla F\left(\mathbf{x}^t\right)\right\|^2\right]\\ \leqslant&\frac{F\left(\mathbf{x}^0\right)-F^*}{\lambda\sqrt{T+1}\rho_1-\lambda^2\rho_2}+\frac{\sqrt{T+1}\rho_3+\lambda\rho_4}{\sqrt{T+1}\rho_1-\lambda\rho_2},\end{aligned} \tag{18}$$

where

$$\rho_1 \triangleq \frac{1}{N}-2\left(r-\frac{r^2}{M}\right)\frac{\sqrt{2C_\alpha^2}}{d_{\min}M}, \tag{19}$$

$$\begin{aligned}\rho_2 \triangleq&\frac{(\phi_1-\phi_2)\,2r^2L}{(1-\alpha)^2Nd_{\min}^2M^2}+\frac{\phi_2L}{(1-\alpha)^2N^2}\\ &+8\frac{LC_\alpha^2}{d_{\min}^2M^2}\left(r-\frac{r^2}{M}\right)^2,\end{aligned} \tag{20}$$

$$\rho_3 \triangleq \frac{\beta^2M\sqrt{2C_\alpha^2}}{d_{\min}}\left(r-\frac{r^2}{M}\right), \tag{21}$$

$$\rho_4 \triangleq \frac{(\phi_1-\phi_2)\,2r^2\beta^2L}{(1-\alpha)^2Nd_{\min}^2}+\frac{8C_\alpha^2\beta^2L}{d_{\min}^2}\left(r-\frac{r^2}{M}\right)^2. \tag{22}$$

Proof. From Assumption 1, we can derive

$$\begin{aligned}&\mathbb{E}\left[F\left(\mathbf{x}^{t+1}\right)\middle|\mathcal{F}^t\right]\\ \leqslant&F\left(\mathbf{x}^t\right)+\mathbb{E}\left[\left\langle\nabla F\left(\mathbf{x}^t\right),\mathbf{x}^{t+1}-\mathbf{x}^t\right\rangle\middle|\mathcal{F}^t\right]\\ &+\frac{L}{2}\mathbb{E}\left(\left\|\mathbf{x}^{t+1}-\mathbf{x}^t\right\|^2\middle|\mathcal{F}^t\right)\\ \overset{\langle1\rangle}{=}&F\left(\mathbf{x}^t\right)-\gamma^t\mathbb{E}\left[\left\langle\nabla F\left(\mathbf{x}^t\right),\hat{\mathbf{g}}^t\right\rangle\middle|\mathcal{F}^t\right]+\frac{L}{2}\mathbb{E}\left(\left\|\gamma^t\hat{\mathbf{g}}^t\right\|^2\middle|\mathcal{F}^t\right)\\ =&F\left(\mathbf{x}^t\right)-\gamma^t\mathbb{E}\left[\left\langle\nabla F\left(\mathbf{x}^t\right),\hat{\mathbf{g}}^t-\bar{\mathbf{g}}^t+\bar{\mathbf{g}}^t\right\rangle\middle|\mathcal{F}^t\right]\\ &+\frac{L(\gamma^t)^2}{2}\mathbb{E}\left(\left\|\hat{\mathbf{g}}^t-\bar{\mathbf{g}}^t+\bar{\mathbf{g}}^t\right\|^2\middle|\mathcal{F}^t\right)\\ \overset{\langle2\rangle}{\leqslant}&F\left(\mathbf{x}^t\right)-\gamma^t\mathbb{E}\left[\left\langle\nabla F\left(\mathbf{x}^t\right),\hat{\mathbf{g}}^t-\bar{\mathbf{g}}^t\right\rangle\middle|\mathcal{F}^t\right]\\ &-\gamma^t\mathbb{E}\left[\left\langle\nabla F\left(\mathbf{x}^t\right),\bar{\mathbf{g}}^t\right\rangle\middle|\mathcal{F}^t\right]\\ &+L(\gamma^t)^2\mathbb{E}\left(\left\|\hat{\mathbf{g}}^t-\bar{\mathbf{g}}^t\right\|^2\middle|\mathcal{F}^t\right)+L(\gamma^t)^2\mathbb{E}\left(\left\|\bar{\mathbf{g}}^t\right\|^2\middle|\mathcal{F}^t\right)\\ \overset{\langle3\rangle}{\leqslant}&F\left(\mathbf{x}^t\right)+\gamma^t\frac{\eta\left\|\nabla F\left(\mathbf{x}^t\right)\right\|^2+\frac{1}{\eta}\mathbb{E}\left[\left\|\hat{\mathbf{g}}^t-\bar{\mathbf{g}}^t\right\|^2\middle|\mathcal{F}^t\right]}{2}\\ &-\gamma^t\frac{1}{N}\left\|\nabla F\left(\mathbf{x}^t\right)\right\|^2\\ &+L(\gamma^t)^2\mathbb{E}\left(\left\|\hat{\mathbf{g}}^t-\bar{\mathbf{g}}^t\right\|^2\middle|\mathcal{F}^t\right)+L(\gamma^t)^2\mathbb{E}\left(\left\|\bar{\mathbf{g}}^t\right\|^2\middle|\mathcal{F}^t\right),\end{aligned} \tag{23}$$

$\forall\eta>0$, where $\langle1\rangle$ is obtained by substituting (8) into (23), $\langle2\rangle$ is derived from the basic inequality given as (43), and $\langle3\rangle$ holds due to Young's Inequality and the following relationship:

$$\mathbb{E}\left[\bar{\mathbf{g}}^t\middle|\mathcal{F}^t\right]=\frac{1}{N}\nabla F\left(\mathbf{x}^t\right). \tag{24}$$

In (23), we can bound $\mathbb{E}\left[\left\|\hat{\mathbf{g}}^t-\bar{\mathbf{g}}^t\right\|^2\middle|\mathcal{F}^t\right]$ as

$$\begin{aligned}&\mathbb{E}\left[\left\|\hat{\mathbf{g}}^t-\bar{\mathbf{g}}^t\right\|^2\middle|\mathcal{F}^t\right]\leqslant C_\alpha^2\mathbb{E}\left[\varsigma^t\middle|\mathcal{F}^t\right]\\ \leqslant&C_\alpha^2\max_{i,j\in\{1,...,N\}}\left\|\mathbf{g}_i^t-\mathbf{g}_j^t\right\|^2,\end{aligned} \tag{25}$$

where $\varsigma^t \triangleq \max_{i\in\mathcal{H}^t}\left\|\bar{\mathbf{g}}^t-\mathbf{g}_i^t\right\|^2$, the first inequality is obtained from (6), (15) and Definition 1, and the second inequality is due to the inequality $\max_{i\in\mathcal{H}^t}\left\|\bar{\mathbf{g}}^t-\mathbf{g}_i^t\right\|^2 \leqslant \max_{i,j\in\mathcal{H}^t}\left\|\mathbf{g}_i^t-\mathbf{g}_j^t\right\|^2 \leqslant \max_{i,j\in\{1,...,N\}}\left\|\mathbf{g}_i^t-\mathbf{g}_j^t\right\|^2$. Substituting Lemma 1 into (25) yields

$$\begin{aligned}&\mathbb{E}\left[\left\|\hat{\mathbf{g}}^t-\bar{\mathbf{g}}^t\right\|^2\middle|\mathcal{F}^t\right]\\ \leqslant&\frac{8C_\alpha^2}{d_{\min}^2}\left(r-\frac{r^2}{M}\right)^2\left[\beta^2+\frac{1}{M^2}\left\|\nabla F\left(\mathbf{x}^t\right)\right\|^2\right].\end{aligned} \tag{26}$$

