Private Set Union Based Approach to Enable Private Federated Submodel Learning

Zhusheng Wang Sennur Ulukus
Department of Electrical and Computer Engineering
University of Maryland, College Park, MD 20742
zhusheng@umd.edu *ulukus@umd.edu*

***Abstract*—We consider the federated submodel learning (FSL) problem and propose an approach where clients are able to update the central model information theoretically privately. Our approach is based on private set union (PSU), which is further based on multi-message symmetric private information retrieval (MM-SPIR). With our scheme, the server does not learn anything further than the subset of submodels updated by the clients: the server does not know which client updated which submodel(s), or anything about the local client data. In comparison to the state-of-the-art private FSL schemes of Jia-Jafar and Vithana-Ulukus, our scheme does not require noisy storage of the model at the databases; and in comparison to the secure aggregation scheme of Zhao-Sun, our scheme incorporates the creation of the required client-side common randomness via random symmetric private information retrieval (RSPIR) and one-time pads. Our system is initialized with a replicated storage of submodels and a sufficient amount of common randomness in two databases at the server-side. The protocol starts with a common randomness generation (CRG) where the two databases establish common randomness at the client-side (FSL-CRG phase). Next, the clients utilize the established client-side common randomness to have the server determine privately the union of indices of submodels to be updated collectively by the clients (FSL-PSU phase). Then, the two databases broadcast the current versions of the submodels in the set union to clients. The clients update the submodels based on their local data. Finally, the clients use a variation of FSL-PSU to write the updates back to the databases privately (FSL-write phase). Our proposed private FSL scheme achieves low communication cost, and is also robust against client drop-outs, client late-arrivals, and database drop-outs.**

I. Introduction

In federated learning (FL), multiple isolated clients collaboratively perform a learning task while protecting the privacy of their stored local data against the global server [1], [2]. An intuitive way for FL to achieve privacy is to use a secure aggregation protocol such that no individual client's update can be inspected by the global server [3]. As a stand-alone topic, secure aggregation has been a continuously active topic in the computer science literature, see [3]–[6]. Recently, information theoretically secure aggregation schemes towards achieving optimal communication cost have been proposed for various common randomness distribution settings among the clients, see [7]–[9]. However, in these papers, the communication costs of input and common randomness are considered separately, without explicitly stating the common randomness generation and allocation in the concrete realization.

This work was supported by ARO Grant W911NF2010142.

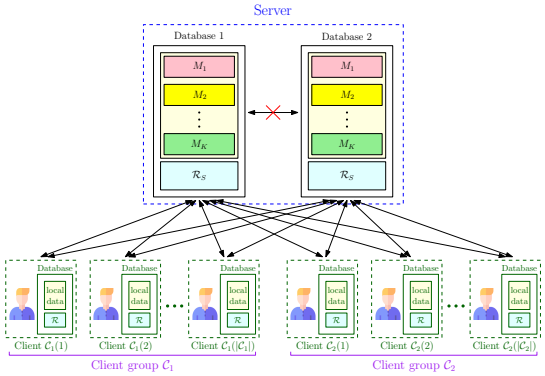


Fig. 1: Distributed federated submodel learning system model.

Recently, a new framework called federated submodel learning (FSL) has been proposed to further reduce the communication and computation overhead at both server and client sides [10]. In the FSL framework, the full learning model stored in the server is divided into multiple submodels based on their data characteristics; see the upper part of Fig. 1. Instead of accessing and updating the full model as in conventional FL, each selected client downloads only the needed submodel(s) from the server and then uploads the corresponding submodel updates based on the local data type in FSL. As pointed out by [10], there are two fundamental problems in FSL: One is how can each client download its desired submodels from the curious server without revealing the indices of these submodels to the server. The other is how can each client update these desired submodels still without revealing the indices or the content of the updated submodels to the curious server. The first one is a *private read* problem, and the second one is a *private write* problem.

In a computationally secure sense, reference [10] proposes a weak-privacy approach where each client trains only part of its desired submodels according to the inaccurate union result of desired submodel indices from all the selected clients. This idea follows from secure aggregation, but update efficiency of the clients is sacrificed. In an information theoretically secure sense, a strong-privacy FSL approach is introduced in [11] based on cross subspace alignment [12]. In this approach, only one client who is interested in a specific submodel participates in one round of the FSL process and databases (with noisy model storage) are completely unable to tell which submodel has been updated and what the updated value is. Concurrently,

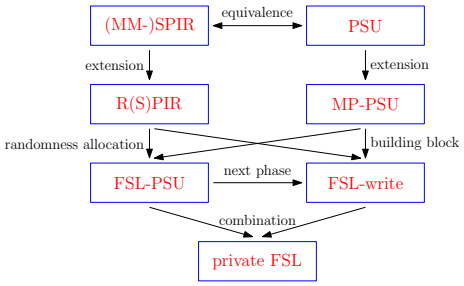


Fig. 2: Techniques used, their relationships, and the roadmap of the development of the private FSL in this paper.

an improved scheme in terms of communication cost efficiency is given in [13], and extended to the case of sparsified updates which further reduces the communication cost [14].

