Received July 13, 2021, accepted July 20, 2021, date of publication July 26, 2021, date of current version July 30, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3099400

Fast and Privacy-Preserving Federated Joint Estimator of Multi-sUGMs

XIAO TAN, TIANYI MA, AND TONGTONG SU
School of Computer Science and Engineering, Southeast University, Nanjing 211189, China
School of Artificial Intelligence, Southeast University, Nanjing 211189, China

Corresponding author: Xiao Tan (seu_tanxiao@outlook.com)

This work was supported in part by the National Key Research and Development Program of China under Grant 2018AAA0100500, in part by the Fundamental Research Funds for the Central Universities, in part by the Jiangsu Provincial Key Laboratory of Network and Information Security under Grant BM2003201, and in part by the Key Laboratory of Computer Network and Information Integration of Ministry of Education of China under Grant 93K-9.

ABSTRACT Learning multiple related graphs from many distributed and privacy-required resources is an important and common task in neuroscience applications. Medical researchers can comprehensively investigate the diagnostic evidence and understand the cause of certain brain diseases via analyzing the commonalities and differences of the brain connectomes predicted from the fMRI data across multiple hospitals. Previous sparse Undirected Graphical Model (sUGM) methods either cannot take full usage of the heterogeneous data while preserving privacy or miss the capability of handling the nonparanormal data, which is highly non-independent and identically distributed (non-i.i.d.). This paper proposes a novel and efficient approach, FEDJEM (federated joint estimator of multiple sUGMs), that trains the multi-sUGMs over a massive network encompassing various local devices and the global center. In order to efficiently process the datasets with different nonparanormal distributions, the proposed federated algorithm fully exploits the computing power of the local devices and cloud center while federated updates ensure that personal data remain local, thus defending the privacy. We also implement a general federated learning framework for multi-task learning based on our method. We apply our method on multiple simulation datasets to evaluate its speed and accuracy in comparison with relevant baselines and develop a strategy accordingly to balance its computation and communication abilities. Finally, we predict several informative groups of connectomes based on the real-world dataset.

INDEX TERMS Federated learning, multi-task learning, graphical model.

I. INTRODUCTION

There has been a wild revolution in collecting massive heterogeneous data [1], [2] across many scientific fields in recent years. For example, different hospitals have been constantly collecting the medical data and information of Alzheimer's patients of different races, genders, ages, and regions with widely used and advanced medical devices. Learning multiple related graphs from such heterogeneous data has become an important task. For instance, we can learn multiple related brain connectomes of Alzheimer's patients from several fMRI datasets obtained from different hospitals. Those connectomes can help researchers better understand the characteristics and underlying causes of the disease. Unlike transfer learning, learning multiple related networks facilitates the exploration of the similarities and differences of connectomes without transferring datasets from related tasks. Doctors can investigate diagnostic evidence based on the shared features revealed by the commonalities of the connectomes. In addition, novel causes of a disease may emerge when analyzing the differences from normal connectomes [3].

Traditionally, the Ising model [4], a method based on ℓ_1-regularized logistic regression, is applied to estimate the structure of a high-dimensional graph. However, three main obstacles that prevent real-world applications from learning multiple graphical models from heterogeneous data exist in this approach [5]:

- **Dilemma between performance and communication:** Massive data can enhance model performance by increasing data heterogeneity, but also create

The associate editor coordinating the review of this manuscript and approving it for publication was Liang-Bi Chen.

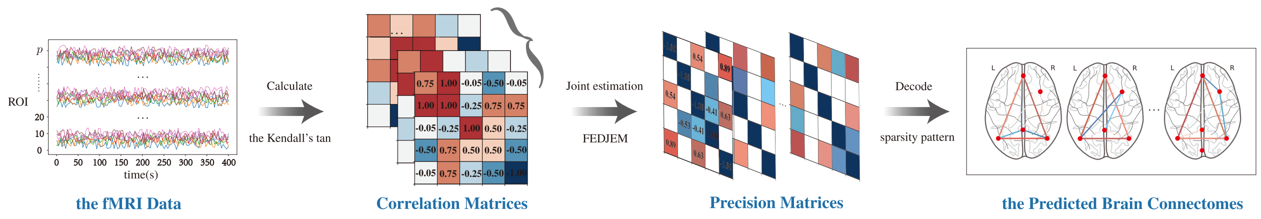

FIGURE 1. An illustration of the training model. Multiple related graphs are inferred from the heterogeneous datasets here over a three-step process. 1) We calculate the Kendall's tau correlation estimation of the fMRI data to obtain the correlation matrices. 2) We apply FEDJEM to the joint estimation of the precision matrices. 3) We recover the brain connectomes by decoding the sparsity pattern of the precision matrices.

unacceptable communication load due to the high data capacity [6], [7]. Applying fewer data samples is an intuitive way to resolve this problem, but it drives down model performance [8]. In most cases, it is not possible to obtain high performance and low communication cost simultaneously.

- **Privacy preservation:** Patient data typically contain personal information [9]. Collecting such data in the cloud center can cause a serious privacy problem. Researchers are not allowed to access raw data from the hospital database [10] directly in most circumstances due to the high risk of leaking private information.
- **Non-independent and identically distributed (non-i.i.d.) datasets:** To enhance the heterogeneity of data, it is necessary to collect massive data from different sources; this gives the datasets disparate distributions. There is a serious bottleneck to exploiting training data if the datasets from different tasks follow non-independent and identically distributions, e.g., one dataset follows a Gaussian distribution but the other one does not. Many existing methods require the datasets from different tasks to follow the same or similar distributions. There is a huge reduction in model performance when applying them to non-i.i.d data.

To overcome these obstacles, researchers have developed a series of sparse Gaussian Graphical Model (sGGM) methods which can effectively predict graphs with relatively few data samples under the Gaussian assumption. The graph structure is hidden in the inverse of the covariance matrix, namely, the precision matrix (i.e., $\Omega = \Sigma^{-1}$). GLasso [11], [12], for example, inferred a graph through decoding the sparsity pattern in a precision matrix. The ℓ_1-penalized log-likelihood method is applied to obtain the precision matrix. Unlike GLasso, CLIME makes an inverse matrix estimation by optimizing an ℓ_1 constrained problem with regards to the precision matrix. Those models overcome communication and privacy challenges successfully because they learn each model on each dataset individually. Each model can be trained in local devices without requiring data transmission (i.e., with no communication cost), thus protecting patient privacy [13]. However, such individual training models still tend to encounter the first obstacle described above; they also tend to have inferior performance compared to models based on heterogeneous data.

In an effort to resolve the model performance issue, researchers proposed Multi-task Learning [14] in 1997. This approach has since been applied throughout various machine-learning domains. Previous researchers have also introduced multi-task learning to improve the generalization of the single sGGM in response to the data scarcity problem. [15]–[24] proposed the multi-task sGGM, which jointly estimates K different but related sGGMs. These methods enhance the generalization ability of the model, but require that all data should be collected in the cloud center. As mentioned above, this implies a significant risk that private data might be leaked and heavy communication load which causes unacceptable transmission time. In addition, current multi-task sGGMs assume that all the data from different tasks follow the Gaussian distribution. This is not the case in most real-world application situations. In general, most multi-task sGGM methods fail to get out of the dilemma and still struggle in the last two challenges.

In this study, we develop a novel model, the federated joint estimator of multiple sparse Undirected Graphical Models (FEDJEM), to estimate multiple sparse Undirected Graphical Models (sUGMs) jointly via a federated learning algorithm. We design this model to address the three challenges discussed above. This model allows local data to be processed and computed in local devices, then communicates their updates to the cloud center in order to train a global graphical model. FEDJEM (illustrated in Fig. 1) can also handle multivariate nonparanormal data, which relaxes the normality assumption that most real-world non-i.i.d. data do not follow. Our contributions can be summarized as follows:

- **Novel model:** We present the novel, privacy-preserving FEDJEM, a federated multi-task sUGM method, wherein a federated update algorithm first separates the computation process into local devices and a global center. We design the model to safeguard private information as only model parameters are transmitted in this process. All personal data are stored and processed locally and prior to model training.
- **Novel relaxation:** Considering the properties of real-world non-i.i.d. data, our method can manage data with a nonparanormal distribution, which is a much larger superset of the Gaussian Distribution.
- **Swiftness and efficiency:** Our method is a fast and efficient federated algorithm. The computing power of local

devices is harnessed to minimize the computation load on the cloud center. The transmitted data are relatively small because only parameter updates are communicated between the edges and center.
- **General framework:** We implement a general federated learning framework for multi-task learning based on our method. This framework would allow others to easily utilize federated multi-task learning while only focusing on the implementation of local and global updates. Our codes can be found on https://github.com/MahjongGod-Saki/FEDJEM

The rest of this paper is organized as follows. Section II provides the relevant background material. Section III introduces our method in detail and discusses its main properties. Section IV gives an analysis of its computation and communication cost and convergence. Section V presents previous work related to our method. Our experimental results are discussed in Section VI. Section VII suggests several potential strategies to balance local and global computation and communication time in practice and provides some ideas of future work. Section VIII provides a brief summary and concluding remarks.

A. NOTATIONS

We choose $\mathbf{X}^{(i)} \in \mathbb{R}^{n_i \times p}$ to represent the i-th dataset with n_i samples and p features. $\{\mathbf{X}^{(i)}\} = \{\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \dots, \mathbf{X}^{(K)}\}$ denotes K datasets generated by K different tasks and nodes. $\Sigma^{(i)} \in \mathbb{R}^{p \times p}$ represents the covariance matrix of the dataset $\mathbf{X}^{(i)}$. $\Omega^{(i)} = \Sigma^{(i)^{-1}} \in \mathbb{R}^{p \times p}$ denotes the inverse of the covariance matrix. $\{\Sigma^{(i)}\} = \{\Sigma^{(1)}, \dots, \Sigma^{(K)}\}$ and $\{\Omega^{(i)}\} = \{\Omega^{(1)}, \dots, \Omega^{(K)}\}$ are the sets of the covariance matrices and precision matrices corresponding to the datasets. We list the notations used in this paper in Table 1.