After that, substituting (26) and (16) in Lemma 2 into (23), we have

$$\begin{aligned}&\gamma^t\left[\frac{1}{N}-\frac{\eta+\frac{8C_\alpha^2}{\eta M^2d_{\min}^2}\left(r-\frac{r^2}{M}\right)^2}{2}\right]\left\|\nabla F\left(\mathbf{x}^t\right)\right\|^2\\ &-L(\gamma^t)^2\left\|\nabla F\left(\mathbf{x}^t\right)\right\|^2\left\{\frac{(\phi_1-\phi_2)\,2r^2}{(1-\alpha)^2Nd_{\min}^2M^2}+\frac{\phi_2}{(1-\alpha)^2N^2}\right.\\ &\left.+C_\alpha^28\frac{1}{d_{\min}^2M^2}\left(r-\frac{r^2}{M}\right)^2\right\}\\ \leqslant&F\left(\mathbf{x}^t\right)-\mathbb{E}\left[F\left(\mathbf{x}^{t+1}\right)\middle|\mathcal{F}^t\right]+\gamma^t\frac{1}{\eta}\frac{4\beta^2C_\alpha^2}{d_{\min}^2}\left(r-\frac{r^2}{M}\right)^2\end{aligned}$$

$$+L\left(\gamma^t\right)^2\left\{\frac{\left(\phi_1-\phi_2\right)2r^2\beta^2}{\left(1-\alpha\right)^2Nd_{\min}^2}+\frac{8C_\alpha^2\beta^2}{d_{\min}^2}\left(r-\frac{r^2}{M}\right)^2\right\}. \tag{27}$$

In (27), by setting $\eta=2\left(r-\frac{r^2}{M}\right)\frac{\sqrt{2C_\alpha^2}}{d_{\min}M}$, it holds that $\frac{1}{N}-\frac{\eta+\frac{8C_\alpha^2}{\eta M^2d_{\min}^2}\left(r-\frac{r^2}{M}\right)^2}{2}>0$ under the condition $C_\alpha<\frac{d_{\min}M}{2\sqrt{2}N\left(r-\frac{r^2}{M}\right)}$. Based on that, we can rewrite (27) as

$$\begin{aligned}&\gamma^t\rho_1\left\|\nabla F\left(\mathbf{x}^t\right)\right\|^2-\left(\gamma^t\right)^2\rho_2\|\nabla F\left(\mathbf{x}\right)\|^2\\ \leqslant&F\left(\mathbf{x}^t\right)-\mathbb{E}\left[F\left(\mathbf{x}^{t+1}\right)\middle|\mathcal{F}^t\right]+\gamma^t\rho_3+\left(\gamma^t\right)^2\rho_4,\end{aligned} \tag{28}$$

where ρ_1,ρ_2,ρ_3 and ρ_4 are all positive constants defined as (19)-(22).

Taking full expectation on both sides of (28) yields

$$\begin{aligned}&\gamma^t\rho_1\mathbb{E}\left(\left\|\nabla F\left(\mathbf{x}^t\right)\right\|^2\right)-\left(\gamma^t\right)^2\rho_2\mathbb{E}\left(\|\nabla F\left(\mathbf{x}\right)\|^2\right)\\ \leqslant&\mathbb{E}\left[F\left(\mathbf{x}^t\right)\right]-\mathbb{E}\left[F\left(\mathbf{x}^{t+1}\right)\right]+\gamma^t\rho_3+\left(\gamma^t\right)^2\rho_4.\end{aligned} \tag{29}$$

Rearranging the terms in (29) and taking average over T iterations, we can obtain

$$\begin{aligned}&\frac{1}{T+1}\sum_{t=0}^{T}\left(\gamma^t\rho_1-\left(\gamma^t\right)^2\rho_2\right)\mathbb{E}\left[\left\|\nabla F\left(\mathbf{x}^t\right)\right\|^2\right]\\ \leqslant&\frac{F\left(\mathbf{x}^0\right)-\mathbb{E}\left[F\left(\mathbf{x}^{T+1}\right)\right]}{T+1}+\frac{1}{T+1}\sum_{t=0}^{T}\left[\gamma^t\rho_3+\left(\gamma^t\right)^2\rho_4\right]\\ \leqslant&\frac{F\left(\mathbf{x}^0\right)-F^*}{T+1}+\frac{1}{T+1}\sum_{t=0}^{T}\left[\gamma^t\rho_3+\left(\gamma^t\right)^2\rho_4\right],\end{aligned} \tag{30}$$

where Assumption 3 is applied to derive the last inequality. With fixed learning rates $\gamma^t=\gamma=\frac{\lambda}{\sqrt{T+1}}$, for $T>\left(\frac{\lambda\rho_2}{\rho_1}\right)^2-1$, we can rewrite (30) as

$$\begin{aligned}&\frac{1}{T+1}\sum_{t=0}^{T}\mathbb{E}\left[\left\|\nabla F\left(\mathbf{x}^t\right)\right\|^2\right]\\ \leqslant&\frac{F\left(\mathbf{x}^0\right)-F^*}{\left(T+1\right)\left(\gamma\rho_1-\gamma^2\rho_2\right)}+\frac{\rho_3+\gamma\rho_4}{\rho_1-\gamma\rho_2}\\ =&\frac{F\left(\mathbf{x}^0\right)-F^*}{\lambda\sqrt{T+1}\rho_1-\lambda^2\rho_2}+\frac{\sqrt{T+1}\rho_3+\lambda\rho_4}{\sqrt{T+1}\rho_1-\lambda\rho_2},\end{aligned} \tag{31}$$

which completes the proof. □

Remark 2. *In Theorem 1, on the right-hand side of (18), the asymptotic learning error approaches*

$$\frac{\rho_3}{\rho_1}\approx\frac{\beta^2M^2\sqrt{2C_\alpha^2}\left(1-\frac{r}{M}\right)}{1-2\left(1-\frac{r}{M}\right)\sqrt{2C_\alpha^2}}, \tag{32}$$

as T approaches infinity, where the approximation holds based on $d_{\min}M\approx Nr$ when the values of $d_1,\ldots,d_M$ do not vary significantly from each other. From (32), as r increases, indicating a greater redundancy in the data allocation, the asymptotic learning error diminishes. In the special case where $r=M$, the asymptotic learning error equals zero, and CRA-DL converges without solution error. This aligns with our intuition that when each device holds a copy of the entire training dataset, even in the presence of Byzantine devices, the true global gradient can be recovered at the server if the number of honest devices exceeds that of the Byzantine devices.

Moreover, as C_α decreases, implying enhanced robustness of the RBA rules, the asymptotic learning error decreases as well. Since CRA-DL is a meta-algorithm that can be employed with any RBA rules, it is promising to achieve better learning performance when incorporating enhanced RBA rules.

