Secure Aggregation for Clustered Federated Learning

Hasin Us Sami and Başak Güler
Department of Electrical and Computer Engineering
University of California, Riverside, CA
hsami003@ucr.edu, bguler@ece.ucr.edu

2023 IEEE International Symposium on Information Theory (ISIT) | 978-1-6654-7554-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/ISIT54713.2023.10206964

***Abstract*—Clustered federated learning is a popular paradigm to tackle data heterogeneity in federated learning, by training personalized models for groups of users with similar data distributions. A critical challenge is to protect the privacy of individual user updates, as the latter can reveal extensive information about sensitive local datasets. To do so, a recent promising approach is information-theoretic secure aggregation, where parties learn the aggregate (sum) of user updates, but no further information is revealed about the individual updates. In this work, we present the first secure aggregation frameworks in the context of clustered federated learning, to learn the aggregate of user updates for any clustering of users, but without learning any information about the local updates or cluster identities. Our frameworks can achieve linear communication complexity under formal information-theoretic privacy guarantees, while providing key trade-offs between communication and computation complexity, adversary tolerance, and resilience to user dropouts.**

I. Introduction

Federated learning (FL) is a distributed learning framework to train machine learning models over the data stored and processed locally across a large number of wireless devices (users) [1]. Unlike traditional centralized training architectures, where all data is collected by a central party who performs training, FL keeps the data on device. Instead, each user updates the trained model locally on their local data, and then the local updates (e.g., gradients) are aggregated (often by a server) to form a global model. In doing so, users always keep the data on device, and send only the intermediate computations (e.g., local gradients).

Due to this *on-device* learning architecture (data never leaves the device, but only the local updates), FL has been highly popular in privacy-sensitive applications, such as healthcare. On the other hand, recent *gradient inversion attacks* have shown that the local updates sent by the users (such as gradients) can still reveal extensive information about the local datasets [2]–[4]. Secure aggregation (SA) protocols have been introduced to address this challenge, by revealing only the *sum of the local updates* to the server during training, while hiding the contents of individual updates sent from each user using information-theoretic or cryptographic tools [5]–[11]. In doing so, SA ensures that no further information is revealed beyond the sum of the local updates, preventing the server from associating the aggregated updates with any particular user. SA can further be combined with complementary privacy-preserving mechanisms such as differential privacy [12], [13] and can even benefit the latter [14], [15].

A major challenge of FL is the severe data heterogeneity across the users, which slows down training, and degrades model accuracy [16]. More importantly, training a single model (across the entire network) may disproportionately penalize the performance of underrepresented users [17]. Clustered FL is a recent approach to tackle this challenge by training multiple models, each adapted to a group of users with similar data distributions [18]–[23]. The training process alternates between clustering the users with respect to their data distributions, and training a distinct model within each cluster. For the latter, the server collects and aggregates the local updates (gradients) from the users assigned to each cluster, to update the model designated for that cluster. Several complementary approaches also explore addressing data heterogeneity by designing a personalized model for *each user* through fine-tuning or meta-learning [24]–[27]. In contrast, clustered FL targets *group-level personalization*, where the server maintains personalized models to serve *groups of users* with similar characteristics, while avoiding excessive memory and storage costs to handle a large number of models.

In this work, our goal is to develop an SA framework for clustered FL. A naive approach is to leverage conventional SA protocols to aggregate the local gradients of the users assigned to each cluster (independently from other clusters). On the other hand, doing so requires the server to learn the cluster identity of each user, which itself is highly sensitive information, revealing which users have similar data distributions [21]. An adversarial user can further infer sensitive information about the characteristics of honest users assigned to the same cluster, simply by leveraging the similarity between the distributions. Importantly, underrepresented users are the most vulnerable to this type of attack, due to the lack of a large number of users with similar data distributions, i.e., same cluster identity. Moreover, clusters may vary throughout the training, using which one may reveal the local gradients by comparing the aggregated updates received at different training rounds [28]. As such, here we ask the following question:

- *How can we enable SA for clustered FL, for the server to learn the aggregate of local gradients for each cluster, but without learning any information about the local*

This research was sponsored in part by the OUSD (R&E)/RT&L under Cooperative Agreement Number W911NF-20-2-0267, NSF CAREER Award CCF-2144927, UCR OASIS Award and UC Regents Faculty Fellowship.

gradients or cluster identities of individual users?

To address this challenge, in this work we propose the first SA frameworks for clustered FL. In all proposed frameworks, the server can perfectly recover the aggregate of local gradients for each cluster, but without learning any further information about the cluster identities or local gradients of the users. All proposed frameworks ensure strong information-theoretic privacy guarantees, while providing a trade-off between the communication and computation overhead, round complexity, and resilience to user dropouts (e.g., due to poor channel conditions). Our contributions can be summarized as follows:

- We propose the first study of SA in the context of clustered FL, to aggregate the local gradients from multiple clusters of users simultaneously, without learning any information about the cluster identities or local gradients.
- We introduce the first SA frameworks for clustered FL, and analyze the trade-offs between the communication/computation overhead, adversary tolerance, round complexity, and resilience to user dropouts.
- For all proposed frameworks, we demonstrate the formal information-theoretic privacy guarantees.