II. BACKGROUND

A. SPARSE GAUSSIAN GRAPHICAL MODEL

A single-task sparse Gaussian Graphical Model (sGGM) assumes that data samples follow a normal distribution $N(\mu, \Sigma)$ with mean vector μ and covariance matrix Σ. The graphical lasso (GLasso) is a penalized maximum likelihood estimator for precision matrix Ω inference. The model can be written as:

$$\begin{aligned}\hat{\Omega}_{\text{glasso}} &= \arg\min_{\Omega \geq 0} L(\Omega) + \lambda\|\Omega\|_1 \\ &= \arg\min_{\Omega \geq 0} -\log\det(\Omega) + <\Omega, \Sigma> + \lambda\|\Omega\|_1, \quad (1)\end{aligned}$$

where $L(\Omega)$ is the log-likelihood function of Ω.

B. FEDERATED MULTI-TASK LEARNING

The federated learning problem involves training a global machine learning model from the data stored locally, i.e., on multiple remote devices. The goal of this learning strategy is to locally store and process data generated by the devices. We only communicate the intermediate updates of parameters periodically utilizing central computing power. The typical optimization problem of federated learning is:

$$\arg\min_{\omega} \mathcal{L}(\omega) = \sum_{i=1}^{m} p_i \mathcal{L}_i(\omega). \quad (2)$$

where $\mathcal{L}_i$ is the objective function the i-th device and p_i is its weight, $p_i \geq 0$ and $\sum_i p_i = 1$.

Federated learning comes with statistical challenges in regards to training machine-learning models, mainly due to the variability of the number of data points on each device. A single global model cannot capture every piece of local knowledge. We should naturally obtain separate models for each node rather than training a single global model across the network. [25] shows that a combination of federated learning and multi-task learning, namely multi-task federated learning, performs significantly better. The federated multi-task learning framework can be formalized as follows:

$$\arg\min_{\boldsymbol{\omega}} (\mathcal{L}(\boldsymbol{\omega}) + \mathcal{R}(\boldsymbol{\omega})) = \sum_{i=1}^{m} p_i \mathcal{L}_i(\omega_i) + \mathcal{R}_{\text{total}}(\boldsymbol{\omega}), \quad (3)$$

where $\boldsymbol{\omega} = (\omega_1, \omega_2, \dots, \omega_m)$, $p_i \geq 0$, and $\sum_i p_i = 1$.

TABLE 1. Notations used in this paper.

Symbol	Meaning
K	the number of tasks, the number of devices
n_i	the number of samples of $\mathbf{X}^{(i)}$
p	the feature dimension of the sample
$\mathbf{X}^{(i)}$	the i-th dataset stored on ith device
$\Sigma^{(i)}$	the i-th covariance matrix corresponding to the $\mathbf{X}^{(i)}$
$\Omega^{(i)}$	the i-th precision matrix corresponding to the $\mathbf{X}^{(i)}$
$\Psi^{(i)}$	ADMM variable which equals $\Omega^{(i)}$
$\mathbf{U}^{(i)}$	ADMM dual variables
$\mathbf{S}^{(i)}$	Kendall's correlation matrix

III. METHOD: FEDERATED JOINT ESTIMATOR OF MULTIPLE sUGMs

We focus here on a federated multi-task undirected graphical model problem. To resolve this problem, we design the proposed method as per four distinct properties. 1) Various data are stored and processed on different local devices in a distributed environment, so every node trains its own model locally and then communicates its updates to the center in order to train a global model. The model parameters are updated in a federated manner. 2) The collected data are in a highly non-i.i.d. manner due to the differences of the storage devices and data source. 3) Personal data are kept safe because each node does not communicate its data with other devices and instead processes it locally. 4) Compared with the single-task graphical model, our federated multi-task graphical model fuses both global similarity knowledge and differences between tasks. This encourages better performance on real-world data, as evidenced by our experimental results. Fig. 2 shows the detailed flow diagram of our proposed method.

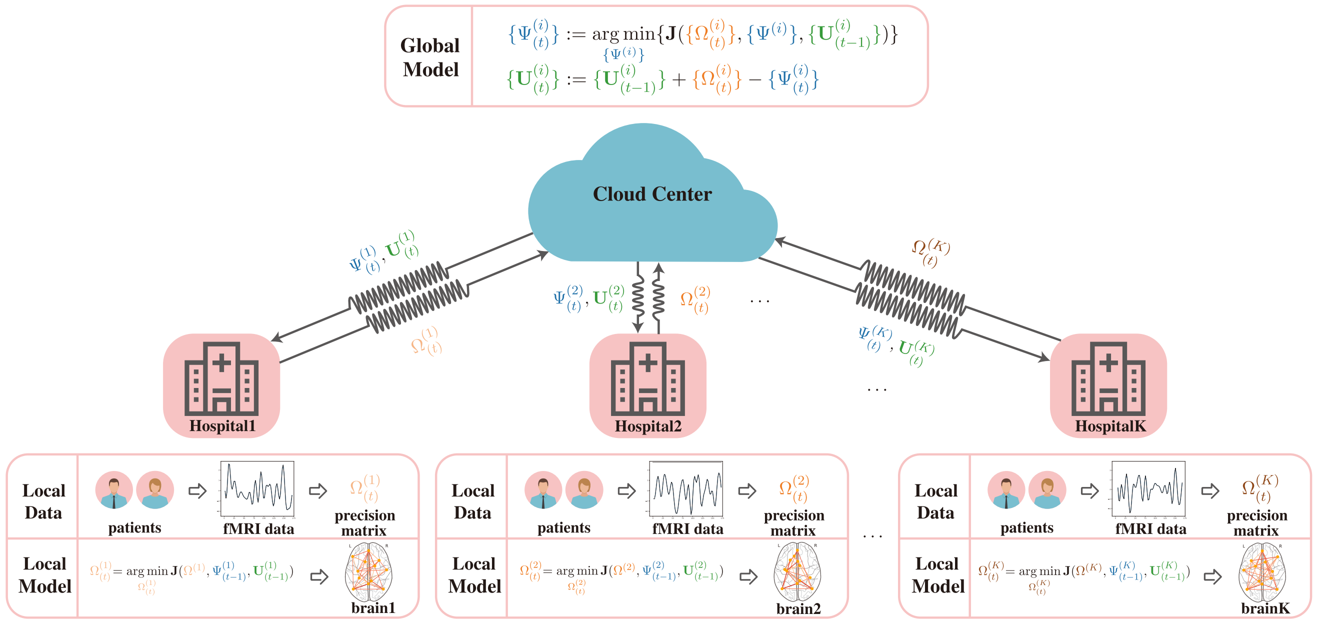
Global Model
$\{\Psi^{(i)}_{(t)}\} := \arg\min_{\{\Psi^{(i)}\}}\{\mathbf{J}(\{\Omega^{(i)}_{(t)}\}, \{\Psi^{(i)}\}, \{\mathbf{U}^{(i)}_{(t-1)}\})\}$
$\{\mathbf{U}^{(i)}_{(t)}\} := \{\mathbf{U}^{(i)}_{(t-1)}\} + \{\Omega^{(i)}_{(t)}\} - \{\Psi^{(i)}_{(t)}\}$
Cloud Center
Hospital1
Hospital2
HospitalK
Local Data
patients
fMRI data
precision matrix
Local Model
brain1
brain2
brainK

FIGURE 2. The flow diagram of proposed method. As the proposed method is applied, each hospital stores and processes the fMRI data of its patients locally. We estimate the precision matrix $\Omega^{(i)}_{(t)}$ of each task in the t-th iteration through the local update step in local devices. Next, we communicate the updated precision matrix to the cloud center. The center updates the variables $\Psi^{(i)}_{(t)}$ and $\mathbf{U}^{(i)}_{(t)}$ according to the received $\{\Omega^{(i)}_{(t)}\}$. Finally, the model transmits the global updates back to the edge for the $(t+1)$th iteration.

Our goal is to estimate multiple related graphs $\{\Omega^{(i)}\}$. Based on (3), our federated multi-task learning framework can be formalized as follows:

$$\{\hat{\Omega}^{(i)}\} = \arg\min_{\Omega^{(i)}\succeq 0} \sum_{i=1}^{K} p_i \mathcal{L}_i(\Omega^{(i)}) + \mathcal{R}_{\text{total}}(\{\Omega^{(i)}\}). \tag{4}$$

Three components remain to be determined in the above equation: 1) The **objective function** $\mathcal{L}_i(\cdot)$ is related to the log-likelihood function mentioned in (1), 2) the **weight** p_i, which should be associated with the number of samples, and 3) the **total regularization function** $\mathcal{R}_{\text{total}}(\cdot)$, which should be able to capture both the sparsity pattern and the heterogeneity of the precision matrices.