Theorem 2 (Convergence performance of CRA-DL with decaying learning rates)**.** *Based on Assumptions 1-3, if $C_\alpha<\frac{d_{\min}M}{2\sqrt{2}N\left(r-\frac{r^2}{M}\right)}$, with decaying learning rates*

$$\gamma^t=\frac{\rho_1-\sqrt{\rho_1^2-4\rho_2\frac{\gamma^0\rho_1-\left(\gamma^0\right)^2\rho_2}{\sqrt{t+1}}}}{2\rho_2}, \tag{33}$$

for $\gamma^0<\frac{\rho_1}{2\rho_2}$, CRA-DL converges as

$$\begin{aligned}&\min_{0\leqslant t\leqslant T}\mathbb{E}\left[\left\|\nabla F\left(\mathbf{x}^t\right)\right\|^2\right]\\ \leqslant&\frac{F\left(\mathbf{x}^0\right)-F^*}{\left[\gamma^0\rho_1-\left(\gamma^0\right)^2\rho_2\right]\sqrt{T+1}}+\frac{\rho_3}{\rho_1-\gamma^0\rho_2}\\ &+\frac{\left(\gamma^0\right)^2\rho_4\left[2+\log\left(T+1\right)\right]}{\left[\gamma^0\rho_1-\left(\gamma^0\right)^2\rho_2\right]\sqrt{T+1}}.\end{aligned} \tag{34}$$

Proof. Similar as the proof of Theorem 1, we can derive (30). With the learning rates in (33), we have

$$\gamma^t\rho_1-\left(\gamma^t\right)^2\rho_2=\frac{\gamma^0\rho_1-\left(\gamma^0\right)^2\rho_2}{\sqrt{t+1}}, \tag{35}$$

where $\gamma^{t+1}<\gamma^t$ under the condition $\gamma^0<\frac{\rho_1}{2\rho_2}$. From (30) and (35), we have

$$\begin{aligned}&\min_{0\leqslant t\leqslant T}\mathbb{E}\left[\left\|\nabla F\left(\mathbf{x}^t\right)\right\|^2\right]\\ \leqslant&\frac{\frac{1}{T+1}\sum_{t=0}^{T}\left(\gamma^t\rho_1-\left(\gamma^t\right)^2\rho_2\right)\mathbb{E}\left[\left\|\nabla F\left(\mathbf{x}^t\right)\right\|^2\right]}{\frac{1}{T+1}\sum_{t=0}^{T}\left(\gamma^t\rho_1-\left(\gamma^t\right)^2\rho_2\right)}\\ \leqslant&\frac{\frac{F\left(\mathbf{x}^0\right)-F^*}{T+1}+\frac{1}{T+1}\sum_{t=0}^{T}\left[\gamma^t\rho_3+\left(\gamma^t\right)^2\rho_4\right]}{\frac{1}{T+1}\sum_{t=0}^{T}\left(\gamma^t\rho_1-\left(\gamma^t\right)^2\rho_2\right)}\\ =&\frac{F\left(\mathbf{x}^0\right)-F^*}{\sum_{t=0}^{T}\frac{\gamma^0\rho_1-\left(\gamma^0\right)^2\rho_2}{\sqrt{t+1}}}+\frac{\sum_{t=0}^{T}\left[\gamma^t\rho_3+\left(\gamma^t\right)^2\rho_4\right]}{\sum_{t=0}^{T}\frac{\gamma^0\rho_1-\left(\gamma^0\right)^2\rho_2}{\sqrt{t+1}}}\\ \leqslant&\frac{F\left(\mathbf{x}^0\right)-F^*}{\left[\gamma^0\rho_1-\left(\gamma^0\right)^2\rho_2\right]\sqrt{T+1}}+\frac{\sum_{t=0}^{T}\left[\gamma^t\rho_3+\left(\gamma^t\right)^2\rho_4\right]}{\sum_{t=0}^{T}\frac{\gamma^0\rho_1-\left(\gamma^0\right)^2\rho_2}{\sqrt{t+1}}},\end{aligned} \tag{36}$$

where the equality is obtained by substituting (33) into (36), and the last equality is derived from the following inequality:

$$\sum_{t=0}^{T}\frac{1}{\sqrt{t+1}}\geqslant\sqrt{T+1}. \tag{37}$$

$$\frac{\rho_3}{\rho_1-\gamma^0\rho_2}\approx\frac{\beta^2M^2\sqrt{2C_\alpha^2}\left(1-\frac{r}{M}\right)}{1-2\left(1-\frac{r}{M}\right)\sqrt{2C_\alpha^2}-\gamma^0\left\{\frac{(\phi_1-\phi_2)2L}{(1-\alpha)^2N^2}+\frac{\phi_2L}{(1-\alpha)^2N}+8\frac{LC_\alpha^2}{N}\left(1-\frac{r}{M}\right)^2\right\}}. \tag{40}$$

Based on (35), we have

$$\begin{aligned}\gamma^t\rho_1-\left(\gamma^t\right)^2\rho_2=\gamma^t\left(\rho_1-\gamma^t\rho_2\right)=&\frac{\gamma^0\rho_1-\left(\gamma^0\right)^2\rho_2}{\sqrt{t+1}}\\ \geqslant&\gamma^t\left(\rho_1-\gamma^0\rho_2\right),\end{aligned} \tag{38}$$

which indicates $\frac{\gamma^0}{\sqrt{t+1}}\geqslant\gamma^t$. Based on that, we can bound the second term on the right hand side of (36) as

$$\begin{aligned}&\frac{\sum_{t=0}^{T}\left[\gamma^t\rho_3+\left(\gamma^t\right)^2\rho_4\right]}{\sum_{t=0}^{T}\frac{\gamma^0\rho_1-(\gamma^0)^2\rho_2}{\sqrt{t+1}}}\leqslant\frac{\sum_{t=0}^{T}\left[\frac{\gamma^0}{\sqrt{t+1}}\rho_3+\left(\frac{\gamma^0}{\sqrt{t+1}}\right)^2\rho_4\right]}{\sum_{t=0}^{T}\frac{\gamma^0\rho_1-(\gamma^0)^2\rho_2}{\sqrt{t+1}}}\\ =&\frac{\sum_{t=0}^{T}\left[\frac{\gamma^0}{\sqrt{t+1}}\rho_3\right]}{\sum_{t=0}^{T}\frac{\gamma^0\rho_1-(\gamma^0)^2\rho_2}{\sqrt{t+1}}}+\frac{\sum_{t=0}^{T}\left[\frac{\left(\gamma^0\right)^2}{t+1}\rho_4\right]}{\sum_{t=0}^{T}\frac{\gamma^0\rho_1-(\gamma^0)^2\rho_2}{\sqrt{t+1}}}\\ \leqslant&\frac{\rho_3}{\rho_1-\gamma^0\rho_2}+\frac{\left(\gamma^0\right)^2\rho_4\left[2+\log\left(T+1\right)\right]}{\left[\gamma^0\rho_1-\left(\gamma^0\right)^2\rho_2\right]\sqrt{T+1}},\end{aligned} \tag{39}$$

where the last inequality is derived from (37) and the inequality $\sum_{t=0}^{T}\frac{1}{t+1}\leqslant 2+\log\left(T+1\right)$. Finally, substituting (39) into (36) yields Theorem 2. □

Remark 3. *In Theorem 2, on the right-hand side of (34), when T approaches infinity, the asymptotic learning error approaches (40), based on $d_{\min}M\approx Nr$ when the values of $d_1,\ldots,d_M$ do not vary significantly from each other. When r increases, i.e., the redundancy of data allocation increases, the asymptotic learning error diminishes. Specially, the asymptotic learning error reaches zero for $r=M$. In addition, by decreasing the value of C_α, the asymptotic learning error decreases and better learning performance can be attained by CRA-DL with decaying learning rates.*

Remark 4. *We would like to note that the numerical results in Section V will demonstrate that the asymptotic learning error in CRA-DL is indeed very close to zero in practice. In other words, CRA-DL can consistently converge to the optimal point with almost no solution error across various scenarios.*

Remark 5. *In Theorem 1 and Theorem 2, the bounds include the constant C_α. Note that C_α depends on the fraction of Byzantine devices, i.e., α, as observed from the values of C_α^2 for some commonly used RBA rules provided in Table I. Based on this, the bounds in Theorem 1 and Theorem 2 are both determined by the fraction of Byzantine devices.*

V. Numerical results

In this section, we demonstrate the performance of the proposed method through numerical results on a linear regression task with a synthetic dataset. Three commonly encountered Byzantine attacks are considered, which are listed below:

1) **Sign-flipping attack [36], [38].** Each Byzantine device transmits the true message multiplied by a negative coefficient. This coefficient is set to -2 in our simulations.
2) **Gaussian attack [36], [38].** Each Byzantine device transmits a vector of the same size as the true message, where the elements are randomly drawn from the Gaussian distribution $\mathcal{N}(0,10000)$.
3) **Sample-duplicating attack [39], [40].** Each Byzantine device randomly selects an honest device, duplicates the message of the selected honest device and sends this message to the server.