II. Problem Formulation

Clustered FL. We consider a clustered FL setting in a network of N users and a server. The local dataset $\mathcal{D}_i$ of user $i \in [N]$ is realized from one of K distributions denoted by $\mathcal{P}_1, \ldots, \mathcal{P}_K$. The goal is to train K models $\mathbf{w}_1, \ldots, \mathbf{w}_K$, where model $\mathbf{w}_k \in \mathbb{R}^d$ is trained to minimize the loss function,

$$F_k(\mathbf{w}_k) := \mathbb{E}_{\xi \sim \mathcal{P}_k}[f(\mathbf{w}_k, \xi)] \quad \forall k \in [K], \tag{1}$$

where ξ is a data sample realized from distribution $\mathcal{P}_k$ and $f(\mathbf{w}_k, \xi)$ denotes the stochastic loss function computed on the data sample ξ and model $\mathbf{w}_k$. Then, the optimal model for each of the K distributions is given by:

$$\mathbf{w}_k^* = \arg\min_{\mathbf{w}_k} F_k(\mathbf{w}_k) \quad \forall k \in [K] \tag{2}$$

To solve (1), clustered FL [18], [21] takes an iterative approach, that alternates between partitioning the users into K clusters with respect to the similarity of their local datasets, and training of K global models (one for each cluster). To do so, at each round t, the server broadcasts the current state of the K global models $\{\mathbf{w}_k(t)\}_{k\in[K]}$ to all users. Then, user $i \in [N]$ computes a local empirical loss,

$$f_i(\mathbf{w}_k(t)) \triangleq \frac{1}{|\mathcal{D}_i|} \sum_{\xi \in \mathcal{D}_i} f_i(\mathbf{w}_k(t), \xi) \tag{3}$$

for each model $\{\mathbf{w}_k(t)\}_{k\in[K]}$, and selects the cluster with the minimum loss,

$$c_i^{(t)} \triangleq \arg\min_{k\in[K]} f_i(\mathbf{w}_k(t)). \tag{4}$$

Next, user $i \in [N]$ computes a local gradient for the model of the selected cluster,

$$\mathbf{g}_i(t) \triangleq \nabla f_i(\mathbf{w}_{c_i^{(t)}}(t)) \tag{5}$$

and sends the local gradient from (5), along with the cluster assignment from (4), to the server. Then, the server updates

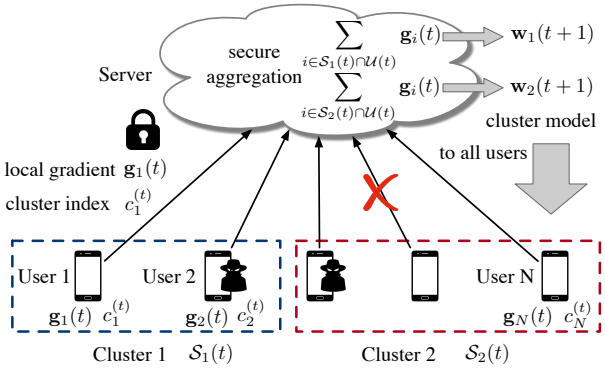

Fig. 1. **Secure aggregation for clustered FL.** At each training round, user i computes a local gradient $\mathbf{g}_i(t)$ for the selected cluster. The server learns the aggregate of local gradients $\sum_{i\in\mathcal{S}_k(t)\cap\mathcal{U}(t)} \mathbf{g}_i(t)$ for each cluster $k \in [K]$, *without learning which users belong to which cluster*, or the local gradients.

the global model for each cluster, by aggregating the local gradients received from users assigned to that cluster,

$$\mathbf{w}_k(t+1) = \mathbf{w}_k(t) - \frac{\eta}{N} \sum_{i\in\mathcal{S}_k(t)\cap\mathcal{U}(t)} \mathbf{g}_i(t) \quad \forall k \in [K], \tag{6}$$

where η is the learning rate, $\mathcal{S}_k(t) \triangleq \{i : c_i^{(t)} = k, i \in [N]\}$ denotes the set of users assigned to cluster k at round t. At each round of training, up to D out of N users may drop out or get delayed due to various reasons, such as poor channel conditions or low battery. Accordingly, $\mathcal{U}(t) \subseteq [N]$ denotes the set of surviving users at round t, who successfully send their local gradient $\mathbf{g}_i(t)$ to the server, where $|\mathcal{U}(t)| \geq N - D$.

Threat model. We consider an honest-but-curious adversary model (as is most common in secure aggregation), where adversaries follow the protocol, but try to reveal additional information about the local datasets of honest users from the messages exchanged. Out of N users, up to T are adversarial, who may collude with one another and/or the server.