Based on (1), we first apply the log-likelihood function to design $\mathcal{L}_i(\Omega^{(i)}) = N \cdot L(\Omega^{(i)})$, where $L(\Omega^{(i)})$ is as in (1) and $N = \sum_i n_i$. To make $\sum_i p_i = 1$, we set $p_i = \frac{n_i}{N}$. We let $\mathcal{R}_{\text{total}}(\{\Omega^{(i)}\}) = \lambda_1\|\Omega^{(i)}\|_1 + \lambda_2\mathcal{R}(\Omega^{(1)}, \ldots, \Omega^{(K)})$. We choose ℓ_1 norm as our first regularization function to constrain the sparsity of every precision matrix $\Omega^{(i)}$. The second regularization function $\mathcal{R}(\cdot)$ enforces the group sparsity or similarity of all the precision matrices $\{\Omega^{(i)}\}$. From (4), our proposed method can be represented as (5).

$$\begin{aligned}\{\hat{\Omega}^{(i)}\} &= \arg\min_{\Omega^{(i)}\succeq 0} \sum_{i=1}^{K} n_i L(\Omega^{(i)}) + \mathcal{R}_{\text{total}}(\{\Omega^{(i)}\})\\ &= \arg\min_{\Omega^{(i)}\succeq 0} \sum_{i=1}^{K} \left(-n_i \log\det(\Omega^{(i)}) + n_i < \Omega^{(i)}, \Sigma^{(i)} >\right)\\ &\quad + \left(\lambda_1\|\Omega^{(i)}\|_1 + \lambda_2\mathcal{R}(\Omega^{(1)}, \ldots, \Omega^{(K)})\right). \end{aligned} \tag{5}$$

A. MULTIPLE NONPARANORMAL GRAPHICAL MODELS FOR NON-I.I.D DATA (MULTI-NGMs)

In the real world, it is not ideal to assume that data always follow a Gaussian distribution. We introduce the nonparanormal distribution here to relax the normality assumption. A nonparanormal dataset $\mathbf{X}^{(i)}$ contains n_i independent observations of a p-dimensional random vector $\mathbf{Z}^{(i)} = (Z_1, Z_2, \ldots, Z_p)^\top$. There exist a set of univariate strictly increasing transformations $f^{(i)} = \{f_j^{(i)}\}_{j=1}^p$ such that:

$$f^{(i)}(\mathbf{Z}^{(i)}) = \left(f_1^{(i)}(Z_1), \ldots, f_p^{(i)}(Z_p)\right)^\top \sim N(\mu^{(i)}, \Sigma^{(i)}). \tag{6}$$

While the variable $\mathbf{Z}^{(i)}$ follows a nonparanormal distribution, the transformation functions make $f^{(i)}(\mathbf{Z}^{(i)})$ follow a Gaussian distribution. The remaining problem is to obtain graphs from data observations. It is impossible to estimate the covariance matrix $\Sigma^{(i)}$ directly in nonparanormal distribution. There is a mathematical relationship between the covariance matrix and correlation matrix $\mathbf{S}$ such that: $\Sigma = \text{diag}(\Sigma_i)\mathbf{S}\text{diag}(\Sigma_i)$. Thereby we have

$$\Sigma^{-1} = \text{diag}(\Sigma_i)^{-1}\mathbf{S}^{-1}\text{diag}(\Sigma_i)^{-1}, \tag{7}$$

where $\Sigma_i = \sqrt{\text{cov}(Z_i, Z_i)}$ and $\text{diag}(\Sigma_i) = \text{diag}(\Sigma_1, \ldots, \Sigma_p)$. As a result, the inverse of correlation matrix $\mathbf{S}^{-1}$ and the inverse of covariance matrix Σ^{-1} have the same nonzero and zero entries. In other words, $\mathbf{S}^{-1}$ and Σ^{-1} have the exact same sparsity pattern. Based on this observation, we can infer the graph structure by utilizing the correlation matrix $\mathbf{S}$ instead of Σ.

Therefore, an efficient nonparametric estimator [26] for the correlation matrix $\mathbf{S}$ has been established. To estimate

S, [26] uses the population Kendall's tau correlation coefficients τ_{jk}. So, given the above nonparanormal distribution $\mathbf{Z}^{(i)} \sim \mathrm{NPN}_p(\mu^{(i)}, \Sigma^{(i)}; f_1^{(i)}, \ldots, f_p^{(i)})$, its correlation matrix can be estimated as follows:

$$\hat{\mathbf{S}}_{jk} = \sin\left(\frac{\pi}{2}\hat{\tau}_{jk}\right), \tag{8}$$

where the Kendall's tau $\hat{\tau}_{jk}$ can be estimated as $\frac{1}{n(n-1)} \sum_{1 \le r \le r' \le n} \mathrm{sign}\left((X_{rj}^{(i)} - X_{r'j}^{(i)})(X_{rk}^{(i)} - X_{r'k}^{(i)})\right)$ and $X_{rj}^{(i)}$ denotes the j-th feature of the r-th sample in the dataset $\mathbf{X}^{(i)}$.

By replacing the covariance matrix $\Sigma^{(i)}$ with the estimated correlation matrix $\hat{\mathbf{S}}^{(i)}$, our objective function (5) can be formalized as follows:

$$\begin{aligned}
&\hat{\Omega}^{(1)}, \hat{\Omega}^{(2)}, \ldots, \hat{\Omega}^{(K)} \\
&\quad = \underset{\Omega^{(i)} \succeq 0}{\arg\min} \sum_{i=1}^{K} \left(-n_i \log\det(\Omega^{(i)}) + n_i < \Omega^{(i)}, \hat{\mathbf{S}}^{(i)} >\right) \\
&\qquad + \left(\lambda_1 \|\Omega^{(i)}\|_1 + \lambda_2 \mathcal{R}(\Omega^{(1)}, \ldots, \Omega^{(K)})\right) \qquad (9)
\end{aligned}$$

B. FEDERATED OPTIMIZATION OF THE FEDJEM

From (9) we observe that:

- When optimizing the multi-task objective function (9), we only use the correlation matrix of the data $\hat{\mathbf{S}}^{(i)}$. There is no data transmission between the local devices and the center server, so the risk of leaking the raw data are low.
- When optimizing $L(\Omega^{(i)}) = -\log\det(\Omega^{(i)}) + < \Omega^{(i)}, \hat{\mathbf{S}}^{(i)} >$, we only use a single precision matrix. As a result, $L(\Omega^{(i)})$ can be updated locally. However, optimizing the regularization function $\mathcal{R}_{\text{total}}(\{\Omega^{(i)}\}) = \lambda_1\|\Omega^{(i)}\|_1 + \lambda_2\mathcal{R}(\Omega^{(1)}, \Omega^{(2)}, \ldots, \Omega^{(K)})$ requires all updates communicated from the distributed devices. Therefore, $\mathcal{R}_{\text{total}}(\{\Omega^{(i)}\})$ should be updated globally in the center server.

Based on these two observations, we choose the alternating method of multipliers (ADMM) method to ensure that $L(\Omega^{(i)})$ and $\mathcal{R}_{\text{total}}(\{\Omega^{(i)}\})$ can be optimized locally and globally, respectively. Consequently, we introduce new variables $\{\Psi^{(i)}\}$ and add a group of constraints $\Omega^{(i)} = \Psi^{(i)}, i = 1, 2, \ldots, K$. The ADMM is used to design the FEDJEM for the federated joint estimator of multiple sUGMs. The objective function is as follows:

$$\begin{aligned}
&\{\hat{\Omega}^{(i)}\}, \{\hat{\Psi}^{(i)}\} \\
&\quad = \underset{\Omega^{(i)} \succeq 0, \Psi^{(i)} \succeq 0}{\arg\min} \sum_{i=1}^{K} \left(-n_i \log\det(\Omega^{(i)}) + n_i < \Omega^{(i)}, \hat{\mathbf{S}}^{(i)} >\right) \\
&\qquad + \left(\lambda_1 \|\Psi^{(i)}\|_1 + \lambda_2 \mathcal{R}(\Psi^{(1)}, \ldots, \Psi^{(K)})\right) \\
&\qquad \text{s.t. } \Omega^{(i)} = \Psi^{(i)}. \qquad (10)
\end{aligned}$$

The augmented Lagrangian function [27] of (10) is given by:

$$\begin{aligned}
&J(\{\Omega^{(i)}\}, \{\Psi^{(i)}\}, \{\mathbf{U}^{(i)}\}) \\
&= \sum_{i=1}^{K} \left(-n_i \log\det(\Omega^{(i)}) + n_i < \Omega^{(i)}, \hat{\mathbf{S}}^{(i)} >\right) \\
&\quad + \left(\sum_{i=1}^{K} \lambda_1 \|\Psi^{(i)}\|_1 + \lambda_2 \mathcal{R}(\Psi^{(1)}, \Psi^{(2)}, \ldots, \Psi^{(K)})\right) \\
&\quad + \frac{\rho}{2} \sum_{i=1}^{K} \|\Omega^{(i)} - \Psi^{(i)} + \mathbf{U}^{(i)}\|_F^2, \qquad (11)
\end{aligned}$$

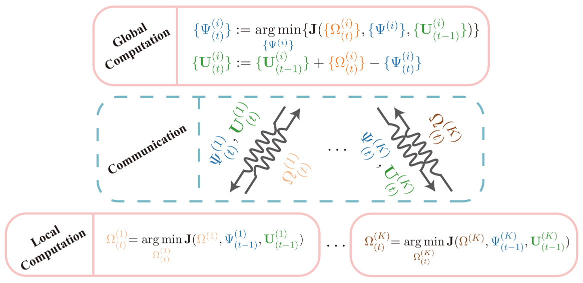
Global Computation
$\{\Psi_{(t)}^{(i)}\} := \underset{\{\Psi^{(i)}\}}{\arg\min}\{\mathbf{J}(\{\Omega_{(t)}^{(i)}\}, \{\Psi^{(i)}\}, \{\mathbf{U}_{(t-1)}^{(i)}\})\}$
$\{\mathbf{U}_{(t)}^{(i)}\} := \{\mathbf{U}_{(t-1)}^{(i)}\} + \{\Omega_{(t)}^{(i)}\} - \{\Psi_{(t)}^{(i)}\}$
Communication
$\Psi_{(t)}^{(1)}, \mathbf{U}_{(t)}^{(1)}$
$\Omega_{(t)}^{(1)}$
…
$\Psi_{(t)}^{(K)}, \mathbf{U}_{(t)}^{(K)}$
$\Omega_{(t)}^{(K)}$
Local Computation
$\Omega_{(t)}^{(1)} = \underset{\Omega_{(t)}^{(1)}}{\arg\min}\,\mathbf{J}(\Omega^{(1)}, \Psi_{(t-1)}^{(1)}, \mathbf{U}_{(t-1)}^{(1)})$
…
$\Omega_{(t)}^{(K)} = \underset{\Omega_{(t)}^{(K)}}{\arg\min}\,\mathbf{J}(\Omega^{(K)}, \Psi_{(t-1)}^{(K)}, \mathbf{U}_{(t-1)}^{(K)})$

FIGURE 3. Visualization of the algorithm 1.

where $\{\mathbf{U}^{(i)}\} = \{\mathbf{U}^{(1)}, \ldots, \mathbf{U}^{(K)}\}$ are dual variables.