In this paper, we propose a new FSL scheme that retains the main advantages of the above-mentioned two approaches with a privacy protection level that is in between. First, the server securely calculates the clients' desired submodel index union. This is well-known as the private set union (PSU) problem and referred to as FSL-PSU phase. Then, the server securely aggregates clients' generated updates in the calculated set union. This is well-known as the secure aggregation problem and referred to as FSL-write phase. In both phases, the server can only learn the ultimate result, without knowing which client has made which contribution to the ultimate result.

In the field of cryptography, private information retrieval (PIR) and symmetric PIR (SPIR) are both fundamental problems [15], [16]. Following the seminal paper that focuses on the information theoretic capacity of multi-database PIR [17], PIR and SPIR have attracted a tremendous amount of attention in the field of information theory recently, e.g., [12], [18]–[37]. As a non-trivial variation of SPIR, in multi-message SPIR (MM-SPIR), the user wishes to retrieve multiple messages at a time [37]. The paper [37] also establishes the equivalence between MM-SPIR and private set intersection (PSI). Note that the constraints in PSI and PSU are analogous. We also establish the equivalence between PSU and MM-SPIR, and extend PSU to multi-party PSU; see the details in [38, Sections 2.1-2.2]. Similar to typical PIR/SPIR formulations, we consider the simplest setting where the FSL server has two databases. Our scheme indeed works for any number of databases after minor modifications. In Fig. 2, we show the techniques used, their relationships, and the roadmap of the development in this paper. The classical information theoretic SPIR serves as a starting point to formulate our new FSL scheme.

In this paper, we propose a new scheme for private distributed FSL primarily through unifying FSL-PSU and FSL-write in the same framework. Due to the long duration of FSL process, it is possible for some clients to drop out. Thus, we design our scheme in such a way that even if some clients lose their connection to the server, our scheme continues to work normally. It is also possible that some clients' generated answers arrive at their associated databases late and the corresponding databases make the wrong judgement that the clients have dropped out. Our scheme is designed such that these late answers do not leak any additional information about these late clients to the databases. Moreover, our scheme continues to work normally even when some of the databases become inactive, especially when the total number of databases is large enough. Finally, our FSL scheme can be run iteratively in multiple rounds until a predefined termination criterion is met.

II. Private Distributed FSL: Problem Formulation

We consider a distributed FSL problem with one server that contains $N = 2$ non-colluding and replicated databases, and C clients that are selected by the server to participate in one round of the FSL process; see Fig. 1. Every client establishes a direct secure and authenticated communication channel with both databases and our scheme only relies on these client-database channels. The full model for learning comprises K submodels, each one of which consisting of L i.i.d. symbols that are uniformly selected from a finite field $\mathbb{F}_q$. So, each database contains the full model $M_{[K]} = \{M_k : k \in [K]\}$[1]. The two databases also share some server-side common randomness $\mathcal{R}_S$ that is unknown to the clients. Each selected client is interested in updating one or more submodels according to its local data. Specifically, for $i \in [C]$, the ith client wishes to update the submodels whose index set is denoted by the random variable $\Gamma^{\langle i \rangle}$, whose realization is denoted by $\gamma^{\langle i \rangle}$. For $i \in [C]$, the random variable $Y^{\langle i \rangle} = \{Y_1^{\langle i \rangle}, \ldots, Y_K^{\langle i \rangle}\}$ is used to denote the corresponding incidence vector of $\Gamma^{\langle i \rangle}$ after mapping to the alphabet as in [37], [39].

We formulate our FSL process following the seminal FSL work in [10]. In the FSL-PSU phase, each individual database needs to calculate the union of the selected clients' desired submodel index sets $\Gamma^{\langle 1 \rangle} \cup \cdots \cup \Gamma^{\langle C \rangle}$ denoted by Γ. Due to the constraint that the two databases cannot communicate with each other directly, our solution is to use randomly selected alive clients as intermediators to route the information received by the two databases rather than to enforce each client to send the same answer to both databases. The main objective of this new approach is to reduce the total communication cost and the needed communication time. Thus, we separate C clients into two groups $\mathcal{C}_1$ and $\mathcal{C}_2$. A potential separation is to rely on each client's communication channel bandwidth (or quality) with the two databases; see Figs. 1 and 3 for depictions.