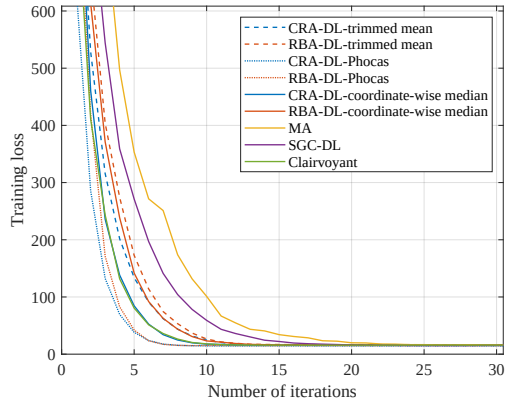
For the proposed method, we adopt three RBA rules, namely coordinate-wise median [13], trimmed mean [13], and Phocas [17]. For comparison, the following baseline methods are considered:

- **Mean Averaging (MA):** The training data are allocated to the devices non-redundantly, and the server uses the mean aggregation rule to aggregate the local gradients from the honest devices and the disruptive messages from the Byzantine devices.
- **RBA-DL:** The training data are allocated to the devices non-redundantly. The server applies RBA rules to the local gradients from the honest devices and the disruptive messages from the Byzantine devices.
- **DL in the original SGC scheme (SGC-DL):** The training data are allocated to the devices in a pair-wise balanced manner before training. During training iterations, the honest devices transmit coded gradients to the server, and the server aggregates the coded gradients from the honest devices and the disruptive information from the Byzantine devices using the mean aggregation rule.
- **Clairvoyant Method:** The training data are allocated to the devices non-redundantly. The server knows the identities of the devices in each iteration and only aggregates the local gradients from the honest devices using the mean aggregation rule.

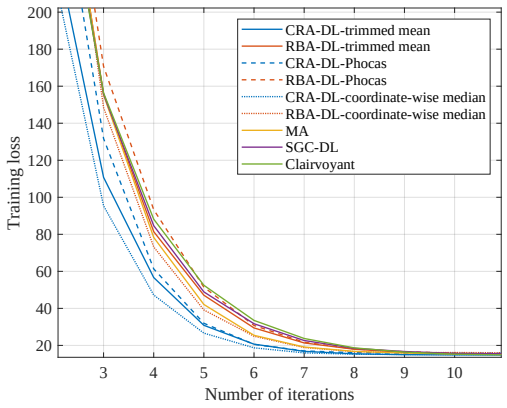
In the linear regression task, the number of devices is $N=100$ and the loss function can be expressed as

$$F\left(\mathbf{x}\right)=\sum_{k=1}^{m}f_k\left(\mathbf{x}\right),f_k\left(\mathbf{x}\right)=\frac{1}{2}(\langle\mathbf{x},\mathbf{z}_k\rangle-y_k)^2, \tag{41}$$

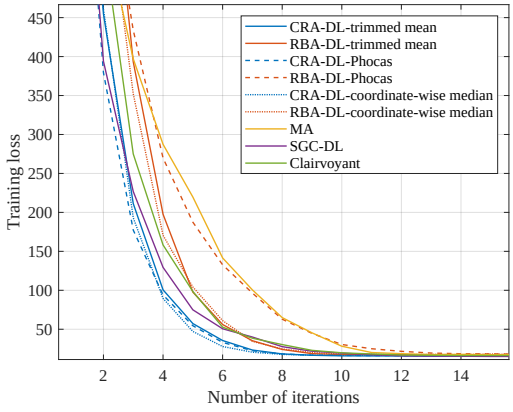
where $m=1000$, $\mathbf{z}_k\in\mathbb{R}^{100}$, $y_k\in\mathbb{R}$, $k=1,...,1000$, and $\mathbf{x}\in\mathbb{R}^{100}$. In this problem, the overall dataset $\mathcal{D}$ consists of $m=1000$ training data samples $\{\mathbf{z}_k,y_k\}$, which are divided into 1000 subsets, each containing one data sample. Before the training starts, the training subsets are allocated uniformly and randomly to the devices so that each device obtains r subsets, which is an effective approximation of the



(a)



(b)



(c)

Figure 2. Training loss as a function of the number of iterations for different methods under various Byzantine attacks. (a) Under Sign-flipping attack. (b) Under Gaussian attack. (c) Under Sample-duplicating attack.

pair-wise balanced allocation of the subsets. In (41), all the elements in $\{\mathbf{z}_1, \ldots, \mathbf{z}_m\}$ are drawn independently from the normal distribution $\mathcal{N}(0, 100)$. To generate the values of y_k, we first generate a random vector $\widehat{\mathbf{x}}$ whose 100 elements are drawn from the standard normal distribution. Accordingly, y_k is generated as $y_k \sim \mathcal{N}\left(\left\langle \mathbf{z}_k, \widehat{\mathbf{x}} \right\rangle, 1\right)$, $\forall k$. Unless specified, the learning rate is fixed at $\gamma = 0.001$ and we set $r = 40$ in our method.

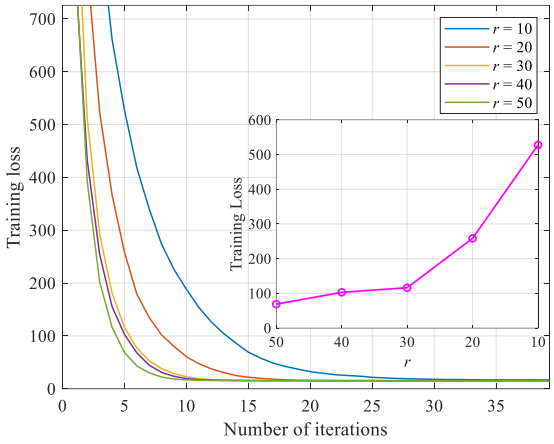
To compare the performance of the proposed method with the baseline methods, we plot the training loss as a function of the number of iterations for different methods under various types of Byzantine attacks in Fig. 2, where a number of RBA rules are applied for the proposed CRA-DL method and the baseline RBA-DL method. Under the Sign-flipping attack, we set $\alpha = 0.2$; under the Gaussian attack, we set $\alpha = 0.03$; and under the Sample-duplicating attack, we set $\alpha = 0.4$. It can be observed that the learning performance of MA is easily influenced by the Byzantine attacks, particularly under the Sign-flipping and Sample-duplicating attacks, which is because the disruptive information from Byzantine devices and the true information from honest devices are treated equally during aggregation. This aligns with our intuition. Among all the methods, the proposed method attains the best learning performance compared to the baseline methods. The advantage of our method over RBA-DL and SGC-DL lies in its ability to leverage the strengths of RBA rules and SGC simultaneously. This improves the robustness of the RBA rules by reducing the distance among messages sent by honest devices and more effectively mitigates the negative impact of disruptive information sent by Byzantine devices. It is worth noting that the clairvoyant method does not achieve the best learning performance, even though the server knows the identities of all devices. This is because the server discards the messages from Byzantine devices, meaning the data samples allocated to those devices, without allocation redundancy, cannot be utilized to update the global model. In contrast, our method allocates training data redundantly to devices. This allows our method not only to evade the disruptive information from Byzantine devices, preventing it from misleading the learning process, but also to compensate for the missing information from Byzantine devices with messages from honest devices based on data allocation redundancy.