Information-theoretic secure aggregation. Our goal is to enable the server to compute the sum of the local gradients $\sum_{i\in\mathcal{S}_k(t)\cap\mathcal{U}(t)} \mathbf{g}_i(t)$ for each cluster $k \in [K]$, to be able to update the model from (6) correctly, but without learning any further information about the local gradients or the cluster identities of the users, as illustrated in Fig. 1. Formally, this condition can be stated as,

$$I\Big(\{\mathbf{g}_i(t), c_i^{(t)}\}_{[N]\setminus\mathcal{T}}; \mathcal{M}_\mathcal{T} \Big| \Big\{\sum_{i\in\mathcal{S}_k(t)\cap\mathcal{U}(t)} \mathbf{g}_i(t)\Big\}_{k\in[K]}, \{\mathbf{g}_i(t), c_i^{(t)}\}_{i\in\mathcal{T}}, \mathcal{G}_\mathcal{T}\Big) = 0 \tag{7}$$

for all $\mathcal{T}$ such that $|\mathcal{T}| \leq T$, where $\mathcal{M}_\mathcal{T}$ denotes the collection of all messages received by the adversaries, and $\mathcal{G}_\mathcal{T}$ is the set of randomness generated by the adversaries during training.

We then ask the following question:

- How can the server learn the aggregate of local gradients from (6) for all K clusters, under the information-theoretic privacy guarantees from (7)?

To address this challenge, in this work we propose three secure aggregation protocols, with different trade-offs in terms of the

communication/computation overhead, round complexity, and dropout resilience, which will be detailed in Section IV.

Similar to [5], [7], [9], our frameworks are bound to finite field computations, where each user converts their local gradient $\mathbf{g}_i(t) \in \mathbb{R}^d$ from the real domain to a finite field $\mathbb{F}_q$ of integers modulo a large prime q. We refer to [7], [29], [30] for the details of this conversion. In the sequel, we use $\overline{\mathbf{g}}_i(t) \in \mathbb{F}_q^d$ to denote the finite field representation of $\mathbf{g}_i(t)$. We next present the details of the proposed frameworks.

III. Clustered Secure Aggregation

We next present three approaches to secure aggregation for clustered FL. For notational clarity, we omit the time index t.

A. Clustered Secret Gradient Sharing (Clustered-GS)

In this framework, users first agree on N distinct public parameters $\alpha_1, \ldots, \alpha_N$ from $\mathbb{F}_q$. Then, each user $i \in [N]$ partitions its local gradient $\overline{\mathbf{g}}_i$ into L equal-sized shards,

$$\overline{\mathbf{g}}_i = \begin{bmatrix} \overline{\mathbf{g}}_{i1}^{\mathrm{T}} & \cdots & \overline{\mathbf{g}}_{iL}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}, \tag{8}$$

and generates T independent (uniformly) random vectors $\mathbf{v}_{i1}, \ldots, \mathbf{v}_{iT} \in \mathbb{F}_q^{\frac{d}{L}}$ to form a degree $KL+T-1$ polynomial,

$$f_i(\alpha) \triangleq \sum_{l=1}^{L} \alpha^{(c_i-1)L+l-1} \overline{\mathbf{g}}_{il} + \sum_{l=1}^{T} \alpha^{KL+l-1} \mathbf{v}_{il}, \tag{9}$$

and sends to each user $j \in [N]$ a *coded gradient*,

$$\widetilde{\mathbf{g}}_{ij} \triangleq f_i(\alpha_j). \tag{10}$$

To recover the *aggregate* of the local gradients, the server requests the aggregate of the coded gradients,

$$\widetilde{\mathbf{g}}_i \triangleq \sum_{j \in \mathcal{U}} \widetilde{\mathbf{g}}_{ji} \tag{11}$$

from each user $i \in [N]$. The computations from (11) can be viewed as evaluations of a degree $KL+T-1$ polynomial,

$$f(\alpha) \triangleq \sum_{j \in \mathcal{U}} f_j(\alpha) \tag{12}$$

at an interpolation point $\alpha = \alpha_i$, where $\widetilde{\mathbf{g}}_i = f(\alpha_i)$. Since any polynomial f of degree $\deg f$ can be reconstructed from at least $\deg f + 1$ evaluation points, the server can reconstruct for each cluster $k \in [K]$, the aggregate of the local gradients,

$$\sum_{j \in \mathcal{S}_k \cap \mathcal{U}} \overline{\mathbf{g}}_j = \begin{bmatrix} \sum_{j \in \mathcal{S}_k \cap \mathcal{U}} \overline{\mathbf{g}}_{j1}^{\mathrm{T}} & \cdots & \sum_{j \in \mathcal{S}_k \cap \mathcal{U}} \overline{\mathbf{g}}_{jL}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}, \tag{13}$$

using polynomial interpolation, upon receiving (11) from any set of at least $KL+T$ users. Parameter L controls a trade-off between communication and dropout resilience. The communication overhead is $O(\frac{dN}{L})$ per user, inversely proportional to L, whereas the maximum number of user dropouts tolerated is $D \leq N-(KL+T)$, which increases by using a smaller L.