After completing a round of the FSL training, the submodels $M_\Gamma = \{M_k : k \in \Gamma\}$ are jointly updated by the participating clients while the remaining submodels are not updated. For $i \in [C]$, $k \in \Gamma$ and $l \in [L]$, the update $\Delta_{k,l}^{\langle i \rangle}$ is used to denote the corresponding increment generated in client i for the submodel symbol $M_{k,l}$. In particular, the update $\Delta_{k,l}^{\langle i \rangle}$ is 0 if $k \notin \Gamma^{\langle i \rangle}$. Thus for $k \in \Gamma$, the overall increment for the submodel symbol $M_{k,l}$ is $\sum_{i \in [C]} \Delta_{k,l}^{\langle i \rangle}$. The full increment sum is then defined as $\Delta_\Gamma = \{\sum_{i \in [C]} \Delta_{k,l}^{\langle i \rangle} : k \in \Gamma, l \in [L]\}$. Therefore, the updated full learning model $M'_{[K]}$ for $l \in [L]$ should be as follows,

$$M'_{k,l} = \begin{cases} M_{k,l}, & \text{if } k \in \Gamma \\ M_{k,l} + \sum_{i \in [C]} \Delta_{k,l}^{\langle i \rangle}, & \text{otherwise} \end{cases} \tag{1}$$

[1]For a positive integer Z, we use the notation $[Z] = \{1, 2, \ldots, Z\}$.

For $j \in [2]$, if $\mathcal{M}_j$ is used to denote all the information that can be obtained by database j, the FSL reliability constraint in one-round FSL training is captured by,

$$\text{[reliability]} \quad H(M'_{[K]}|\mathcal{M}_j) = 0, \quad \forall j \in [2] \tag{2}$$

As introduced in [3], the privacy constraint in FL requires that the aggregator learns nothing beyond the update sum from clients' local data. In FSL, since the full model is divided into multiple submodels, the privacy constraint needs to be tuned accordingly. Each individual database cannot infer any additional information about clients' local data beyond the desired submodel index union Γ and full increment sum Δ_Γ. Let $\mathcal{D}_i$ denote the local data in client i, given $\mathcal{D}_{[C]} = \{\mathcal{D}_i : i \in [C]\}$, the privacy constraint is precisely described by,

$$\text{[privacy]} \quad I(\mathcal{M}_j; \mathcal{D}_{[C]}|\Gamma, \Delta_\Gamma) = 0, \quad \forall j \in [2] \tag{3}$$

Using the multi-user PIR/SPIR problem formulated in [40], [41] as reference, each participating client should not gain any knowledge about the other clients' local data. For $i \in [C]$, let $\mathcal{W}_i$ denote all the information that can be obtained by client i, and let $\mathcal{D}_{\bar{i}}$ denote the set $\{\mathcal{D}_1, \ldots, \mathcal{D}_{i-1}, \mathcal{D}_{i+1}, \ldots, \mathcal{D}_C\}$, the inter-client privacy constraint is formed as follows,

$$\text{[inter-client privacy]} \quad I(\mathcal{W}_i; \mathcal{D}_{\bar{i}}) = 0, \quad \forall i \in [C] \tag{4}$$

A basic one-round FSL achievable scheme is one that satisfies the reliability constraint (2), the privacy constraint (3) and the inter-client privacy constraint (4). Moreover, we want these three basic constraints to remain satisfied in the presence of client drop-outs, client late-arrivals and database drop-outs. Further, it is necessary that this one-round FSL scheme can be executed in an iterative manner until a predefined termination criterion is satisfied, e.g., the accuracy of the updated full model exceeds the preset threshold or a preset maximal number of iterations is reached. In this work, the performance of an FSL scheme is measured by the total communication cost. For given FSL system parameters, our aim is to find a distributed and robust FSL scheme in which the total communication cost is as small as possible.

III. Main Result

Our main result is a new private FSL algorithm as described above. The following theorem gives its performance in terms of the total communication cost in the entire process including FSL-CRG, FSL-PSU, FSL-write phases. The proof of the theorem is given in Section IV-D.

Theorem 1 *The total communication cost of the proposed private FSL scheme in one round is $\mathcal{O}(CK + C|\Gamma|L)$ in q-ary bits, where C is the number of selected clients, K is the total number of submodels, and $|\Gamma|$ is the number of updated submodels in the given round. Here, $\mathcal{O}(CK)$ is due to the FSL-PSU phase, while $\mathcal{O}(C|\Gamma|L)$ is due to the FSL-write phase.*

Our proposed FSL scheme achieves unconditional information theoretic privacy and it is proved to be robust against client drop-outs, client late-arrivals, and database drop-outs. No constraint is imposed on the number of clients that may drop out during the FSL process. The communication cost $\mathcal{O}(CK + C|\Gamma|L)$ outperforms the best-known communication cost in the literature [3]–[6], which is at least $\mathcal{O}(CKL)$. In the seminal FSL work [10], the communication cost is $\mathcal{O}(C|\Gamma|)$ for the PSU phase and $\mathcal{O}(C|\Gamma|L)$ for the whole FSL process with a weaker privacy guarantee. Although this communication cost is a little better than our communication cost in terms of the PSU phase, the PSU [10] yields erroneous results while our PSU yields completely accurate result. Furthermore, the PSU and the subsequent secure aggregation are considered separately in [10]. Noting that the total number of submodels K is generally very large, we can further optimize the communication cost by adjusting the size of K. Specifically, as we decrease K, the product of $|\Gamma|$ and L will likely increase such that K and $|\Gamma|L$ will have the same order. Moreover, in the practical implementations, for each client, the upload speeds are typically much slower than download speeds during the client-database communications. Unlike the classical secure aggregation scheme in [3], the total communication time in our FSL process is further improved as almost all of the alive clients send only one answer to one database in each phase. In addition, while determining the two client groups, we can further improve the total communication time based on the actual bandwidth/quality of each client-database communication channel.