To investigate the influence of data allocation redundancy on the learning performance of our method, we depict the training loss of CRA-DL as a function of the number of iterations under various values of r in Fig. 3. To clearly illustrate the trade-off, we also present the Pareto front between data allocation redundancy and learning performance under a fixed number of iterations. In Fig. 3a, the Pareto front is shown after 5 iterations, while in Fig. 3b, it is shown after 10 iterations. In this scenario, a Sign-flipping attack is considered, and the coordinate-wise median is adopted as the RBA rule. It can be observed that as the value of r increases, indicating greater redundancy in data allocation, the learning performance of the proposed method improves. This observation aligns with both our theoretical analysis in Section IV and our intuition, which implies that improving learning performance and enhancing robustness against Byzantine attacks come at the cost of increased computation and storage burdens on the devices. In practice, an appropriate trade-off between learning performance and the computation and storage burdens should be determined.

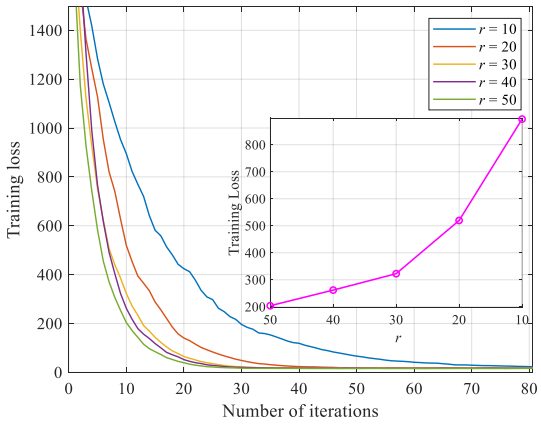
To verify that the asymptotic learning error of CRA-DL is negligible and that CRA-DL can converge to the optimal point with almost no solution error, we plot the training loss as a function of the number of iterations for CRA-DL under

various values of α in Fig. 4, where a Sign-flipping attack is considered and the coordinate-wise median is adopted as the RBA rule. It can be seen that as the value of α decreases, the learning performance of the proposed method improves. This aligns with our intuition, considering that better learning performance is expected with fewer Byzantine devices. When $\alpha = 0$, there is no Byzantine attack in the system, and it is guaranteed to attain the optimal point. From Fig. 4, we can observe that the training loss under different values of α converges to the same value, demonstrating that the proposed method incurs negligible solution error even under Byzantine attacks. In addition, as shown in Fig. 4, when $\alpha = 0.1$, the proposed method achieves almost the same learning performance as in the case without any Byzantine attacks. It is worth noting that existing gradient coding methods designed to handle Byzantine attacks, such as [22], can also achieve the same learning performance, matching that of the case without Byzantine attacks. However, as pointed out in [22], the lower bound on the redundancy level in training data allocation required to achieve such robustness is $r = 110$ in this considered setting. In contrast, the proposed method achieves comparable learning performance with a significantly lower redundancy level of $r = 40$. This substantial reduction in redundancy leads to much lower computational and storage burdens on the devices, highlighting the practical efficiency of the proposed approach under Byzantine attacks. The rationale behind the superiority of the proposed method is as follows. Given that machine learning algorithms are generally robust to noise, it is not necessary to fully recover the true global gradient in each iteration to update the global model, as is required in existing gradient coding methods designed to handle Byzantine attacks. Instead, it is often more efficient to obtain an approximate version of the true global gradient, as is done in the proposed method. As a result, the proposed method only requires a modest level of redundancy to achieve the same level of robustness to Byzantine attacks, thereby reducing the computational and storage burdens on the devices.

To demonstrate the robustness of the proposed method to the heterogeneity among data subsets in the training data, we compare its performance with RBA-DL and plot the training loss as a function of the number of iterations in Fig. 5 for both methods under various levels of heterogeneity, where the Sign-flipping attack is adopted with $\alpha = 0.2$. Here, both methods use coordinate-wise median as the RBA rule for server-side aggregation. The key difference is that the proposed method aggregates coded gradients, whereas RBA-DL aggregates local gradients directly. To control the heterogeneity among the data subsets, y_k is generated as $y_k \sim \mathcal{N}\left(\left\langle \mathbf{z}_k, \widehat{\mathbf{x}} + \tilde{\mathbf{x}}_k \right\rangle, 1\right)$, for all k, where $\tilde{\mathbf{x}}_k \sim \mathcal{N}(0, \sigma_H^2)$. A larger value of σ_H corresponds to a higher level of heterogeneity among the subsets. As shown in Fig. 5, the learning performance of the proposed method is significantly better than that of the baseline, especially under high heterogeneity. This is because, in such scenarios, the local gradients sent by honest devices differ significantly, which causes the global update obtained through direct aggregation in RBA-DL to deviate more from the true global gradient. In contrast, the coded gradients sent by honest devices in



(a)



(b)

Figure 3. Training loss as a function of the number of iterations for CRA-DL under various values of r. (a) $\alpha = 0.2$. (b) $\alpha = 0.4$.

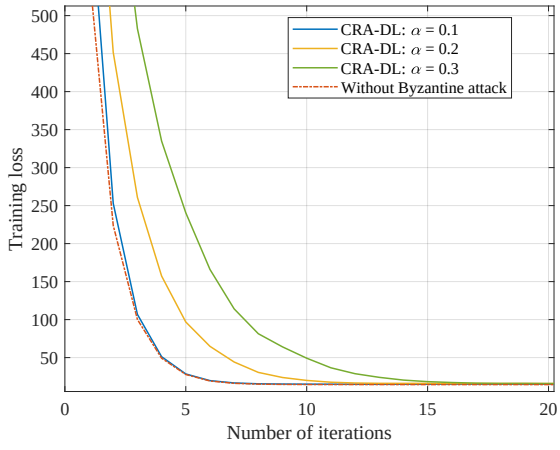


Figure 4. Training loss as a function of the number of iterations for CRA-DL under various values of α, where the case "Without Byzantine attack" corresponds to $\alpha = 0$.

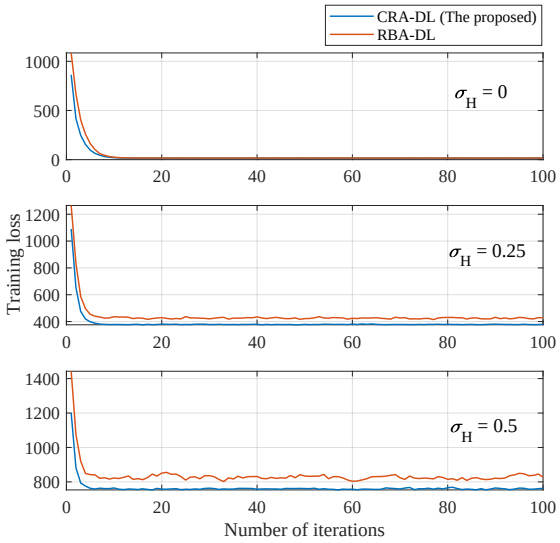


Figure 5. Training loss as a function of the number of iterations for CRA-DL and RBA-DL under various levels of heterogeneity among data subsets.

the proposed method remain close to each other, even under substantial heterogeneity. This improves the robustness of the RBA rule at the server and results in a more accurate global model update, thereby enhancing learning performance.