At the t-th iteration, we can solve (11) as follows:

1) **Local update**:
$\{\Omega_{(t)}^{(i)}\} := \underset{\{\Omega^{(i)}\}}{\arg\min} \left\{J(\{\Omega^{(i)}\}, \{\Psi_{(t-1)}^{(i)}\}, \{\mathbf{U}_{(t-1)}^{(i)}\})\right\}$.
2) **Global update**:
$\{\Psi_{(t)}^{(i)}\} := \underset{\{\Psi^{(i)}\}}{\arg\min} \left\{J(\{\Omega_{(t)}^{(i)}\}, \{\Psi^{(i)}\}, \{\mathbf{U}_{(t-1)}^{(i)}\})\right\}$.
3) **Global update**:$\{\mathbf{U}_{(t)}^{(i)}\} = \{\mathbf{U}_{(t-1)}^{(i)}\} + \{\Omega_{(t)}^{(i)}\} - \{\Psi_{(t)}^{(i)}\}$.

The pseudo code of FEDJEM is summarized in Algorithm 1 and Fig. 3 is a visualization of the algorithm.

C. FEDERATED LOCAL UPDATE OF $\Omega^{(i)}$

Taking the derivative of (11) with respect to $\Omega^{(i)}$, we can update $\Omega^{(i)}$ as the minimizer of

$$\begin{aligned}
\hat{\Omega}^{(i)} &= \underset{\Omega^{(i)} \succeq 0}{\arg\min} -n_i \log\det(\Omega^{(i)}) + n_i < \Omega^{(i)}, \hat{\mathbf{S}}^{(i)} > \\
&\quad + \frac{\rho}{2}\|\Omega^{(i)} - \Psi^{(i)} + \mathbf{U}^{(i)}\|_F^2. \qquad (12)
\end{aligned}$$

Let the derivative be 0 to obtain (13):

$$\Omega^{(i)-1} - \frac{\rho}{n_i}\Omega^{(i)} = \hat{\mathbf{S}}^{(i)} - \frac{\rho}{n_i}\Psi^{(i)} + \frac{\rho}{n_i}\mathbf{U}^{(i)}. \tag{13}$$

The updated $\Omega^{(i)}$ can be represented as

$$\hat{\Omega}^{(i)} = \mathbf{V}\mathrm{diag}\left(\frac{n_i}{2\rho}\left(-D_{jj} + \sqrt{D_{jj}^2 + \frac{4\rho}{n_i}}\right)\right)\mathbf{V}^{\top}. \tag{14}$$

D_{jj} denotes the j-th diagonal element of the diagonal matrix $\mathbf{D}$ and $\mathbf{VDV}^{\top}$ denotes the eigendecomposition of $\hat{\mathbf{S}}^{(i)} - \frac{\rho}{n_i}\Psi^{(i)} + \frac{\rho}{n_i}\mathbf{U}^{(i)}$, the right side of (13).

Algorithm 1 FEDJEM

```
Data: Number of tasks $K$, original data $\{\mathbf{X}^{(i)}\}$,
      the maximum of iterations $T$, tuning
      parameters$\{\lambda_1, \lambda_2, \rho, \epsilon\}$.
1  initialize the model variables:
   $\Omega^{(i)} = \mathbf{I}, \Psi^{(i)} = \mathbf{0}, \mathbf{U}^{(i)} = 0$ for $i = 1, 2, \ldots, K$.
2  for iterations $t = 1$ to $T$ do
3      for tasks $i \in \{1, 2, \ldots, K\}$ in parallel over $K$ devices do
4          $\Omega^{(i)}_{(t)} := \arg\min_{\Omega^{(i)}} J(\Omega^{(i)}, \Psi^{(i)}_{(t-1)}, \mathbf{U}^{(i)}_{(t-1)})$
5      end
6      for variables $\{\Psi^{(i)}_{(t)}\}, \{\mathbf{U}^{(i)}_{(t)}\}$ in the center server do
7          $\{\Psi^{(i)}_{(t)}\} := \arg\min_{\{\Psi^{(i)}\}} \left\{J(\{\Omega^{(i)}_{(t)}\}, \{\Psi^{(i)}\}, \{\mathbf{U}^{(i)}_{(t-1)}\})\right\}$
8          $\{\mathbf{U}^{(i)}_{(t)}\} := \{\mathbf{U}^{(i)}_{(t-1)}\} + \{\Omega^{(i)}_{(t)}\} - \{\Psi^{(i)}_{(t)}\}$
9      end
10     if $\sum_{i=1}^{K} \frac{\|\Omega^{(i)}_{(t)} - \Omega^{(i)}_{(t-1)}\|_F}{\|\Omega^{(i)}_{(t)}\|_2} < \epsilon$ then
11         break;
12     end
13 end
Result: Precision matrices $\{\Omega^{(i)}\}$
```

D. FEDERATED GLOBAL UPDATE OF $\Psi^{(i)}$

Taking the derivative of (11) with respect to $\{\Psi^{(i)}\}$, we can update $\{\Psi^{(i)}\}$ as the minimizer of

$$\{\hat{\Psi}^{(i)}\} = \arg\min_{\{\Psi^{(i)}\}} \left(\sum_{i=1}^{K} \lambda_1 \|\Psi^{(i)}\|_1 + \lambda_2 \mathcal{R}(\Psi^{(1)}, \ldots, \Psi^{(K)})\right) + \frac{\rho}{2} \sum_{i=1}^{K} \|\Omega^{(i)} + \mathbf{U}^{(i)} - \Psi^{(i)}\|_F^2. \quad (15)$$

We choose the fused graphical lasso and group graphical lasso as our regularization function $\mathcal{R}(\cdot)$. In simplicity of the notations, we denote $\mathbf{B}^{(i)} = \Omega^{(i)} + \mathbf{U}^{(i)}$.

1) VARIATION I: FEDJEM-GROUP

If $\mathcal{R}(\cdot)$ is the group-2 norm of the parameter, we can plug it into (11). Then, the loss function $J(\{\Omega^{(i)}\}, \{\Psi^{(i)}\}, \{\mathbf{U}^{(i)}\})$ has the following formulation:

$$\begin{aligned} &J(\{\Omega^{(i)}\}, \{\Psi^{(i)}\}, \{\mathbf{U}^{(i)}\}) \\ &= \sum_{i=1}^{K} n_i L(\Omega^{(i)}) + \lambda_1 \sum_{i=1}^{K} \sum_{j \neq k} |\Psi^{(i)}_{jk}| \\ &\quad + \lambda_2 \sum_{j \neq k} \sqrt{\sum_{i=1}^{K} \Psi^{(i)2}_{jk}} + \frac{\rho}{2} \sum_{i=1}^{K} \|\Psi^{(i)} - \mathbf{B}^{(i)}\|_F^2. \end{aligned} \quad (16)$$

It follows that (16) is a linear system of $B^{(i)}_{jk}$ and $\Psi^{(i)}_{jk}$. By solving the linear system of $B^{(i)}_{jk}$, we can reach the solution of (16) as:

$$\hat{\Psi}^{(i)}_{jk} := S(B^{(i)}_{jk}, \frac{\lambda_1}{\rho}) \max\left(1 - \frac{\lambda_2}{\rho\sqrt{\sum_{i=1}^{K} S(B^{(i)}_{jk}, \frac{\lambda_1}{\rho})^2}}, 0\right), \quad (17)$$

where S denotes the soft-thresholding operator.

2) VARIATION II: FEDJEM-FUSED

If the $\mathcal{R}(\cdot)$ is generalized fused lasso penalty, then the loss function $J(\{\Omega^{(i)}\}, \{\Psi^{(i)}\}, \{\mathbf{U}^{(i)}\})$ has the following formulation:

$$\begin{aligned} &J(\{\Omega^{(i)}\}, \{\Psi^{(i)}\}, \{\mathbf{U}^{(i)}\}) \\ &= \sum_{i=1}^{K} n_i L(\Omega^{(i)}) + \lambda_1 \sum_{i=1}^{K} \sum_{j \neq k} |\Psi^{(i)}_{jk}| \\ &\quad + \lambda_2 \sum_{i<i'} \sum_{j,k} |\Psi^{(i)}_{jk} - \Psi^{(i')}_{jk}| + \frac{\rho}{2} \sum_{i=1}^{K} \|\Psi^{(i)} - \mathbf{B}^{(i)}\|_F^2. \end{aligned} \quad (18)$$

We can use an iterative solution to obtain the optimal value of $\Psi^{(i)}_{jk}$ in (18). In the case of $K = 2$ and $\lambda_1 = 0$, the solution to (18) is:

$$\begin{aligned} &\left(\hat{\Psi}^{(1)}_{jk}, \hat{\Psi}^{(2)}_{jk}\right) \\ &:= \begin{cases} \left(B^{(1)}_{jk} - \frac{\lambda_2}{\rho}, B^{(2)}_{jk} + \frac{\lambda_2}{\rho}\right) & \text{if } B^{(1)}_{jk} > B^{(2)}_{jk} + \frac{2\lambda_2}{\rho} \\ \left(B^{(1)}_{jk} + \frac{\lambda_2}{\rho}, B^{(2)}_{jk} - \frac{\lambda_2}{\rho}\right) & \text{if } B^{(2)}_{jk} > B^{(1)}_{jk} + \frac{2\lambda_2}{\rho} \\ \left(\frac{B^{(1)}_{jk} + B^{(2)}_{jk}}{2}, \frac{B^{(1)}_{jk} + B^{(2)}_{jk}}{2}\right) & \text{if } \left|B^{(1)}_{jk} - B^{(2)}_{jk}\right| \leq \frac{2\lambda_2}{\rho} \end{cases} \end{aligned} \quad (19)$$

IV. PRACTICAL CONSIDERATION

A. COMPUTATION ADAPTION ON LOCAL DEVICES

The whole computation cost can be split into three parts according to the three steps. The first step is to update $\Omega^{(1)}, \Omega^{(2)}, \ldots, \Omega^{(K)}$, the computational complexity of which is mainly decided by the eigendecomposition of K $p \times p$ matrices, $O(Kp^3)$ [28]. The second step is to update $\Psi^{(1)}, \Psi^{(2)}, \ldots, \Psi^{(K)}$ which involves some basic operations on the matrices. The computational complexity of this step is $O(Kp^2)$, as is that of the third step. Thus, the computational complexity of our method is $O(Kp^3)$.