IV. General FSL Achievable Scheme

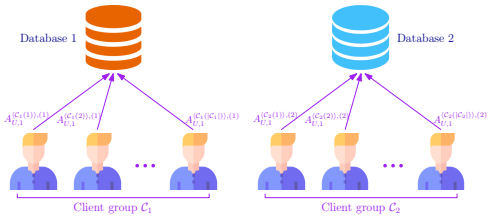
Our proposed scheme consists of three phases: common randomness generation phase (FSL-CRG), private determination of the index union of submodels to be updated (FSL-PSU), and private writing of the updated submodels in the union back to the databases (FSL-write), which are illustrated in detail in the following three subsections. In the last subsection, we analyze the reliability, privacy, robustness and performance of our proposed FSL scheme.

A. Common Randomness Generation (FSL-CRG) Phase

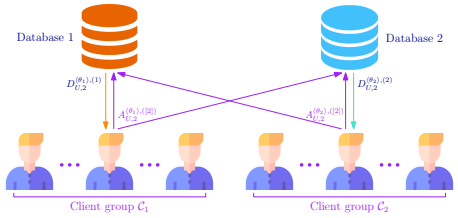
The two databases aim to establish two types of common randomness across the clients: The first type is a global common randomness symbol c that is uniformly selected from $\mathbb{F}_q \backslash \{0\}$. The second type is a set of general common randomness symbols $\{R_0, R_1, \ldots, R_C\}$ with a flexible set length $C+1$, where each symbol is uniformly selected from $\mathbb{F}_q$ and the sum of the last C symbols equals 0, i.e., $\sum_{i\in[C]} R_i = 0$. As a result, R_0 can be used as u_k or $w_{k,l}$ while $R_{[C]}$ can be used as $u_k^{\langle [C] \rangle}$ or $w_{k,l}^{\langle [C] \rangle}$ in the next two phases.

We start with a scheme for the second type. First, each database individually selects a random client from its client group as routing clients. Their indices are denoted by θ_1 and θ_2, respectively.[2] Second, both databases randomly select a set of symbols with size C from $\mathbb{F}_q$ under a uniform distribution, and then broadcast this set to the routing clients and the last

[2]Since the clients θ_1 and θ_2 may also drop-out, a potential solution is that each database randomly selects a small set of clients to route the information in parallel while its cardinality is based on the observed client drop-out rate.

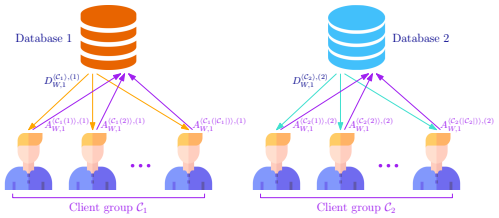


(a) FSL-PSU phase step 1.

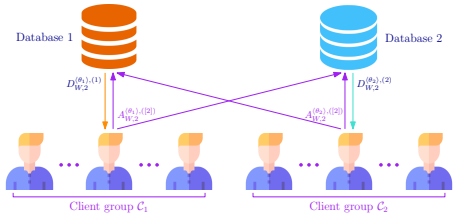


(b) FSL-PSU phase step 2.

Fig. 3: Data flow in the FSL-PSU phase of our system model.



(a) FSL-write phase step 1.



(b) FSL-write phase step 2.

Fig. 4: Data flow in the FSL-write phase of our system model.

client. Thus, these databases obtain a new set of symbols with size C through element-wise summation, and then append one more symbol R_C to the existing set such that the sum of the last C symbols equals zero. Moreover, each database also sends its $i+1$th random symbol to client i for all $i \in [C-1]$. Thus, client i can obtain the symbol R_i through summation. Each individual database has no knowledge about client-side common randomness because of the one-time pad encryption.

We next consider the first type. Since c should be uniform in $\mathbb{F}_q \backslash \{0\}$, we utilize a broadcast variation of the RSPIR scheme provided in [42, Section V] with the parameters being $N = 2, K = |q| - 1, L = 1$ and the message set being $W_1 = 1, W_2 = 2, \ldots, W_{|q|-1} = |q| - 1$. More details are provided in [38, Section 5.1]. Due to the user privacy constraint in RSPIR [42], c will be unknown to each individual database.