VI. Conclusions

In this paper, the DL problem under Byzantine attacks was studied. To overcome the limitation of current DL methods applying RBA rules, we proposed CRA-DL, a new DL method dealing with Byzantine attacks based on coded robust aggregation. In the proposed method, before the training, the training data are allocated to the devices in a pair-wise balanced manner. During training iterations, the server receives coded gradients from the honest devices and disruptive information from the Byzantine devices, and aggregates these information using RBA rules. By doing this, the global gradient is approximately recovered by the server to update the global model. The convergence performance of CRA-DL was analyzed and we provided numerical results to demonstrate the superiority of the proposed method compared to the baselines. The proposed CRA-DL method has important practical implications for real-world DL systems. First, it enhances robustness to Byzantine attacks while requiring only a modest level of data allocation redundancy, thereby reducing the computational and storage burdens on devices. This is an essential advantage for resource-constrained systems. Second, CRA-DL maintains strong robustness in the presence of data heterogeneity among subsets, making it suitable for applications such as healthcare, finance, and autonomous systems, where Byzantine resilience is critically needed under diverse operating conditions. Finally, as a meta-algorithm, CRA-DL is compatible with a wide range of existing RBA rules, enabling seamless integration into existing DL frameworks that already employ such rules. In the future, we plan to extend the proposed method to scenarios where communication resources are highly limited, by compressing the communication between the devices and the server to mitigate the communication overhead. Besides, we will extend CRA-DL to a variety of real-world application scenarios to further validate its effectiveness in various practical settings.

Appendix A
Proof of Lemma 1

Based on allocation of the training data in the pair-wise balanced manner, for $\forall i \neq j$, we have

$$\begin{aligned}\mathbf{g}_i^t - \mathbf{g}_j^t &= \sum_{k_1 \in \{ k_1 | s(i,k_1) \neq 0\}} \frac{1}{d_{k_1}} \nabla f_{k_1}\left(\mathbf{x}^t\right) \\ &- \sum_{k_2 \in \{ k_2 | s(j,k_2) \neq 0\}} \frac{1}{d_{k_2}} \nabla f_{k_2}\left(\mathbf{x}^t\right) \\ &= \sum_{k_1 \in \{ k_1 | s(i,k_1) \neq 0, s(j,k_1) = 0\}} \frac{1}{d_{k_1}} \nabla f_{k_1}\left(\mathbf{x}^t\right) \\ &- \sum_{k_2 \in \{ k_2 | s(i,k_2) = 0, s(j,k_2) \neq 0\}} \frac{1}{d_{k_2}} \nabla f_{k_2}\left(\mathbf{x}^t\right), \end{aligned} \tag{42}$$

according to (5). Based on (42), we can obtain (44), where the first three inequalities are derived from the following basic inequality:

$$\left\| \sum_{i=1}^{n} \mathbf{a}_i \right\|^2 \leq n \sum_{i=1}^{n} \left\| \mathbf{a}_i \right\|^2, \forall \mathbf{a}_i \in \mathbb{R}^D, \tag{43}$$

together with the fact that the set $\{ k_1 |\, s\left(i, k_1\right) \neq 0, s\left(j, k_1\right) = 0 \}$ and the set $\{ k_2 |\, s\left(i, k_2\right) = 0, s\left(j, k_2\right) \neq 0 \}$ both contain $\left(r - \frac{r^2}{M}\right)$ elements, the fourth inequality is derived from Assumption 2, and the last inequality can be easily obtained according to the definition of d_{min}. This completes the proof.

Appendix B
Proof of Lemma 2

Let us define

$$\mathbf{A}^t \triangleq \left[\frac{1}{d_1} \nabla f_1\left(\mathbf{x}^t\right), \frac{1}{d_2} \nabla f_2\left(\mathbf{x}^t\right), ..., \frac{1}{d_M} \nabla f_M\left(\mathbf{x}^t\right) \right], \tag{45}$$

and

$$\mathbf{G}^t \triangleq \mathbf{A}^t \mathbf{S}^T = \left[\mathbf{g}_1^t, ..., \mathbf{g}_N^t \right], \tag{46}$$

which are two matrices of size $D \times M$ and size $D \times N$, respectively. In (46), $\mathbf{S}^T$ is the transpose of the data allocation matrix $\mathbf{S}$. In addition, we define $\mathbf{h}^t$, an $N \times 1$ vector, to indicate the identities of the devices in iteration t, where the i-th element being 1 implies device i is honest in iteration t, and the the i-th element being 0 implies the opposite.

Based on the above definitions, we can express

$$\bar{\mathbf{g}}^t = \frac{1}{|\mathcal{H}^t|} \sum_{i \in \mathcal{H}^t} \mathbf{g}_i^t = \mathbf{G}^t \mathbf{h}^t \frac{1}{(1-\alpha) N}. \tag{47}$$

From (47), we have

$$\begin{aligned}\left\| \bar{\mathbf{g}}^t \right\|^2 &= \frac{1}{(1-\alpha)^2 N^2} \left(\mathbf{h}^t\right)^T \left(\mathbf{G}^t\right)^T \mathbf{G}^t \mathbf{h}^t \\ &= \frac{1}{(1-\alpha)^2 N^2} Tr\left[\mathbf{h}^t \left(\mathbf{h}^t\right)^T \left(\mathbf{G}^t\right)^T \mathbf{G}^t \right], \end{aligned} \tag{48}$$

$$
\begin{aligned}
&\max_{i,j\in\{1,\ldots,N\}} \left\|\mathbf{g}_i^t-\mathbf{g}_j^t\right\|^2\\
\leqslant& 2\max_{i,j\in\{1,\ldots,N\}} \left\|\sum_{k_1\in\{k_1|s(i,k_1)\neq 0,s(j,k_1)=0\}} \frac{1}{d_{k_1}}\nabla f_{k_1}\left(\mathbf{x}^t\right)\right\|^2+\left\|\sum_{k_2\in\{k_2|s(i,k_2)=0,s(j,k_2)\neq 0\}} \frac{1}{d_{k_2}}\nabla f_{k_2}\left(\mathbf{x}^t\right)\right\|^2\\
\leqslant& 2\max_{i,j\in\{1,\ldots,N\}} \left\{\left(r-\frac{r^2}{M}\right)\sum_{k_1\in\{k_1|s(i,k_1)\neq 0,s(j,k_1)=0\}} \left\|\frac{1}{d_{k_1}}\nabla f_{k_1}\left(\mathbf{x}^t\right)-\frac{1}{M}\frac{1}{d_{k_1}}\nabla F\left(\mathbf{x}^t\right)+\frac{1}{M}\frac{1}{d_{k_1}}\nabla F\left(\mathbf{x}^t\right)\right\|^2\right.\\
&\left.+\left(r-\frac{r^2}{M}\right)\sum_{k_2\in\{k_2|s(i,k_2)=0,s(j,k_2)\neq 0\}} \left\|\frac{1}{d_{k_2}}\nabla f_{k_2}\left(\mathbf{x}^t\right)-\frac{1}{M}\frac{1}{d_{k_2}}\nabla F\left(\mathbf{x}^t\right)+\frac{1}{M}\frac{1}{d_{k_2}}\nabla F\left(\mathbf{x}^t\right)\right\|^2\right\}\\
\leqslant& 2\max_{i,j\in\{1,\ldots,N\}} \left\{2\left(r-\frac{r^2}{M}\right)\sum_{k_1\in\{k_1|s(i,k_1)\neq 0,s(j,k_1)=0\}} \left\|\frac{1}{d_{k_1}}\nabla f_{k_1}\left(\mathbf{x}^t\right)-\frac{1}{M}\frac{1}{d_{k_1}}\nabla F\left(\mathbf{x}^t\right)\right\|^2+\left\|\frac{1}{M}\frac{1}{d_{k_1}}\nabla F\left(\mathbf{x}^t\right)\right\|^2\right.\\
&\left.+2\left(r-\frac{r^2}{M}\right)\sum_{k_2\in\{k_2|s(i,k_2)=0,s(j,k_2)\neq 0\}} \left\|\frac{1}{d_{k_2}}\nabla f_{k_2}\left(\mathbf{x}^t\right)-\frac{1}{M}\frac{1}{d_{k_2}}\nabla F\left(\mathbf{x}^t\right)\right\|^2+\left\|\frac{1}{M}\frac{1}{d_{k_2}}\nabla F\left(\mathbf{x}^t\right)\right\|^2\right\}\\
\leqslant& 2\max_{i,j\in\{1,\ldots,N\}} \left\{2\left(r-\frac{r^2}{M}\right)\sum_{k_1\in\{k_1|s(i,k_1)\neq 0,s(j,k_1)=0\}} \left(\frac{1}{d_{k_1}^2}\beta^2+\frac{1}{d_{k_1}^2M^2}\left\|\nabla F\left(\mathbf{x}^t\right)\right\|^2\right)\right.\\
&\left.+2\left(r-\frac{r^2}{M}\right)\sum_{k_2\in\{k_2|s(i,k_2)=0,s(j,k_2)\neq 0\}} \left(\frac{1}{d_{k_2}^2}\beta^2+\frac{1}{d_{k_2}^2M^2}\left\|\nabla F\left(\mathbf{x}^t\right)\right\|^2\right)\right\}\\
\leqslant& 8\frac{1}{d_{\min}^2}\left(r-\frac{r^2}{M}\right)^2\left(\beta^2+\frac{1}{M^2}\left\|\nabla F\left(\mathbf{x}^t\right)\right\|^2\right). \qquad (44)
\end{aligned}
$$