B. Clustered Masked Gradient Aggregation (Clustered-MA)

Our second framework builds on an online-offline trade-off, by dividing the communication into online (data-dependent) and offline (data-agnostic) phases. The former depends on the datasets, hence can only be carried out after training starts. The latter is independent from data (such as randomness generation), and can be carried out in advance when the network load is low. The key intuition is to transfer the intensive communication overhead incurred by large N to the offline phase, by increasing the number of communication rounds. As demonstrated next, one can achieve an online communication overhead of $O(dK)$ (independent from the number of users) while keeping the offline overhead as $O(\frac{dN}{L})$. We next describe the details of each phase.

Offline. In the offline phase, users first agree on N distinct public parameters $\alpha_1, \ldots, \alpha_N$ from $\mathbb{F}_q$. User $i \in [N]$ then generates K random masks $\{\mathbf{r}_{ik}\}_{k \in [K]}$ of size d uniformly at random from $\mathbb{F}_q$, each partitioned into L shards,

$$\mathbf{r}_{ik} = \begin{bmatrix} \mathbf{r}_{ik1}^{\mathrm{T}} & \cdots & \mathbf{r}_{ikL}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \tag{14}$$

Next, user i constructs a polynomial of degree $KL+T-1$,

$$f_i(\alpha) \triangleq \sum_{k=1}^{K} \sum_{l=1}^{L} \alpha^{(k-1)L+l-1} \mathbf{r}_{ikl} + \sum_{l=1}^{T} \alpha^{KL+l-1} \mathbf{v}_{il}, \tag{15}$$

where $\mathbf{v}_{il} \in \mathbb{F}_q^{\frac{d}{L}}$ are generated uniformly at random for all $l \in [T]$, and sends an *encoded mask*,

$$\widetilde{\mathbf{r}}_{ij} \triangleq f_i(\alpha_j) \tag{16}$$

to each user $j \in [N]$. The masks $\{\mathbf{r}_{ik}\}_{k \in [K]}$ will be used to hide the true content of the local gradients in the online phase, whereas the random vectors $\{\mathbf{v}_{il}\}_{l \in [T]}$ will hide the *true value of the masks* against up to T adversaries.

Online. In the online phase, each user $i \in [N]$ sends to the server a *masked gradient*,

$$\mathbf{x}_{ik} \triangleq \begin{cases} \overline{\mathbf{g}}_i + \mathbf{r}_{ik} & \text{if } i \in \mathcal{S}_k \\ \mathbf{r}_{ik} & \text{otherwise} \end{cases}, \tag{17}$$

for each cluster $k \in [K]$. The server then aggregates the received masked gradients $\sum_{i \in \mathcal{U}} \mathbf{x}_{ik}$ for each cluster $k \in [K]$. On the other hand, to recover the aggregate of the *true gradients* $\sum_{i \in \mathcal{S}_k \cap \mathcal{U}} \overline{\mathbf{g}}_i$, the server has to remove the random masks $\sum_{i \in \mathcal{U}} \mathbf{r}_{ik}$ from the masked gradients $\sum_{i \in \mathcal{U}} \mathbf{x}_{ik}$. To do so, the server requests the aggregate of the coded masks,

$$\widetilde{\mathbf{r}}_i \triangleq \sum_{j \in \mathcal{U}} \widetilde{\mathbf{r}}_{ji} \tag{18}$$

from each user $i \in [N]$. The computations from (18) can be viewed as evaluations of a degree $KL+T-1$ polynomial,

$$f(\alpha) \triangleq \sum_{j \in \mathcal{U}} f_j(\alpha) \tag{19}$$

at an interpolation point $\alpha = \alpha_i$, where $\widetilde{\mathbf{r}}_i = f(\alpha_i)$. Hence, upon receiving the evaluations (18) from any set $\mathcal{I}$ of at least $|\mathcal{I}| = KL+T$ users, the server can reconstruct the aggregate of the random masks,

$$\sum_{i \in \mathcal{U}} \mathbf{r}_{ik} = \begin{bmatrix} \sum_{i \in \mathcal{U}} \mathbf{r}_{ik1}^{\mathrm{T}} \cdots \sum_{i \in \mathcal{U}} \mathbf{r}_{ikL}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \text{ for } k \in [K] \tag{20}$$

via polynomial interpolation. Then, the server can recover the aggregate of *true gradients for each cluster*, by removing the (aggregated) random masks in (20) from the masked gradients,

$$\sum_{i \in \mathcal{S}_k \cap \mathcal{U}} \overline{\mathbf{g}}_i = \sum_{i \in \mathcal{U}} \mathbf{x}_{ik} - \sum_{i \in \mathcal{U}} \mathbf{r}_{ik} \text{ for } k \in [K] \tag{21}$$

Clustered-MA achieves a per-user online communication overhead of $O(dK)$, by offloading the $O(\frac{dN}{L})$ (online) overhead of Clustered-GS to the offline phase, while providing equal resilience against user dropouts $D \leq N-(KL+T)$. On the other hand, when the number of clusters K is large, as is often the case in highly heterogeneous networks, the $O(dK)$ overhead is still significant. Our next framework overcomes this challenge by reducing the online overhead to $O(d+K)$, by trading-off communication with tolerance to user dropouts.