B. COMMUNICATION ADAPTATION

The communication cost is determined by the size of the transmitted data. If we communicate the datasets $\{\mathbf{X}^{(i)}\}$ to the center server directly, the number of samples of $\mathbf{X}^{(i)}$ dramatically affects the communication efficiency [29]. Usually, the number of samples n_i is several times greater than the dimensions ($n_i \gg p$), which is extremely costly. In our method, we only exchange the updated variance matrices $\{\Omega^{(i)}\}, \{\Psi^{(i)}\}, \{\mathbf{U}^{(i)}\}$, which reduces the communication cost substantially.

C. CONVERGENCE GUARANTEE

We assume that the functions $L(\cdot)$ and $\mathcal{R}_{\text{total}}(\cdot)$ are closed, proper and convex, and that the augmented Lagrangian $J(\cdot)$ has a saddle point. According to the theorem in [27], if there exists an optimal point Ω^*, then $\Omega_{(t)} \to \Omega^*$ when $t \to \infty$.

D. DATA PRIVACY

We do not share the original datasets $\{\mathbf{X}^{(i)}\}$, but rather transfer the model updates between the edge devices and the center server. Thus, we hardly infer the real data from the updates so that the data remain private.

V. RELATED WORK

A. JOINT ESTIMATATION OF MULTI-sGGM

The joint graphical lasso is a technique for jointly estimating multiple graphical models from datasets belonging to different but related classes. This method is based on a penalized log likelihood approach, where the penalty function includes two parts. The first part is a lasso [30] that encourages the precision matrices to be sparse. The choice of the second penalty depends on the characteristics of the graphical models that we expect to be shared. We chose several relevant studies to support our work developing a joint graphical lasso in this study. The fused JGL [15], for example, uses a fused norm to encourage a shared pattern of sparsity (shared positions of zeros); the group JGL [15] uses a $\{\mathcal{G}, 2\}$ norm to encourage shared non-zero elements. Innovative penalties were also used by SIMONE [16], Node-based JGL [31], and some other researchers recently to capture special similarity among graphs.

B. CLIME FOR ESTIMATING SPARSE GAUSSIAN GRAPHICAL MODEL

The constrained ℓ_1 minimization method for inverse matrix estimation (CLIME) estimator can be used to estimate the precision matrix Ω via an ℓ_1 constrained optimization:

$$\begin{aligned} &\arg\min_{\Omega} \; \|\Omega\|_1 \\ &\text{s.t. } \|\Sigma\Omega - \mathbf{I}\|_\infty \le \lambda, \end{aligned} \tag{20}$$

where the tuning parameter $\lambda > 0$. CLIME can be solved column-by-column. We assume here that β is one of the column vectors in the precision matrix Ω. We can then estimate each column β of Ω as follows rather than estimating the entire Ω:

$$\begin{aligned} &\arg\min_{\beta} \; \|\beta\|_1 \\ &\text{s.t. } \|\Sigma\beta - \mathbf{e}_j\|_\infty \le \lambda. \end{aligned} \tag{21}$$

Finally, CLIME uses the following operation to maintain the symmetric property of the estimator:

$$\begin{aligned} \hat{\Omega}_{ij} = \hat{\Omega}_{ji} = \hat{\Omega}_{ij}\,\text{sign}\left(\max\left(\left|\hat{\Omega}_{ij}\right| - \left|\hat{\Omega}_{ji}\right|, 0\right)\right) \\ +\hat{\Omega}_{ji}\,\text{sign}\left(\max\left(\left|\hat{\Omega}_{ji}\right| - \left|\hat{\Omega}_{ij}\right|, 0\right)\right) \end{aligned} \tag{22}$$

C. SOLUTIONS TO CHALLENGES IN FEDERATED LEARNING

There are four core challenges inherent to federated learning and several current approaches available to mitigate them. **1) Expensive communication.** The federated network involves large amounts of devices and data, so the communication speed is restricted by limited resources. Current approaches focus on reducing either the number of communication rounds or the size of data for each round. The local updating method proposed in [32] allows for a variable to be applied on each device in parallel at each round. Compression schemes significantly reduce the size of transmitted data by forcing the updating models to be sparse, as was the case in [29], [33], [34]. **2) System heterogeneity.** The performance of different devices may differ due to hardware conditions, network connectivity, or power availability. Sometimes an active device may drop out at a certain iteration due to system problems like poor network connection, which exacerbates problems such as stragglers or fault tolerance. A practical strategy to deal with the device-dropping-out problem is to simply ignore such device failures, which may introduce bias toward devices [6]. System heterogeneity can also be managed using asynchronous communication [35], [36]. **3) Statistical heterogeneity.** The size of datasets may vary significantly between devices. We use the MOCHA [25] framework here to balance our datasets by controlling each device's optimization quality. Additionally, for each node, the data may be collected in a non-i.i.d. manner across the network, which contributes to an underlying statistical structure that captures the relationship among devices and their associated distributions [37], [38]. [39] focused on reducing the variance of the model performance across devices to obtain relative fairness beyond accuracy. **4) Privacy.** Potential private data leakage is a typical concern in federated learning. Though shared information has a gradient or is processed beyond raw data, it is still subject to leakage. Recently, researchers have used tools based on previous cryptographic protocols such as SMC to ensure data security [40], [41]; however, these tools sacrifice model performance and system efficiency. [42] applied differential privacy for global differential privacy, which makes a trade-off between security and model performance.

VI. EXPERIMENTS

We conduct three types of experiments as described in this section. We first evaluate the performance of the federated learning framework FEDJEM under different circumstances. We apply the local computation, global computation, and communication time of every iteration as our time metric to observe the balance capacity of our framework with different variables. We also implement our method on a simulation dataset to evaluate its performance. We draw the predicted neural connections in the brain according to the estimated precision matrix and compare it with the true brain connectome.

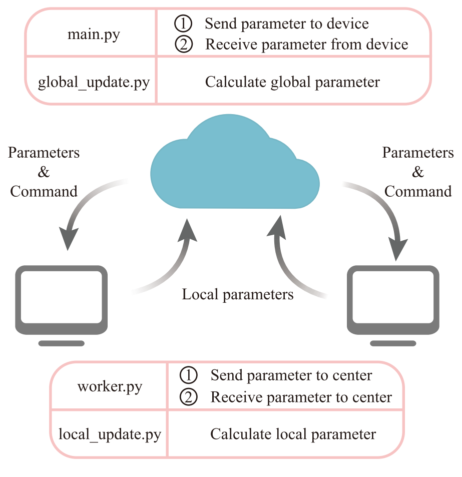
main.py
① Send parameter to device
② Receive parameter from device
global_update.py
Calculate global parameter
Parameters & Command
Parameters & Command
Local parameters
worker.py
① Send parameter to center
② Receive parameter to center
local_update.py
Calculate local parameter

FIGURE 4. Experimental implementation of proposed framework.

Next, we comprehensively compare FEDJEM and other baselines with respect to time, accuracy, energy consumption and privacy. We compare FEDJEM with other baselines and the total computation time when varying the feature dimension. We also compare the model performance by drawing FPR versus TPR curves for our method and baselines with both Gaussian and non-Gaussian data. We determine the energy consumption and raw data size in the transmission of JGL, GLasso, CLIME, and FEDJEM to show its advantages in both energy efficiency and privacy preservation.

Finally, we implement FEDJEM on a real-world dataset to predict the brain connectomes for further scientific research. The real-world dataset aggregates functional and structural brain imaging data of patients with Parkinson's disease. The dataset contains two tasks with 54 features and 103 samples in one task and 234 in the other.

A. EXPERIMENTAL SETTING

1) BASELINES

We compare FEDJEM with the following baselines: 1) The multi-task joint graphical lasso (JGL), 2) the single-task GLasso baseline (i.e., where each task uses GLasso independently); and 3) the single-task CLIME baseline (i.e., where each task uses CLIME independently).

2) EXPERIMENTAL ENVIRONMENT

We run our experiments on four servers with one dual-core 16 GB RAM, 40 GB cloud storage, and 5M bandwidth plus three single-core 4 GB RAM and 40 GB cloud storage. The center server is much more powerful than the local devices and the bandwidth is limited. This environment satisfies the conditions for federated learning.

3) IMPLEMENTATION

We implement a general federated learning framework for multi-task learning based on our method as well. Fig. 4 shows our federated learning framework with the function of every python file. We update the variable $\Omega^{(i)}$ in *local_update.py* and $\Psi^{(i)}$ and $\mathbf{U}^{(i)}$ in *global_update.py*. The updates of every iteration are transmitted between the cloud center and local devices through *main.py* and *worker.py*. Our code is available on https://github.com/MahjongGod-Saki/FEDJEM.