where $Tr(\cdot)$ is the trace of a square matrix. According to (48), we can derive

$$
\begin{aligned}
&\mathbb{E}\left(\left\|\bar{\mathbf{g}}^t\right\|^2\middle|\mathcal{F}^t\right)\\
=&\frac{1}{(1-\alpha)^2N^2}\mathbb{E}\left(Tr\left[\mathbf{h}^t\left(\mathbf{h}^t\right)^T\left(\mathbf{G}^t\right)^T\mathbf{G}^t\right]\middle|\mathcal{F}^t\right)\\
=&\frac{1}{(1-\alpha)^2N^2}Tr\left[\mathbb{E}\left[\mathbf{h}^t\left(\mathbf{h}^t\right)^T\right]\left(\mathbf{G}^t\right)^T\mathbf{G}^t\right]. \qquad (49)
\end{aligned}
$$

In (49), we have

$$
\mathbb{E}\left[\mathbf{h}^t\left(\mathbf{h}^t\right)^T\right]=(\phi_1-\phi_2)\mathbf{I}+\phi_2\mathbf{1}\mathbf{1}^T. \qquad (50)
$$

This is derived from the fact that, in each iteration, a fraction α of the devices are Byzantine devices, which is totally random. From this perspective, for $i\neq j$, the probability of device i and j being honest devices is $\Pr\left(h_i^t=1,h_j^t=1\right)=\phi_2$ and the probability of device i being honest is $\Pr\left(h_i^t=1\right)=\phi_1$, where ϕ_1 and ϕ_2 are defined in (17).

Next, substituting (50) into (49), we have

$$
\begin{aligned}
&\mathbb{E}\left(\left\|\bar{\mathbf{g}}^t\right\|^2\middle|\mathcal{F}^t\right)\\
=&\frac{1}{(1-\alpha)^2N^2}Tr\left[\left((\phi_1-\phi_2)\mathbf{I}+\phi_2\mathbf{1}\mathbf{1}^T\right)\left(\mathbf{G}^t\right)^T\mathbf{G}^t\right]\\
=&\frac{1}{(1-\alpha)^2N^2}Tr\left[(\phi_1-\phi_2)\mathbf{I}\left(\mathbf{G}^t\right)^T\mathbf{G}^t\right]\\
&+\frac{1}{(1-\alpha)^2N^2}Tr\left[\phi_2\mathbf{1}\mathbf{1}^T\left(\mathbf{G}^t\right)^T\mathbf{G}^t\right]\\
=&\frac{(\phi_1-\phi_2)}{(1-\alpha)^2N^2}\sum_{i=1}^N\left\|\mathbf{g}_i^t\right\|^2+\frac{\phi_2}{(1-\alpha)^2N^2}Tr\left[\mathbf{1}^T\left(\mathbf{G}^t\right)^T\mathbf{G}^t\mathbf{1}\right]\\
=&\frac{(\phi_1-\phi_2)}{(1-\alpha)^2N^2}\sum_{i=1}^N\left\|\mathbf{g}_i^t\right\|^2+\frac{\phi_2}{(1-\alpha)^2N^2}\left\|\nabla F\left(\mathbf{x}^t\right)\right\|^2, \qquad (51)
\end{aligned}
$$

based on (5) and (46). In (51), we can derive the bound for $\sum_{i=1}^N\left\|\mathbf{g}_i^t\right\|^2$ in (52) by applying the basic inequality in (43) and Assumption 2. Substituting (52) into (51), we have (16), which completes the proof.

REFERENCES

[1] M. Chen, D. Gündüz, K. Huang, W. Saad, M. Bennis, A. V. Feljan, and H. V. Poor, "Distributed learning in wireless networks: Recent progress and future challenges," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3579–3605, 2021.

[2] J. Verbraeken, M. Wolting, J. Katzy, J. Kloppenburg, T. Verbelen, and J. S. Rellermeyer, "A survey on distributed machine learning," *Acm computing surveys (csur)*, vol. 53, no. 2, pp. 1–33, 2020.

[3] M. M. H. Shuvo, S. K. Islam, J. Cheng, and B. I. Morshed, "Efficient acceleration of deep learning inference on resource-constrained edge devices: A review," *Proceedings of the IEEE*, vol. 111, no. 1, pp. 42–91, 2022.

[4] X. Wang, Y. Han, V. C. Leung, D. Niyato, X. Yan, and X. Chen, "Convergence of edge computing and deep learning: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 2, pp. 869–904, 2020.

[5] S. AbdulRahman, H. Tout, H. Ould-Slimane, A. Mourad, C. Talhi, and M. Guizani, "A survey on federated learning: The journey from centralized to distributed on-site learning and beyond," *IEEE Internet of Things Journal*, vol. 8, no. 7, pp. 5476–5497, 2020.

$$\begin{aligned}
&\sum_{i=1}^{N}\left\|\mathbf{g}_i^t\right\|^2=\sum_{i=1}^{N}\left\|\sum_{k\in\{\,k|s(i,k)\neq 0\}}\frac{1}{d_k}\nabla f_k\left(\mathbf{x}^t\right)\right\|^2\\
\leqslant&\frac{r}{d_{\min}^2}\sum_{i=1}^{N}\sum_{k\in\{\,k|s(i,k)\neq 0\}}\left\|\nabla f_k\left(\mathbf{x}^t\right)\right\|^2=\frac{r}{d_{\min}^2}\sum_{i=1}^{N}\sum_{k\in\{\,k|s(i,k)\neq 0\}}\left\|\nabla f_k\left(\mathbf{x}^t\right)-\frac{1}{M}\nabla F\left(\mathbf{x}^t\right)+\frac{1}{M}\nabla F\left(\mathbf{x}^t\right)\right\|^2\\
\leqslant&\frac{2r}{d_{\min}^2}\sum_{i=1}^{N}\sum_{k\in\{\,k|s(i,k)\neq 0\}}\left\{\left\|\nabla f_k\left(\mathbf{x}^t\right)-\frac{1}{M}\nabla F\left(\mathbf{x}^t\right)\right\|^2+\left\|\frac{1}{M}\nabla F\left(\mathbf{x}^t\right)\right\|^2\right\}\\
\leqslant&\frac{2r}{d_{\min}^2}\sum_{i=1}^{N}\sum_{k\in\{\,k|s(i,k)\neq 0\}}\left[\beta^2+\left\|\frac{1}{M}\nabla F\left(\mathbf{x}^t\right)\right\|^2\right]=\frac{2Nr^2}{d_{\min}^2}\left[\beta^2+\frac{1}{M^2}\left\|\nabla F\left(\mathbf{x}^t\right)\right\|^2\right].
\end{aligned}\tag{52}$$

[6] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," *Advances in neural information processing systems*, vol. 30, 2017.