B. Private Set Union (FSL-PSU) Phase

After the FSL-CRG phase is completed, each selected client will obtain all the required client-side common randomness symbols as desired. Following our distributed FSL model in Section II, C selected clients are divided into two groups $\mathcal{C}_1$ and $\mathcal{C}_2$. Then, the ith client in $\mathcal{C}_1$ constructs its answer as,[3]

$$A_{U,1}^{\langle i \rangle,(1)} = \{c(Y_1^{\langle i \rangle} + u_1^{\langle i \rangle}), \ldots, c(Y_K^{\langle i \rangle} + u_K^{\langle i \rangle})\} \tag{5}$$

Similarly, the ith client in $\mathcal{C}_2$ constructs its answer as,

$$A_{U,1}^{\langle i \rangle,(2)} = \{c(Y_1^{\langle i \rangle} + u_1^{\langle i \rangle}), \ldots, c(Y_K^{\langle i \rangle} + u_K^{\langle i \rangle})\} \tag{6}$$

Once database 1 receives all the answers from its associated clients in $\mathcal{C}_1$, it produces a response to be downloaded as,

$$D_{U,2}^{\langle \theta_1 \rangle,(1)} = \left\{ c \sum_{i \in \mathcal{C}_1} (Y_k^{\langle i \rangle} + u_k^{\langle i \rangle}) + S_k : k \in [K] \right\} \tag{7}$$

[3]In this work, we use the value in $\langle\rangle$ to denote the index of client and the value in $()$ to denote the index of database for clarity. The first subscript of D or A is used to show it is within the FSL-PSU phase or FSL-write phase (the letter U stands for union and the letter W stands for write), whereas the second subscript is used to denote the step number within this phase.

where $\{S_k : k \in [K]\}$ are server-side common randomness symbols that are uniformly selected from $\mathbb{F}_q$. This produced response $D_{U,2}^{\langle \theta_1 \rangle,(1)}$ will then be downloaded by the routing client θ_1 in $\mathcal{C}_1$. Afterwards, client θ_1 forwards the following further processed answer to both databases,

$$A_{U,2}^{\langle \theta_1 \rangle,([2])} = \left\{ c \sum_{i \in \mathcal{C}_1} (Y_k^{\langle i \rangle} + u_k^{\langle i \rangle}) + u_k + S_k : k \in [K] \right\} \tag{8}$$

Likewise, the second database produces a response as follows,

$$D_{U,2}^{\langle \theta_2 \rangle,(2)} = \left\{ c \sum_{i \in \mathcal{C}_2} (Y_k^{\langle i \rangle} + u_k^{\langle i \rangle}) - S_k : k \in [K] \right\} \tag{9}$$

This produced response will then be downloaded by the routing client θ_2 in $\mathcal{C}_2$. Afterwards, this client forwards the following answer to both databases,

$$A_{U,2}^{\langle \theta_2 \rangle,([2])} = \left\{ c \sum_{i \in \mathcal{C}_2} (Y_k^{\langle i \rangle} + u_k^{\langle i \rangle}) - u_k - S_k : k \in [K] \right\} \tag{10}$$

After collecting these two answer sets in the second communication step, each individual database is ready to derive the union Γ by performing the following element-wise summation,

$$A_{U,2}^{\langle \theta_1 \rangle,(j)} + A_{U,2}^{\langle \theta_2 \rangle,(j)} = \left\{ c \sum_{i \in [C]} Y_1^{\langle i \rangle}, \ldots, c \sum_{i \in [C]} Y_K^{\langle i \rangle} \right\} \tag{11}$$

Each individual database utilizes the value of its calculated expression $c \sum_{i \in [C]} Y_k^{\langle i \rangle}$ (whether it is zero or not) to judge whether the index k is in the union Γ or not, and thereby, to determine Γ.

C. Private Write (FSL-Write) Phase

When the FSL-PSU phase is complete, each database individually sends the set of submodels M_Γ to its associated clients. Then, the ith client in $\mathcal{C}_1$ will generate the increments for each desired submodel whose index belongs to $\Gamma^{\langle i \rangle}$ according to its local data and construct a well-processed answer

accordingly. Specifically, for all $k \in \Gamma^{\langle i \rangle}$, the answer symbols are generated in the following form,

$$A_{W,1}^{\langle i \rangle,(1)}(k) = \{\Delta_{k,1}^{\langle i \rangle} + w_{k,1}^{\langle i \rangle}, \ldots, \Delta_{k,L}^{\langle i \rangle} + w_{k,L}^{\langle i \rangle}\} \tag{12}$$

For all $k \in \Gamma \backslash \Gamma^{\langle i \rangle}$, the answer symbols are generated as follows without any updates concerning the current submodel,

$$A_{W,1}^{\langle i \rangle,(1)}(k) = \{w_{k,1}^{\langle i \rangle}, \ldots, w_{k,L}^{\langle i \rangle}\} \tag{13}$$