4) METRIC

a: PREDICTION ACCURACY

We use the edge-level false positive rate (FPR) and true positive rate (TPR) to measure the difference between the true and predicted graphs. Here, FPR $= \frac{\text{FP}}{\text{FP+TN}}$ and TPR $= \frac{\text{TP}}{\text{TP+FP}}$. The true positive (TP) and true negative (TN) values indicate the number of true non-zero entries and true-zero entries, respectively. We draw several FPR versus TPR curves to illustrate the performance of our method over a range of the regularization parameters λ_1, λ_2.

b: TIME

- Communication time: We record the time of parameter transmission between the local devices and global center in each iteration. We use this measurement to evaluate the communication costs of our method.
- Local computation time: In each iteration, we use the temporal costs of updating $\Omega^{(i)}$ to measure the local computation consumption.
- Global computation time: We determine the time it takes for the cloud center to update $\Psi^{(i)}$ as a measure of global computation cost.

5) HYPER-PARAMETER SELECTION

In this subsection, we discuss the effects of hyper-parameters, which play a crucial role in the convergence rate, sparsity, and accuracy of our experimental results.

a: λ_1

As the regularization parameter of ℓ_1 norm, λ_1 can control the sparsity of the precision matrix. A larger λ_1 leads to a sparser estimated network.

b: λ_2

As the regularization parameter of the group-2 penalty or fused lasso penalty, λ_2 encourages a similar pattern of sparsity across all the estimated precision matrices in the group lasso. It also controls the similarity of many elements among all the estimated precision matrices in the fused lasso. A larger λ_2 drives the edges across the estimated networks toward zero.

c: ρ

As the augmented Lagrangian parameter, tuning ρ determines the step size of every iteration. Our algorithm converges faster but performs worse when ρ is relatively large.

6) SIMULATION DATASETS

We conduct simulations to explore a three-class problem. We generate three networks corresponding to three classes, each consisting of c equally sized unconnected subnetworks with d features and a certain power degree distribution. Each network has $p = c \times d$ dimensions. Among the c subnetworks, the three networks share $(c - 2)$ subnetworks. Of the last two subnetworks, the first network has both, the second has one, and the third has none.

To satisfy the structure of the three networks, we first generate c covariance matrices with ones on the diagonal. The values on elements corresponding to edges obey a uniform distribution $U([-0.4, -0.1] \cup [0.1, 0.4])$. The values on elements not corresponding to edges equal zero. We could then obtain 1.5 times the sum of the absolute values of off-diagonal elements of each row and divide every off-diagonal element of this row. We average the matrix with its transpose, then add every diagonal element by 1.5 times the sum of the off-diagonal elements in its row. The strictly diagonally dominant matrix is positive definite in this case, so we obtain a positive-definite covariance matrix $\tilde{\Sigma}$. For the covariance matrices of the three networks,

$$(\Sigma_1)_{ij} = d_{ij}(\tilde{\Sigma}^{-1})_{ij} / \sqrt{(\tilde{\Sigma}^{-1})_{ii}(\tilde{\Sigma}^{-1})_{jj}}, \tag{23}$$

where $d_{ij} = 0.6$ if $i \neq j$ and $d_{ij} = 1$ if $i = j$. Resetting one of the c subnetwork blocks in Σ_1 to the identity yields the covariance matrix of the second class Σ_2. Resetting an additional subnetwork block to the identity yields Σ_3. For each class, we generate independent, identically distributed samples from a $N(\mathbf{0}, \Sigma_i)$, $i = 1, 2, 3$ distribution. We generate three categories of datasets $\{\mathbf{X}^{(i)}\}_{i=1}^{3}$, $\mathbf{X}^{(i)} \in \mathbb{R}^{n \times p}$ using the method described above. Then we design three groups of different distinct monotone increasing transformation functions $f^{(i)} = \{f_j^{(i)}\}_{j=1}^{p}$, $i = 1, 2, 3$. We obtain three datasets with different nonparanormal distributions $\{\mathbf{Y}^{(i)}\}_{i=1}^{3}$ via the transformations of the inverse functions: $Y_{jk}^{(i)} = {f_k^{(i)}}^{-1}(X_{jk}^{(i)})$, $j = 1, 2, \ldots, n$ and $k = 1, 2, \ldots, p$.

Note that d represents the feature dimension of every subnetwork rather than the whole network. We choose $d = 16, 32, 64, 128$ for comparison when $n = 160$ and $s = 0.01$ in the first type of dataset (i.e., $p = 160, 320, 640, 1280$). n represents the number of samples. We choose $n = d, 2d, 4d, 8d$ for comparison when $d = 32$ and $s = 0.01$ in the second type. The sparsity s is linearly positively related to the exponent of a power law distribution, so a larger s means a sparser network. We choose $s = 1.0, 0.5, 0.1, 0.01$ for comparison when $d = 128$ and $n = 150$ in the third type. The simulation datasets are shown in Table 2.

B. THE PERFORMANCE OF FEDJEM UNDER VARIOUS CIRCUMSTANCES

We use local computation, global computation, and communication time as three metrics to analyze different aspects of computational performance while varying the dimension d, number of samples n, and sparsity s of the simulation datasets as shown in Table 2. We conduct three series of experiments in total. By combining them, we find that changes in n, d, and s impact the final results in distinct ways as shown in Fig. 5.

TABLE 2. Simulation datasets for time consumption analysis. All simulated graphs in have 10 subnetworks (i.e. c = 10).

Simulation type	d	n	s
Varying d	16,32,64,128	150	0.01
Varying n	32	32,64,128,256	0.01
Varying s	128	150	1.0, 0.5, 0.1, 0.01

1) VARYING THE NUMBER OF FEATURES d

The first experiment on simulation datasets with varying d is conducted to observe the changes in computation and communication time as the feature dimension of the subnetwork d varies in $\{16, 32, 64, 128\}$. Fig. 5 (a)(b)(c) shows that communication takes the most time among the three processes when d is small [6], but local computation time is bottlenecked when d exceeds a certain threshold. Fig. 5 (j) shows that "local computation" is lower than "communication" at first but surpasses "communication" as d increases.

Fig. 5 (b) shows that global computation in the cloud center consistently consumes the least time among the three processes. This is because global computation involves only some basic matrix operations. Local computation, conversely, takes much more time than global computation due to its eigendecomposition operation in each iteration. Another point of concern is that available physical memory runs out rapidly as iteration increases when d is large. At a certain point, it is necessary to use SWaP space reserved in advance. This consumes extra time to read data from the hard disk and to save data into it. Therefore, Fig. 5 (b) presents a sharp increment with slight fluctuations as iterations exceed 180 when $d = 128$. Fig. 5 (c) suggests that network fluctuations result in a sudden change in communication time.

2) VARYING THE NUMBER OF SAMPLES n

The second simulation experiment with varying n is conducted to observe the changes in computation and communication time as the number of samples n varies in $\{d, 2d, 4d, 8d\}$. Fig. 5 (e)(f) shows that the number of samples does not influence the global computation or communication time. However, Fig. 5 (d) shows that more local computation time is consumed as n reaches a certain value, e.g., $n = 8d$. This is because in (13), $\hat{\mathbf{S}}^{(i)} - \frac{\rho}{n_i}\Psi^{(i)} + \frac{\rho}{n_i}\mathbf{U}^{(i)}$ is singular when n is small at the beginning of the algorithm. It takes less time to apply eigendecomposition to a singular matrix for local computation. When n is large, the $\hat{\mathbf{S}}^{(i)}$ is nonsingular initially. Therefore, the local computation spends a similar amount of time on each iteration. Due to our privacy-preserving strategy, the algorithm itself is irrelevant to the value of the samples. Our experimental results satisfy our expectations regarding the irrelevance between the number of samples and the implementation time.

3) VARYING THE SPARSITY s

The third experiment on simulation datasets with varying s is conducted to observe the changes in computation and