[7] R. Alkhadra, J. Abuzaid, M. AlShammari, and N. Mohammad, "Solar winds hack: In-depth analysis and countermeasures," in *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*. IEEE, 2021, pp. 1–7.

[8] 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 Processing Magazine*, vol. 37, no. 3, pp. 146–159, 2020.

[9] Z. Zhang and Y. Li, "Nspfl: A novel secure and privacy-preserving federated learning with data integrity auditing," *IEEE Transactions on Information Forensics and Security*, 2024.

[10] Z. Zhang, L. Wu, C. Ma, J. Li, J. Wang, Q. Wang, and S. Yu, "Lsfl: A lightweight and secure federated learning scheme for edge computing," *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 365–379, 2022.

[11] A. Abrardo, M. Barni, K. Kallas, and B. Tondi, "A game-theoretic framework for optimum decision fusion in the presence of byzantines," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 6, pp. 1333–1345, 2016.

[12] X. Liu, T. J. Lim, and J. Huang, "Optimal byzantine attacker identification based on game theory in network coding enabled wireless ad hoc networks," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 2570–2583, 2020.

[13] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *International Conference on Machine Learning*. Pmlr, 2018, pp. 5650–5659.

[14] S. P. Karimireddy, L. He, and M. Jaggi, "Learning from history for byzantine robust optimization," in *International Conference on Machine Learning*. PMLR, 2021, pp. 5311–5319.

[15] Y. Chen, L. Su, and J. Xu, "Distributed statistical machine learning in adversarial settings: Byzantine gradient descent," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 1, no. 2, pp. 1–25, 2017.

[16] Q. Xia, Z. Tao, Z. Hao, and Q. Li, "Faba: an algorithm for fast aggregation against byzantine attacks in distributed neural networks," in *IJCAI*, 2019.

[17] C. Xie, O. Koyejo, and I. Gupta, "Phocas: dimensional byzantine-resilient stochastic gradient descent," *arXiv preprint arXiv:1805.09682*, 2018.

[18] P. Zhao, F. Yu, and Z. Wan, "A huber loss minimization approach to byzantine robust federated learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 19, 2024, pp. 21 806–21 814.

[19] X. Dong, Z. Wu, Q. Ling, and Z. Tian, "Byzantine-robust distributed online learning: Taming adversarial participants in an adversarial environment," *IEEE Transactions on Signal Processing*, 2023.

[20] S. Hong, H. Yang, Y. Yoon, and J. Lee, "Group-wise verifiable coded computing under byzantine attacks and stragglers," *IEEE Transactions on Information Forensics and Security*, 2024.

[21] D. Data, L. Song, and S. N. Diggavi, "Data encoding for byzantine-resilient distributed optimization," *IEEE Transactions on Information Theory*, vol. 67, no. 2, pp. 1117–1140, 2020.

[22] C. Hofmeister, L. Maßny, E. Yaakobi, and R. Bitar, "Byzantine-resilient gradient coding through local gradient computations," *IEEE Transactions on Information Theory*, 2025.

[23] L. Chen, H. Wang, Z. Charles, and D. Papailiopoulos, "Draco: Byzantine-resilient distributed training via redundant gradients," in *International Conference on Machine Learning*. PMLR, 2018, pp. 903–912.

[24] R. Tandon, Q. Lei, A. G. Dimakis, and N. Karampatziakis, "Gradient coding: Avoiding stragglers in distributed learning," in *International Conference on Machine Learning*. PMLR, 2017, pp. 3368–3376.

[25] E. Ozfatura, D. Gündüz, and S. Ulukus, "Gradient coding with clustering and multi-message communication," in *2019 IEEE Data Science Workshop (DSW)*. IEEE, 2019, pp. 42–46.

[26] B. Buyukates, E. Ozfatura, S. Ulukus, and D. Gündüz, "Gradient coding with dynamic clustering for straggler-tolerant distributed learning," *IEEE Transactions on Communications*, 2022.

[27] M. Glasgow and M. Wootters, "Approximate gradient coding with optimal decoding," *IEEE Journal on Selected Areas in Information Theory*, vol. 2, no. 3, pp. 855–866, 2021.

[28] H. Wang, Z. Charles, and D. Papailiopoulos, "Erasurehead: Distributed gradient descent without delays using approximate gradient coding," *arXiv preprint arXiv:1901.09671*, 2019.

[29] R. Bitar, M. Wootters, and S. El Rouayheb, "Stochastic gradient coding for straggler mitigation in distributed learning," *IEEE Journal on Selected Areas in Information Theory*, vol. 1, no. 1, pp. 277–291, 2020.

[30] C. Li and M. Skoglund, "Distributed learning based on 1-bit gradient coding in the presence of stragglers," *IEEE Transactions on Communications*, 2024.

[31] ——, "Gradient coding in decentralized learning for evading stragglers," *arXiv preprint arXiv:2402.04193*, 2024.

[32] J. Liu, J. Huang, Y. Zhou, X. Li, S. Ji, H. Xiong, and D. Dou, "From distributed machine learning to federated learning: A survey," *Knowledge and Information Systems*, vol. 64, no. 4, pp. 885–917, 2022.

[33] K. Mishchenko, E. Gorbunov, M. Takáč, and P. Richtárik, "Distributed learning with compressed gradient differences," *Optimization Methods and Software*, pp. 1–16, 2024.

[34] A. Beznosikov, S. Horváth, P. Richtárik, and M. Safaryan, "On biased compression for distributed learning," *Journal of Machine Learning Research*, vol. 24, no. 276, pp. 1–50, 2023.

[35] E. Gorbunov, K. P. Burlachenko, Z. Li, and P. Richtárik, "Marina: Faster non-convex distributed learning with compression," in *International Conference on Machine Learning*. PMLR, 2021, pp. 3788–3798.

[36] H. Zhu and Q. Ling, "Byzantine-robust distributed learning with compression," *IEEE Transactions on Signal and Information Processing over Networks*, 2023.

[37] R. Jin, Y. Liu, Y. Huang, X. He, T. Wu, and H. Dai, "Sign-based gradient descent with heterogeneous data: Convergence and byzantine resilience," *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[38] J. Shi, W. Wan, S. Hu, J. Lu, and L. Y. Zhang, "Challenges and approaches for mitigating byzantine attacks in federated learning," in *2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. IEEE, 2022, pp. 139–146.

[39] Z. Wu, T. Chen, and Q. Ling, "Byzantine-resilient decentralized stochastic optimization with robust aggregation rules," *IEEE transactions on signal processing*, 2023.

[40] J. Peng, Z. Wu, Q. Ling, and T. Chen, "Byzantine-robust variance-reduced federated learning over distributed non-iid data," *Information Sciences*, vol. 616, pp. 367–391, 2022.