C. Secure Aggregation with Masked Clusters (Clustered-SA)

Our last framework also builds on an online/offline trade-off, where we offload the communication intensive operations to the latter. On the other hand, instead of aggregating the masked gradients for each cluster, each user now sends a *one-shot masked gradient* along with a *masked cluster identity*. The two are then combined with encoded random masks generated in the offline phase, in a way that the server can correctly recover the sum of the true gradients for each cluster, without learning any information about their true value. We next describe the details of each phase.

Offline. In this phase, users generate three Lagrange polynomials, where the first two are used to mask the local gradients and cluster identities in the online phase, while the third one is used to ensure the information theoretic privacy during the final reconstruction of the sum of local gradients. Initially, users agree on $2(N+KL+T)-1$ distinct public parameters $\{\alpha_i\}_{i\in[N]}$, $\{\beta_m\}_{m\in[KL+T]}$, $\{\theta_m\}_{m\in\{KL+1,\ldots,2(KL+T-1)+1\}}$, $\{\lambda_m\}_{m\in[N-T]}$ from $\mathbb{F}_q$. Next, each user $i\in[N]$ generates a random mask,

$$\mathbf{r}_i \triangleq \begin{bmatrix}\mathbf{r}_{i1}^{\mathrm{T}} & \cdots & \mathbf{r}_{iL}^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}}, \tag{22}$$

where $\mathbf{r}_{il}\in\mathbb{F}_q^{\frac{d}{L}}$ for all $l\in[L]$ are generated uniformly at random (and independently from other elements), and then forms a Lagrange polynomial of degree $KL+T-1$,

$$f_i(\alpha) \triangleq \sum_{l\in[L]}\mathbf{r}_{il}\sum_{k\in[K]}\prod_{\substack{m\in[KL+T]\\ \setminus\{(k-1)L+l\}}}\frac{\alpha-\beta_m}{\beta_{(k-1)L+l}-\beta_m} + \sum_{l=KL+1}^{KL+T}\mathbf{v}_{il}\prod_{m\in[KL+T]\setminus\{l\}}\frac{\alpha-\beta_m}{\beta_l-\beta_m}, \tag{23}$$

where $\mathbf{v}_{il}\in\mathbb{F}_q^{\frac{d}{L}}$ are uniformly random vectors for all $l\in\{KL+1,\ldots,KL+T\}$. Then, user i sends an *encoded mask*,

$$\widetilde{\mathbf{r}}_{ij} \triangleq f_i(\alpha_j) \tag{24}$$

to each user $j\in[N]$. In addition, user i generates K random masks $z_{i1},\ldots,z_{iK}\in\mathbb{F}_q$ (uniformly at random), forms a Lagrange polynomial of degree $KL+T-1$,

$$h_i(\alpha) \triangleq \sum_{k\in[K]}z_{ik}\sum_{l\in[L]}\prod_{\substack{m\in[KL+T]\\ \setminus\{(k-1)L+l\}}}\frac{\alpha-\beta_m}{\beta_{(k-1)L+l}-\beta_m} + \sum_{l=KL+1}^{KL+T}u_{il}\prod_{m\in[KL+T]\setminus\{l\}}\frac{\alpha-\beta_m}{\beta_l-\beta_m}, \tag{25}$$

where $u_{il}\in\mathbb{F}_q$ are generated uniformly random for $l\in\{KL+1,\ldots,KL+T\}$, and sends an encoded mask,

$$\widetilde{z}_{ij} \triangleq h_i(\alpha_j) \tag{26}$$

to user $j\in[N]$. Finally, user i generates a third Lagrange polynomial of degree $2(KL+T-1)$:

$$v_i(\alpha) \triangleq \sum_{l=KL+1}^{2(KL+T-1)+1}\mathbf{n}_{il}\prod_{m\in[2(KL+T-1)+1]\setminus\{l\}}\frac{\alpha-\theta_m}{\theta_l-\theta_m} \tag{27}$$

where $\theta_l\triangleq\beta_l$ for $l\in[KL]$, and $\mathbf{n}_{il}$ is a uniformly random vector of size $\frac{d}{L(N-T)}$ for $l\in\{KL+1,\ldots,2(KL+T-1)+1\}$. User i then sends an encoded vector,

$$\widetilde{\mathbf{n}}_{ij} \triangleq v_i(\alpha_j) \tag{28}$$

to user $j\in[N]$. After receiving $\{\widetilde{\mathbf{n}}_{ji}\}_{j\in[N]}$, user i computes:

$$\widetilde{\mathbf{n}}_i \triangleq \begin{bmatrix}\sum_{j\in[N]}\lambda_1^{j-1}\widetilde{\mathbf{n}}_{ji}^{\mathrm{T}} & \cdots & \sum_{j\in[N]}\lambda_{N-T}^{j-1}\widetilde{\mathbf{n}}_{ji}^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}} \tag{29}$$

where (29) can be viewed as evaluations of a Lagrange polynomial $v(\alpha)$ of degree $2(KL+T-1)$,