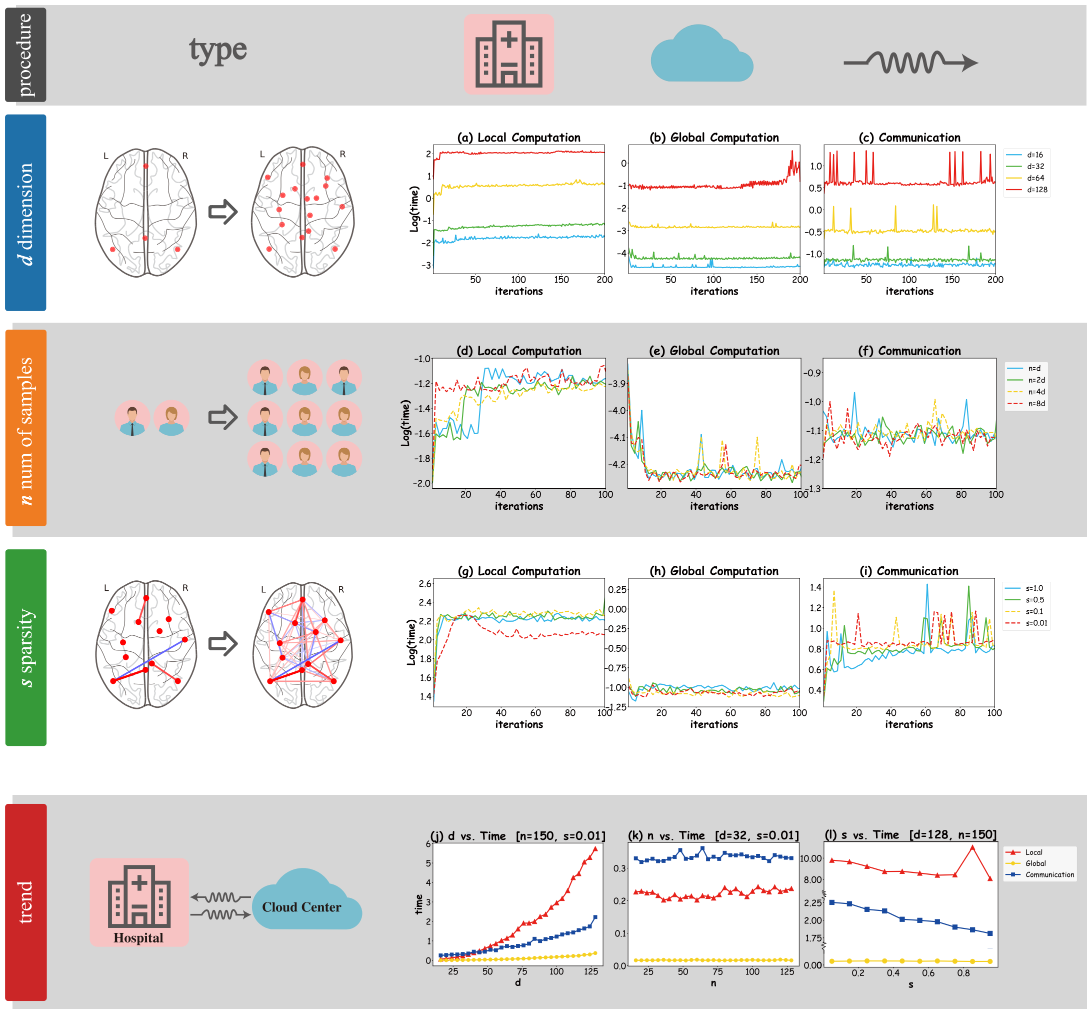

FIGURE 5. **Comparison of time in different domains. We mainly record three types of time cost, local computation time, global computation time, and communication time, when varying the different parameters of the simulation datasets. Subfigure (a)(b)(c) show iteration vs. time (in log-seconds) on three domains by varying the feature dimension of the subnetwork d in $\{16, 32, 64, 128\}$. Subfigure (d)(e)(f) show iteration vs. time (in log-seconds) on three domains by varing the number of samples in $\{d, 2d, 4d, 8d\}$. Subfigure (g)(h)(i) show iteration vs. time (in log-seconds) on three domains by varying the sparsity of the network in $\{1.0, 0.5, 0.1, 0.01\}$. Subfigure (j) shows d vs. time (in seconds) when $n = 150, s = 0.01$. Subfigure (k) shows n vs. time (in seconds) when $d = 32, s = 0.01$. Subfigure (l) shows s vs. time (in seconds) when $d = 128, n = 150$.**

communication time as the sparsity of the network s varied in $\{1.0, 0.5, 0.1, 0.01\}$. Sparsity s is linearly positively related to the exponent of a power law distribution in the data generation process. (We use power law distribution to mimic the structure of biological networks [43].) Namely, it is more likely to generate a dense network when s is relatively small. Fig. 5 (g) shows that estimating a denser network takes less local computation time without the format transmission operator of the sparse matrix representation. As expected, the curve of "$s = 0.01$" is much lower than other curves. Sparsity has little influence on global computation, but does increase the communication time when more dense parameters are transferred. Fig. 5 (i)(l) shows that sparse parameters with sparse representation reduce the size of data that needs to be communicated, which results in less communication time. However, the time cost returns to an average level quickly when $s = 1.0$, as shown in Fig. 5 (i). After several iterations, the parameters communicated to the center are not as sparse as ones in the beginning of the process. The communication time is markedly reduced initially and then increases to an average level as the parameters reach a certain level of sparsity.

4) THE PREDICTED BRAIN CONNECTOMES FROM THE SIMULATION DATASET

We also test our model's predictive power according to an estimated brain connectome. We choose 54 ROIs in the brain as target nodes. Using the data generation method described in Section VI-A6, we generate three 54-dimensional networks and 150 non-i.i.d. samples for each network. We draw three true brain images according to the networks, then implement our method on the three sets to infer the edges in order. Finally, we draw the brain connectome according to the prediction results. Fig. 6 shows a comparison between the

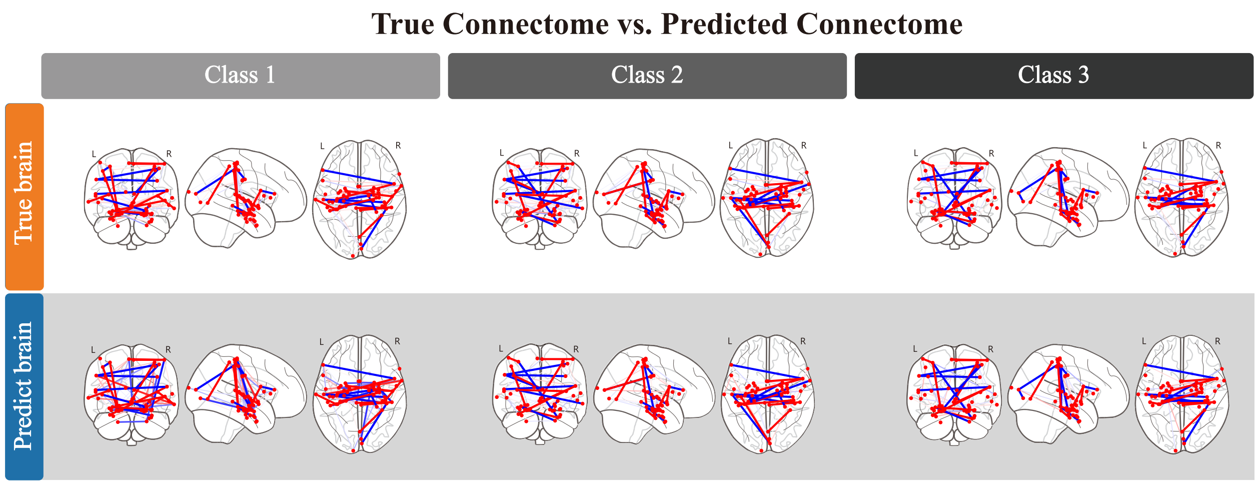

FIGURE 6. Comparison between true connectome and predicted connectome. We predicted three groups of predicted connectomes corresponding to the three networks of the simulation dataset.

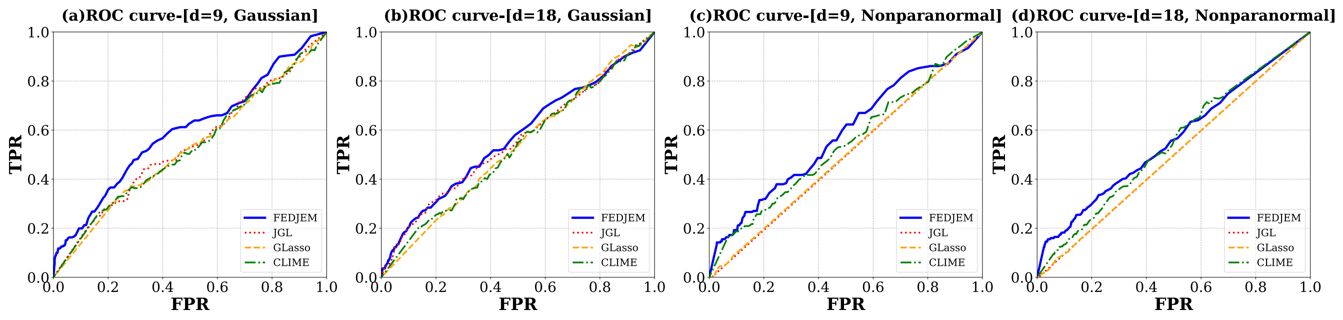

FIGURE 7. The FPR-TPR curve graph for different methods on four simulation datasets. Subfigure (a)(b) are for Gaussian simulation datasets with the feature dimension d varying in {9, 18}. Subfigure (c)(d) are for nonparanormal datasets with the feature dimension d varying in {9, 18}. We can see that curves from FEDJEM are above the single-sGGM estimators "GLasso" and "CLIME" and the multi-task sGGM estimator "JGL".

true and predicted brain connectomes. Most edges representing strong connections are predicted accurately. In addition, the predicted brain connectome of the three classes recovers some shared edges, which reflects correlation among the three tasks.

C. COMPARISON WITH BASELINES

We then conduct four categories of comparison experiments to evaluate the prediction accuracy, time consumption, energy efficiency, and privacy of the proposed method.

1) ACCURACY COMPARISON

To compare the prediction power of our estimator, we generate four simulation datasets with the number of samples $n = 150$, the number of the subnetwork $c = 6$. Two are simulated Gaussian data and the other two are non-Gaussian with the feature dimension of subnetwork d varying in $\{9, 18\}$ (i.e. $p = 54, 108$). Fig. 7(a)(b)(c)(d) shows the FPR-TPR curves of FEDJEM, JGL, GLasso, and CLIME on the simulation datasets. We draw four curves in each subfigure by tuning the regularization parameter λ_1 from 0.001 to 0.1 with a step size of 0.001. A larger area under the FPR-TPR curve represents better model performance. Fig. 7(a)(b) shows that our method obtains better curves than the other methods in the Gaussian cases. For non-Gaussian cases in Fig. 7(c)(d), our method achieves even better performance as it can effectively manage highly non-i.i.d heterogeneous data. Overall, the experimental results are consistent with our expectations in Section III.

2) COMPUTATION COST COMPARISON

We next compare the time consumption between FEDJEM and the baselines. We conduct one experiment over a series of simulation datasets to observe the computation time as the number of samples n changes in the set of $\{2, 10, 100, 1000, 10000\}$. Fig. 8 provides four different experimental results for computation time when the feature dimension of the subnetwork d varies in $\{16, 32, 64, 128\}$ (i.e. $p = 160, 320, 640, 1280$).

Fig. 8 (a)(b)(c)(d) shows that the time consumption of FEDJEM is consistently much lower than the three baselines. In addition, our method is not significantly affected by the number of samples n while the traditional multi-sGGM method JGL is very sensitive to it due to the increase in communication cost. The communication consumption of JGL makes the most contribution to the time cost when n increases, and thus becomes the dominant factor.