Thus, the ultimate answer generated by this client in the first step is $A_{W,1}^{\langle i \rangle,(1)} = \{A_{W,1}^{\langle i \rangle,(1)}(k) : k \in \Gamma\}$. The ith client in $\mathcal{C}_2$ will generate an ultimate answer $A_{W,1}^{\langle i \rangle,(2)} = \{A_{W,1}^{\langle i \rangle,(2)}(k) : k \in \Gamma\}$ in the same way. Subsequently, each client sends its answer to its associated database. These two databases also share another set of server-side common randomness symbols $\{S_{k,l} : k \in [K], l \in [L]\}$ from $\mathbb{F}_q$. Let $C_k^{(1)}$ be the index set of clients in $\mathcal{C}_1$ whose desired submodel index set includes the index k, i.e., $C_k^{(1)} = \{i \in \mathcal{C}_1 | k \in \Gamma^{\langle i \rangle}\}$. Similarly, $C_k^{(2)}$ and C_k are defined as $\{i \in \mathcal{C}_2 | k \in \Gamma^{\langle i \rangle}\}$ and $\{i \in [C] \| k \in \Gamma^{\langle i \rangle}\}$, respectively. After collecting all the answers $A_{W,1}^{\langle \mathcal{C}_1 \rangle,(1)}$ from $\mathcal{C}_1$, database 1 calculates the following aggregation increment for the lth symbol of the kth submodel where k belongs to Γ,

$$\sum_{i \in C_k^{(1)}} \left(\Delta_{k,l}^{\langle i \rangle} + w_{k,l}^{\langle i \rangle}\right) + \sum_{i \in \mathcal{C}_1 \backslash C_k^{(1)}} w_{k,l}^{\langle i \rangle} = \sum_{i \in C_k^{(1)}} \Delta_{k,l}^{\langle i \rangle} + \sum_{i \in \mathcal{C}_1} w_{k,l}^{\langle i \rangle} \tag{14}$$

As in the last FSL-PSU phase, the corresponding response is produced as follows and will be downloaded by the client θ_1,

$$D_{W,2}^{\langle \theta_1 \rangle,(1)} = \left\{ \sum_{i \in C_k^{(1)}} \Delta_{k,l}^{\langle i \rangle} + \sum_{i \in \mathcal{C}_1} w_{k,l}^{\langle i \rangle} + S_{k,l} : k \in \Gamma, l \in [L] \right\} \tag{15}$$

Once this response is received by the client θ_1, this client forwards the following answer to both databases,

$$A_{W,2}^{\langle \theta_1 \rangle,([2])} = \left\{ \sum_{i \in C_k^{(1)}} \Delta_{k,l}^{\langle i \rangle} + \sum_{i \in \mathcal{C}_1} w_{k,l}^{\langle i \rangle} + w_{k,l} + S_{k,l} : k \in \Gamma, l \in [L] \right\} \tag{16}$$

Meanwhile, database 2 produces the following response and this response will be downloaded by the client θ_2,

$$D_{W,2}^{\langle \theta_2 \rangle,(2)} = \left\{ \sum_{i \in C_k^{(2)}} \Delta_{k,l}^{\langle i \rangle} + \sum_{i \in \mathcal{C}_2} w_{k,l}^{\langle i \rangle} - S_{k,l} : k \in \Gamma, l \in [L] \right\} \tag{17}$$

The answer forwarded by the client θ_2 to both databases is,

$$A_{W,2}^{\langle \theta_2 \rangle,([2])} = \left\{ \sum_{i \in C_k^{(2)}} \Delta_{k,l}^{\langle i \rangle} + \sum_{i \in \mathcal{C}_2} w_{k,l}^{\langle i \rangle} - w_{k,l} - S_{k,l} : k \in \Gamma, l \in [L] \right\} \tag{18}$$

At this point, each individual database is ready to aggregate the updates as desired from all the selected clients by summing up $A_{W,2}^{\langle \theta_1 \rangle,([2])}$ and $A_{W,2}^{\langle \theta_2 \rangle,([2])}$ in an element-wise manner. The updated lth symbol of the kth submodel stored in the server after performing this round of FSL-write is finally $M_{k,l} + \sum_{i \in C_k} \Delta_{k,l}^{\langle i \rangle}$, which is exactly the expected $M'_{k,l}$.

D. Analysis of the Proposed Scheme

Note that the FSL-write phase is essentially a simplified application of the FSL-PSU phase without using the symbol c. Thus, we only need to analyze the FSL-PSU phase in terms of reliability, privacy and robustness, and these will be inherited directly by the FSL-write phase. By combining the facts that reliability, privacy and robustness constraints are satisfied by both phases, we readily obtain the reliability, privacy and robustness for the entirety of our FSL scheme.

a) Reliability constraint: Reliability is shown at the end of FSL-PSU phase and FSL-write phase.

b) Privacy constraint: For $i \in [C]$ and $k \in \Gamma$, $u_k^{\langle i \rangle}$ is used to protect the privacy of $Y_k^{\langle i \rangle}$ such that each database knows nothing about the value of $Y_k^{\langle i \rangle}$ due to the one-time pad encryption. Further, for $k \in \Gamma$, c is used to protect the privacy of $\sum_{i \in [C]} Y_k^{\langle i \rangle}$ such that each database knows nothing about the value of $\sum_{i \in [C]} Y_k^{\langle i \rangle}$ beyond that this sum is zero or not due to the finite cyclic group under multiplication in $\mathbb{F}_q \backslash \{0\}$. The concrete proof follows from the proof of client's privacy in [39, Subsection V.B] as the received answer in each database contains less information than the answer set $\{A_{U,1}^{\langle \mathcal{C}_1 \rangle,(1)}, A_{U,1}^{\langle \mathcal{C}_2 \rangle,(2)}\}$ with respect to the incidence vectors $Y^{\langle [C] \rangle}$. Thus, each database can only learn Γ from the clients.