$$v(\alpha) \triangleq \sum_{l=KL+1}^{2(KL+T-1)+1}\mathbf{n}_l\prod_{m\in[2(KL+T-1)+1]\setminus\{l\}}\frac{\alpha-\theta_m}{\theta_l-\theta_m} \tag{30}$$

such that $\widetilde{\mathbf{n}}_i=v(\alpha_i)$, whereas $v(\theta_l)=\mathbf{0}$ for $l\in[KL]$, and

$$v(\theta_l)=\mathbf{n}_l\triangleq\begin{bmatrix}\sum_{j\in[N]}\lambda_1^{j-1}\mathbf{n}_{jl}^{\mathrm{T}} & \cdots & \sum_{j\in[N]}\lambda_{N-T}^{j-1}\mathbf{n}_{jl}^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}} \tag{31}$$

for all $l\in\{KL+1,\ldots,2(KL+T-1)+1\}$.

Online. In the online phase, each user $i\in[N]$ initially broadcasts a masked local gradient,

$$\mathbf{x}_i \triangleq \overline{\mathbf{g}}_i-\mathbf{r}_i \tag{32}$$

along with a masked cluster indicator for each cluster k,

$$y_{ik} \triangleq b_{ik}-z_{ik}, \tag{33}$$

where $k\in[K]$, and

$$b_{ik} \triangleq \begin{cases}1 & \text{if } i\in\mathcal{S}_k\\ 0 & \text{otherwise}\end{cases} \tag{34}$$

is a binary indicator variable representing whether user i is assigned to cluster $k\in[K]$. To reconstruct the aggregate of the local gradients, the server requests from all users $i\in[N]$,

$$\widetilde{\mathbf{a}}_i \triangleq \sum_{j\in\mathcal{U}}\Big(\sum_{k\in[K]}y_{jk}\sum_{l\in[L]}\prod_{\substack{m\in[KL+T]\\ \setminus\{(k-1)L+l\}}}\frac{\alpha_i-\beta_m}{\beta_{(k-1)L+l}-\beta_m}+\widetilde{z}_{ji}\Big) \times\Big(\sum_{l\in[L]}\mathbf{x}_{jl}\sum_{k\in[K]}\prod_{\substack{m\in[KL+T]\\ \setminus\{(k-1)L+l\}}}\frac{\alpha_i-\beta_m}{\beta_{(k-1)L+l}-\beta_m}+\widetilde{\mathbf{r}}_{ji}\Big)-\widetilde{\mathbf{n}}_i \tag{35}$$

where $\mathbf{x}_j=[\mathbf{x}_{j1}^{\mathrm{T}}\cdots\mathbf{x}_{jL}^{\mathrm{T}}]^{\mathrm{T}}$ is partitioned into L shards of size $\frac{d}{L}$. Note that the computations from (35) can be viewed as evaluations of a degree $2(KL+T-1)$ polynomial,

$$f(\alpha) \triangleq \Big(\sum_{j\in\mathcal{U}}\phi_j(\alpha)\psi_j(\alpha)\Big)-v(\alpha) \tag{36}$$

TABLE I
COMPARISON OF COMMUNICATION COMPLEXITY (PER-USER) AND DROPOUT RESILIENCE (MAXIMUM NUMBER OF USER DROPOUTS).

Communication complexity			Dropout resilience
Clustered-GS	online	$O(dN/L)$	$D \leq N-(KL+T)$
	offline	–	
Clustered-MA	online	$O(dK)$	$D \leq N-(KL+T)$
	offline	$O(dN/L)$	
Clustered-SA	online	$O(d+K)$	$D \leq N-2(KL+T)+1$
	offline	$O(dN/L)$	

where $\widetilde{\mathbf{a}}_i = f(\alpha_i)$, such that

$$\phi_j(\alpha) \triangleq \sum_{k\in[K]} b_{jk} \sum_{l\in[L]} \prod_{\substack{m\in[KL+T]\\ \setminus\{(k-1)L+l\}}} \frac{\alpha-\beta_m}{\beta_{(k-1)L+l}-\beta_m} + \sum_{l=KL+1}^{KL+T} u_{jl} \prod_{m\in[KL+T]\setminus\{l\}} \frac{\alpha-\beta_m}{\beta_l-\beta_m}, \quad (37)$$

$$\psi_j(\alpha) \triangleq \sum_{l\in[L]} \overline{\mathbf{g}}_{jl} \sum_{k\in[K]} \prod_{\substack{m\in[KL+T]\\ \setminus\{(k-1)L+l\}}} \frac{\alpha-\beta_m}{\beta_{(k-1)L+l}-\beta_m} + \sum_{l=KL+1}^{KL+T} \mathbf{v}_{jl} \prod_{m\in[KL+T]\setminus\{l\}} \frac{\alpha-\beta_m}{\beta_l-\beta_m} \quad (38)$$