c) Inter-client privacy constraint: Only the routing clients θ_1 and θ_2 receive information from outside. Due to the unknown server-side common randomness in the downloads, neither the θ_1th client nor the θ_2th client can learn any knowledge about the local data within the other clients.

d) Client drop-out robustness: The basic idea is that the routing clients θ_1 and θ_2 can adjust the answer in the second step by additionally appending the sum of missing client-side common randomness symbols incurred by those clients that drop-out. The detailed analysis is available in [38, Section 5.2].

e) Client late-arrival robustness: Assume that an answer generated by an arbitrary client $i \in \mathcal{C}_j$ in the first step arrives at database j late. Even though database j receives the information $A_{U,1}^{\langle i \rangle,(j)}$ separately, it is still unable to extract any information about the incidence vector $Y^{\langle i \rangle}$ because of the unknown extra common randomness $\{u_k : k \in [K]\}$. This conclusion can be extended to an arbitrary set of late clients.

f) Database drop-out robustness: If database 1 drops-out and cannot function temporarily, database 2 needs to communicate with client θ_2 one more time for the values of $\{c \sum_{i \in \mathcal{C}_1} u_k^{\langle i \rangle} : k \in [K]\}$ to derive the union of the client group $\mathcal{C}_2$. This solution also applies to the drop-out of database 2.

g) Communication cost: In the FSL-PSU phase, without considering the cost generated in the FSL-CRG phase, the communication cost is $(C+6)K$. The extra communication cost is about $8CK$ for the required client-side common randomness, while the communication cost of the symbol c is negligible. Therefore, the total communication cost in this phase is about $(9C+6)K$. Following the same calculation in the FSL-PSU phase, the total communication cost in the FSL-write phase is about $(10C+6)|\Gamma|L$ in which $C|\Gamma|L$ is for the clients to download the submodels from the server.

REFERENCES

[1] B. McMahan and D. Ramage. Federated learning: Collaborative machine learning without centralized training data. Available at https://ai.googleblog.com/2017/04/federated-learning-collaborative.html.

[2] Q. Yang, Y. Liu, T. Chen, and Y. Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology*, 10(2):1–19, March 2019.

[3] K. Bonawitz, V. Ivanov, et al. Practical secure aggregation for privacy preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, page 1175–1191, 2017.

[4] J. Bell, K. A. Bonawitz, A. Gascón, T. Lepoint, and M. Raykova. Secure single-server aggregation with (poly)logarithmic overhead. In *Cryptology ePrint Archive*, 2020.

[5] J. So, B. Guler, and A. S. Avestimehr. Turbo-aggregate: Breaking the quadratic aggregation barrier in secure federated learning. In *Cryptology ePrint Archive*, 2020.

[6] S. Kadhe, N. Rajaraman, O. O. Koyluoglu, and K. Ramchandran. Fastsecagg: Scalable secure aggregation for privacy-preserving federated learning. Available at arXiv:2009.11248.

[7] Y. Zhao and H. Sun. Information theoretic secure aggregation with user dropouts. *IEEE Trans. on Info. Theory*, 68(11):7471–7484, November 2022.

[8] K. Wan, H. Sun, M. Ji, and G. Caire. Information theoretic secure aggregation with uncoded groupwise keys. Available at arXiv:2204.11364.

[9] Y. Zhao and H. Sun. Secure summation: Capacity region, groupwise key, and feasibility. Available at arXiv:2205.08458.

[10] C. Niu, F. Wu, S. Tang, L. Hua, R. Jia, C. Lv, Z. Wu, and G. Chen. Billion-scale federated learning on mobile clients: A submodel design with tunable privacy. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, pages 1–14, 2020.

[11] Z. Jia and S. A. Jafar. X-secure T-private federated submodel learning with elastic dropout resilience. *IEEE Trans. on Info. Theory*, 68(8):5418–5439, August 2022.

[12] Z. Jia, H. Sun, and S. A. Jafar. Cross subspace alignment and the asymptotic capacity of X-secure T-private information retrieval. *IEEE Trans. on Info. Theory*, 65(9):5783–5798, September 2019.

[13] S. Vithana and S. Ulukus. Efficient private federated submodel learning. In *IEEE ICC*, pages 3394–3399, May 2022.

[14] S. Vithana and S. Ulukus. Private read update write (PRUW) in federated submodel learning (FSL): Communication efficient schemes with and without sparsification. Available at arXiv:2209.04421.

[15] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan. Private information retrieval. *Journal of the ACM*, 45(6):965–981, November 1998.

[16] Y. Gertner, Y. Ishai, E. Kushilevitz, and T. Malkin. Protecting data privacy in private information retrieval schemes. *Journal of Computer and System Sciences*, 60(3):592–629, June 2000.

[17] H. Sun and S. A. Jafar. The capacity of private information retrieval. *IEEE Trans. on Info. Theory*, 63(7):4075–4088, July 2017.

[18] K. Banawan and S. Ulukus. The capacity of private information retrieval from coded databases. *IEEE Trans. on Info. Theory*, 64(3):1945–1956, March 2018.

[19] H. Sun and S. A. Jafar. The capacity of robust private information retrieval with colluding databases. *IEEE Trans. on Info. Theory*, 64(4):2361–2370, April 2018.

[20] K. Banawan and S. Ulukus. The capacity of private information retrieval from Byzantine and colluding databases. *IEEE Trans. on Info. Theory*, 65(2):1206–1219, February 2019.

[21] Q. Wang, H. Sun, and M. Skoglund. The capacity of private information retrieval with eavesdroppers. *IEEE Trans. on Info. Theory*, 65(5):3198–3214, May 2019.

[22] K. Banawan and S. Ulukus. Multi-message private information retrieval: Capacity results and near-optimal schemes. *IEEE Trans. on Info. Theory*, 64(10):6842–6862, October 2018.

[23] R. Tandon. The capacity of cache aided private information retrieval. In *Allerton Conference*, pages 1078–1082, October 2017.

[24] Y.-P. Wei, K. Banawan, and S. Ulukus. Fundamental limits of cache-aided private information retrieval with unknown and uncoded prefetching. *IEEE Trans. on Info. Theory*, 65(5):3215–3232, May 2019.

[25] Z. Chen, Z. Wang, and S. A. Jafar. The capacity of T-private information retrieval with private side information. *IEEE Trans. on Info. Theory*, 66(8):4761–4773, August 2020.

[26] M. A. Attia, D. Kumar, and R. Tandon. The capacity of private information retrieval from uncoded storage constrained databases. *IEEE Trans. on Info. Theory*, 66(11):6617–6634, November 2020.

[27] C. Tian. On the storage cost of private information retrieval. *IEEE Trans. on Info. Theory*, 66(12):7539–7549, December 2020.

[28] C. Tian, H. Sun, and J. Chen. Capacity-achieving private information retrieval codes with optimal message size and upload cost. *IEEE Trans. on Info. Theory*, 65(11):7613–7627, November 2019.

[29] I. Samy, M. Attia, R. Tandon, and L. Lazos. Asymmetric leaky private information retrieval. *IEEE Trans. on Info. Theory*, 67(8):5352–5369, August 2021.

[30] S. Vithana, K. Banawan, and S. Ulukus. Semantic private information retrieval. *IEEE Trans. on Info. Theory*, 68(4):2635–2652, April 2022.

[31] H. Sun and S. A. Jafar. The capacity of symmetric private information retrieval. *IEEE Trans. on Info. Theory*, 65(1):322–329, January 2019.

[32] Z. Wang and S. Ulukus. Symmetric private information retrieval at the private information retrieval rate. *IEEE Jour. on Selected Areas in Info. Theory*, 3(2):350–361, June 2022.

[33] Z. Wang and S. Ulukus. Communication cost of two-database symmetric private information retrieval: A conditional disclosure of multiple secrets perspective. In *IEEE ISIT*, pages 402–407, June 2022.

[34] Q. Wang and M. Skoglund. On PIR and symmetric PIR from colluding databases with adversaries and eavesdroppers. *IEEE Trans. on Info. Theory*, 65(5):3183–3197, May 2019.

[35] Q. Wang and M. Skoglund. Symmetric private information retrieval from MDS coded distributed storage with non-colluding and colluding servers. *IEEE Trans. on Info. Theory*, 65(8):5160–5175, August 2019.

[36] J. Cheng, N. Liu, W. Kang, and Y. Li. The capacity of symmetric private information retrieval under arbitrary collusion and eavesdropping patterns. *IEEE Trans. on Info. Forensics and Security*, 17:3037–3050, August 2022.

[37] Z. Wang, K. Banawan, and S. Ulukus. Private set intersection: A multi-message symmetric private information retrieval perspective. *IEEE Trans. on Info. Theory*, 68(3):2001–2019, March 2022.

[38] Z. Wang and S. Ulukus. Private federated submodel learning via private set union. Available at arXiv:2301.07686.

[39] Z. Wang, K. Banawan, and S. Ulukus. Multi-party private set intersection: An information-theoretic approach. *IEEE Jour. on Selected Areas in Info. Theory*, 2(1):366–379, March 2021.

[40] Y. Lu, Z. Jia, and S. A. Jafar. Double blind T-private information retrieval. *IEEE Jour. on Selected Areas in Info. Theory*, 2(1):428–440, March 2021.

[41] J. Zhu, Q. Yan, and X. Tang. Multi-user blind symmetric private information retrieval from coded servers. *IEEE Jour. on Selected Areas in Commun.*, 40(3):815–831, March 2022.

[42] Z. Wang and S. Ulukus. Digital blind box: Random symmetric private information retrieval. In *IEEE ITW*, pages 95–100, November 2022.