where $\overline{\mathbf{g}}_j = \begin{bmatrix}\overline{\mathbf{g}}_{j1}^{\mathrm{T}} & \cdots & \overline{\mathbf{g}}_{jL}^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}}$ denotes the local gradient of user j partitioned into L equal-sized shards. Since $f(\beta_{(k-1)L+l}) = \sum_{j\in\mathcal{U}} b_{jk}\overline{\mathbf{g}}_{jl} = \sum_{j\in\mathcal{S}_k\cap\mathcal{U}} \overline{\mathbf{g}}_{jl}$ corresponds to the *true sum of the local gradients* for each cluster $k\in[K]$ and shard $l\in[L]$, after receiving the evaluations (35) from a set of at least $2(KL+T-1)+1$ users, the server can reconstruct $f(\alpha)$ through polynomial interpolation, and recover the sum,

$$\sum_{j\in\mathcal{S}_k\cap\mathcal{U}} \overline{\mathbf{g}}_j = \begin{bmatrix} f(\beta_{(k-1)L+1})^{\mathrm{T}} & \cdots & f(\beta_{(k-1)L+L})^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}} \quad (39)$$

of the local gradients for each cluster $k\in[K]$.

Remark 1. *Clustered-SA reduces the per-user online communication overhead to $O(d+K)$ (down from the $O(Kd)$ overhead of Clustered-MA), while keeping the offline overhead the same. This is achieved by a trade-off between communication overhead and dropout resilience; Clustered-SA slashes the online communication complexity, while requiring a larger number of surviving users for correct recovery of the aggregated gradients. A comparison of the communication complexity and dropout resilience of the three frameworks are demonstrated in Table I, which will be detailed in Section IV.*

Remark 2. *The key intuition behind the polynomial $v(\alpha)$ in* (36) *is to ensure privacy during the* reconstruction *of the final outcomes by the server. Since $v(\beta_{(k-1)L+l}) = 0$ for all $k\in[K], l\in[L]$, in principle, the final outcomes in* (39) *can be recovered by interpolating the polynomial $\sum_{j\in\mathcal{U}}\phi_j(\alpha)\psi_j(\alpha)$ directly, by collecting the evaluations $\sum_{j\in\mathcal{U}}\phi_j(\alpha_i)\psi_j(\alpha_i)$ from the users, however, additional information may be leaked (beyond the desired outcomes) from the intermediate polynomial coefficients. The masking with $\widetilde{\mathbf{n}}_i = v(\alpha_i)$ prevents such information leakage, as will be demonstrated in Theorem 4.*

IV. THEORETICAL ANALYSIS

We first present the per-user communication/computation complexity, adversary robustness, and resilience to user dropouts. The dropout resilience of a given framework is quantified by the *recovery threshold*, defined as the minimum number of surviving users required for correct recovery of the aggregate of local gradients.

Theorem 1. *Clustered-GS has a per-user communication complexity of $O(\frac{dN}{L})$, per-user computation complexity of $O(\frac{dN}{L}\log^2(KL+T)\log\log(KL+T))$, and a recovery threshold of $N-D\geq KL+T$.*

Theorem 2. *Clustered-MA has a per-user communication complexity of $O(dK)$ online and $O(\frac{dN}{L})$ offline, per-user computation complexity of $O(N\frac{d}{L})$ online and $O(\frac{dN}{L}\log^2(KL+T)\log\log(KL+T))$ offline, and a recovery threshold of $N-D\geq KL+T$.*

Theorem 3. *Clustered-SA has a per-user communication complexity of $O(d+K)$ online and $O(\frac{dN}{L})$ offline, per-user computational complexity of $O(N(K+d))$ online and $O(\frac{dN}{L}\log^2(KL+T)\log\log(KL+T))$ offline, and a recovery threshold of $N-D\geq 2(KL+T)-1$.*

Remark 3. *The three frameworks provide a trade-off among the online/offline communication complexity, computation cost, and the recovery threshold. Clustered-MA reduces the online communication overhead of Clustered-GS by introducing an additional offline phase. Clustered-SA reduces the online communication by a factor of K compared to Clustered-MA, while increasing the recovery threshold by a constant factor.*

We next present the privacy guarantees from (7).

Theorem 4. *(Information-theoretic privacy) Clustered-GS, Clustered-MA, and Clustered-SA all guarantee the information-theoretic privacy of honest users from* (7) *against any set $\mathcal{T}$ of up to $|\mathcal{T}|\leq T$ adversaries,*

$$I\Big(\{\overline{\mathbf{g}}_i, c_i\}_{[N]\setminus\mathcal{T}}; \mathcal{M}_{\mathcal{T}} \Big| \Big\{\sum_{i\in\mathcal{S}_k\cap\mathcal{U}} \overline{\mathbf{g}}_i\Big\}_{k\in[K]}, \{\overline{\mathbf{g}}_i, c_i\}_{i\in\mathcal{T}}, \mathcal{G}_{\mathcal{T}}\Big) = 0 \quad (40)$$

where $\mathcal{M}_{\mathcal{T}}$ denotes the collection of all messages received by the server and adversarial users, and $\mathcal{G}_{\mathcal{T}}$ is the set of randomness generated by the adversarial users during training.

Proof. (Sketch) The proof follows from comparing the entropy of the masked/encoded gradients observed by the adversaries with respect to the entropy of the uniform distribution. □

V. CONCLUSION

This work is the first study of secure aggregation for clustered federated learning, to aggregate the local gradients for any cluster of users without learning the local gradients or cluster identities. Our frameworks achieve a linear online communication overhead, while ensuring formal guarantees for information-theoretic privacy and resilience to user dropouts.

REFERENCES

[1] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Int. Conf. on Artificial Int. and Stat. (AISTATS)*, 2017, pp. 1273–1282.

[2] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, 2015, pp. 1322–1333.

[3] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2019, pp. 14 747–14 756.

[4] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients - how easy is it to break privacy in federated learning?" in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[5] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1175–1191.

[6] J. H. Bell, K. A. Bonawitz, A. Gascón, T. Lepoint, and M. Raykova, "Secure single-server aggregation with (poly) logarithmic overhead," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, 2020, pp. 1253–1269.

[7] J. So, B. Güler, and A. S. Avestimehr, "Turbo-aggregate: Breaking the quadratic aggregation barrier in secure federated learning," *IEEE Journal on Selected Areas in Information Theory*, 2021.

[8] Y. Zhao and H. Sun, "Information theoretic secure aggregation with user dropouts," in *IEEE International Symposium on Information Theory, ISIT'21*, 2021.

[9] J. So, C.-S. Yang, S. Li, Q. Yu, R. E Ali, B. Guler, and S. Avestimehr, "Lightsecagg: a lightweight and versatile design for secure aggregation in federated learning," *Proceedings of Machine Learning and Systems*, vol. 4, pp. 694–720, 2022.

[10] T. Jahani-Nezhad, M. A. Maddah-Ali, S. Li, and G. Caire, "Swiftagg: Communication-efficient and dropout-resistant secure aggregation for federated learning with worst-case security guarantees," in *IEEE International Symposium on Information Theory, ISIT*. IEEE, 2022, pp. 103–108.

[11] ——, "Swiftagg+: Achieving asymptotically optimal communication load in secure aggregation for federated learning," *CoRR*, vol. abs/2203.13060, 2022.

[12] C. Dwork, F. McSherry, K. Nissim, and A. D. Smith, "Calibrating noise to sensitivity in private data analysis," *J. Priv. Confidentiality*, vol. 7, no. 3, pp. 17–51, 2016.

[13] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 308–318.

[14] P. Kairouz, Z. Liu, and T. Steinke, "The distributed discrete gaussian mechanism for federated learning with secure aggregation," in *International Conference on Machine Learning*. PMLR, 2021, pp. 5201–5212.

[15] W. Chen, C. A. Choquette-Choo, P. Kairouz, and A. T. Suresh, "The fundamental price of secure aggregation in differentially private federated learning," in *International Conference on Machine Learning, ICML*, vol. 162. PMLR, 2022, pp. 3056–3089.

[16] P. Kairouz and H. B. McMahan, "Advances and open problems in federated learning," *Foundations and Trends in Machine Learning*, vol. 14, no. 1, 2021.

[17] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning," in *International Conference on Learning Representations*, 2020.

[18] Y. Mansour, M. Mohri, J. Ro, and A. T. Suresh, "Three approaches for personalization with applications to federated learning," *CoRR*, vol. abs/2002.10619, 2020.

[19] C. Briggs, Z. Fan, and P. Andras, "Federated learning with hierarchical clustering of local updates to improve training on non-iid data," in *2020 International Joint Conference on Neural Networks (IJCNN)*, 2020, pp. 1–9.

[20] F. Sattler, K.-R. Müller, and W. Samek, "Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints," *IEEE Trans. on Neural Networks and Learning Systems*, vol. 32, no. 8, pp. 3710–3722, 2021.

[21] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, "An efficient framework for clustered federated learning," *IEEE Transactions on Information Theory*, vol. 68, no. 12, pp. 8076–8091, 2022.

[22] M. Nafea, E. Shin, and A. Yener, "Proportional fair clustered federated learning," in *2022 IEEE International Symposium on Information Theory (ISIT)*, 2022, pp. 2022–2027.

[23] Y. Ruan and C. Joe-Wong, "Fedsoft: Soft clustered federated learning with proximal local updating," in *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI*. AAAI Press, 2022, pp. 8124–8131.

[24] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

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

[26] C. T. Dinh, N. H. Tran, and T. D. Nguyen, "Personalized federated learning with moreau envelopes," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

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

[28] J. So, R. E. Ali, B. Guler, J. Jiao, and S. Avestimehr, "Securing secure aggregation: Mitigating multi-round privacy leakage in federated learning," *CoRR*, vol. abs/2106.03328, 2021.

[29] J. So, B. Güler, and S. Avestimehr, "A scalable approach for privacy-preserving collaborative machine learning," in *Advances in Neural Information Processing Systems: Annual Conference on Neural Information Processing Systems, NeurIPS*, Dec. 2020.

[30] J. So, B. Güler, and A. S. Avestimehr, "Byzantine-resilient secure federated learning," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 7, pp. 2168–2181, 2021.