The four subfigures also show that the time consumption of all the methods increases substantially as the

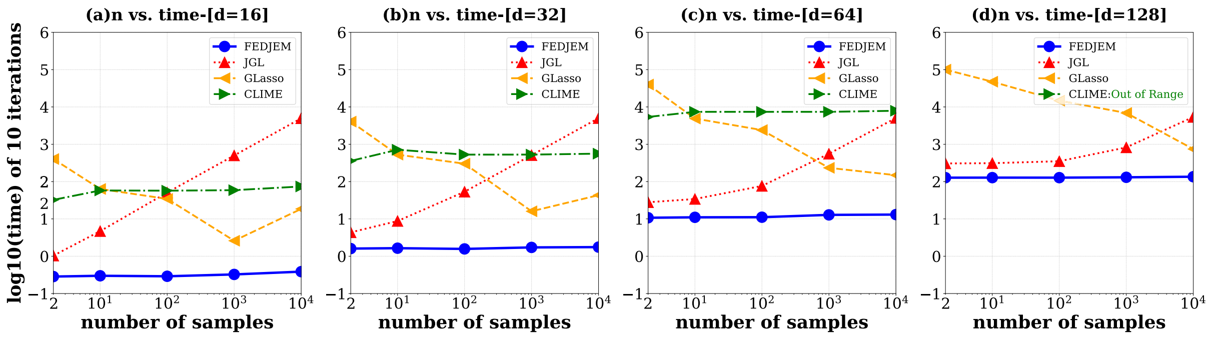

FIGURE 8. The number of samples n versus time for different methods on four simulation datasets. Subfigure (a)(b)(c)(d) show the number of samples n versus time of 10 iterations (in log10-seconds) by varying the feature dimension d in $\{16, 32, 64, 128\}$. Note that the values of the abscissa are the power of 10 except 2. The time of CLIME is not available in Subfigure (d). We can see that the time consumption of FEDJEM is much lower than the three baselines.

TABLE 3. Energy consumption and data in danger of proposed method and baselines. The proposed method shows the lowest energy cost without any raw data in transmission or the cloud center.

Methods	FEDJEM	JGL	GLasso	CLIME
Energy Cost (J)	<u>738.09</u>	1801.59	11966.27	>40000.00
Data (MB) in Danger	<u>0</u>	18.397	<u>0</u>	<u>0</u>

feature dimension increases. FEDJEM, however, is consistently less sensitive to d compared to the other three baselines. This is consistent with our theoretical computational analysis (Section IV). Overall, FEDJEM outperforms the baselines with faster computation, especially when the datasets are of large scale.

3) THE ENERGY CONSUMPTION AND PRIVACY COMPARISON

Finally, we compare our method and baselines with respect to energy consumption and privacy preservation. We select a dataset generating three networks with 768 features, 150 samples, and sparsity $s = 0.01$. To test energy consumption, we apply a package called "CodeCarbon" [44] to track the carbon emissions produced by all four methods and use joule units as the metric (i.e., "Energy Cost" in Table 3). To test privacy preservation, we select data transmitted from a local device to the global center as the evaluation metric (i.e., "Data in Danger" in Table 3). The results are shown in Table 3.[1] The first row of the table shows that our method consumes much less energy than the three baselines, which reflects the benefits of the federated learning framework. The second row of the table shows that our method has no data transferred to the cloud center while the JGL transfers 18.397MB. Therefore, FEDJEM has no risk of leaking private information.

[1]CLIME spends more than 90 min to finish running. Consequently, we record the energy consumption as larger than 40000.00J.

D. THE PREDICTED BRAIN CONNECTOMES FROM THE REAL-WORLD DATASETS

In the end, we apply FEDJEM on a real-world dataset consisting of functional and structural brain imaging data for Parkinson's disease patients. The dataset contains two tasks, one containing 103 samples of normal people and the other containing 234 samples of Parkinson's patients. Both share the same 54 features.

Fig. 9 shows the predicted brain connectome based on the real-world dataset. All connections can be separated into three groups, which are distinguishable in the figure by different colors. The results here align with our expectations. We find that some ROIs in the left and right sides of the brain are connected, which is generally accepted by researchers. We also identify some different edges between the two brain images that may be attributable to Parkinson's.

VII. DISCUSSION

As computation is distributed from one single cloud center to many local devices, our method releases large amount of computing power in the cloud center. This is a significant advantage, however, it necessitates communication during the distribution, which creates an extra step to properly balance communication and computation. There must be a trade-off [45] between local and global computation and communication time in any practical application of this method. If communication is more time-consuming than other processes, for instance, in cases where the network feature dimension is small or the true network is relatively sparse (e.g., Fig. 5), we suggest two potential solutions. 1) **Applying a special data structure**, such as sparse representation [46]–[48], during communication of model updates. 2) **Optimizing the communication network**, such as increasing the bandwidth or upgrading from 4G to 5G. It is also possible that local computation is the most time-consuming. For example, Fig. 5 shows that the local computation costs are unacceptable when the feature dimension is relatively large. We suggest three potential solutions: 1) **upgrading the local devices** with better computing power, 2) **adding an edge layer** between the local and global

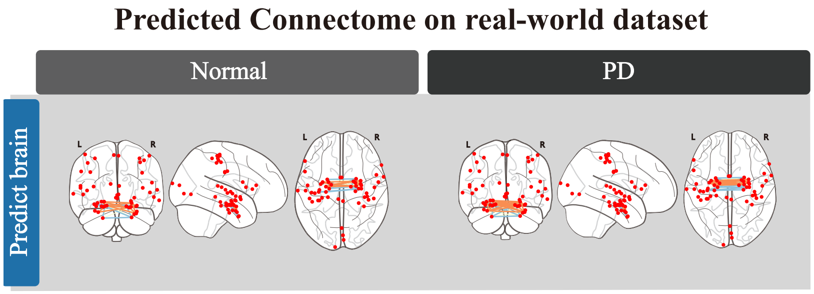

FIGURE 9. Two predicted connectomes of normal people and Parkinson's patients on real-world dataset.

computation as a three-layer federated learning framework, or 3) **optimizing the local updates**, such as by a fast decomposition solver [49], [50].

In the future, we will extend our method to extract other group patterns by applying different group regularization functions (i.e., different $\mathcal{R}_{\text{total}}$). Such penalty functions like group infinity norm group provide different multi-task patterns. We will then evaluate their performance and draw the conclusion about the usage criteria. In addition, based on our experimental results, we realize that we can have different implementation strategies for different device environments. We will establish an automatic mechanism to balance the three types of costs in the federated multi-task undirected graphical model as well.

VIII. CONCLUSION

In this study, we investigate a federated multi-task sUGM problem centered on learning multiple related graphs from many distributed and privacy-requiring resources in the context of neuroscience. We develop a novel method, FEDJEM that can make full use of non-i.i.d. heterogeneous data to jointly estimate the precision matrices with alternating global and local updates. Our method sufficiently uses the computing power of local devices, thus reducing the computational load acting on the cloud center while safeguarding private data. The method shows excellent performance in a series of experiments.

ACKNOWLEDGMENT

The authors are very grateful to our advisor, Dr. Beilun Wang, for his careful guidance of our writing skills. In addition, Xiao Tan thanks her mother for contacting her every day and preparing her dinner.

REFERENCES

[1] M. Schmidt, K. Murphy, G. Fung, and R. Rosales, "Structure learning in random fields for heart motion abnormality detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–8.

[2] W. H. Hsu, "Genetic wrappers for feature selection in decision tree induction and variable ordering in Bayesian network structure learning," *Inf. Sci.*, vol. 163, nos. 1–3, pp. 103–122, Jun. 2004.

[3] J. T. Vogelstein, Y. Park, T. Ohyama, R. A. Kerr, J. W. Truman, C. E. Priebe, and M. Zlatic, "Discovery of brainwide neural-behavioral maps via multiscale unsupervised structure learning," *Science*, vol. 344, no. 6182, pp. 386–392, Apr. 2014.

[4] M. J. Wainwright, P. Ravikumar, and J. D. Lafferty, "High-dimensional graphical model selection using ℓ_1-regularized logistic regression," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 19, 2007, p. 1465.

[5] L. Huang, Y. Yin, Z. Fu, S. Zhang, H. Deng, and D. Liu, "LoAdaBoost: Loss-based AdaBoost federated machine learning with reduced computational complexity on IID and non-IID intensive care data," *PLoS ONE*, vol. 15, no. 4, Apr. 2020, Art. no. e0230706.

[6] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, H. Brendan McMahan, T. Van Overveldt, D. Petrou, D. Ramage, and J. Roselander, "Towards federated learning at scale: System design," 2019, *arXiv:1902.01046*. [Online]. Available: http://arxiv.org/abs/1902.01046

[7] B. Wang, J. Gao, and Y. Qi, "A theoretical framework for robustness of (deep) classifiers against adversarial examples," 2016, *arXiv:1612.00334*. [Online]. Available: http://arxiv.org/abs/1612.00334

[8] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," 2018, *arXiv:1812.06127*. [Online]. Available: http://arxiv.org/abs/1812.06127

[9] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi, "Beyond inferring class representatives: User-level privacy leakage from federated learning," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, Apr. 2019, pp. 2512–2520.

[10] J. Xu, B. S. Glicksberg, C. Su, P. Walker, J. Bian, and F. Wang, "Federated learning for healthcare informatics," *J. Healthcare Informat. Res.*, vol. 5, no. 1, pp. 1–19, Mar. 2021.

[11] M. Yuan and Y. Lin, "Model selection and estimation in the Gaussian graphical model," *Biometrika*, vol. 94, no. 1, pp. 19–35, 2007.

[12] O. Banerjee, L. El Ghaoui, and A. d'Aspremont, "Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data," *J. Mach. Learn. Res.*, vol. 9, pp. 485–516, Mar. 2008.

[13] M. J. Sheller, G. A. Reina, B. Edwards, J. Martin, and S. Bakas, "Multi-institutional deep learning modeling without sharing patient data: A feasibility study on brain tumor segmentation," in *Proc. Int. MICCAI Brainlesion Workshop*. Granada, Spain: Springer, 2018, pp. 92–104.

[14] R. Caruana, "Multitask learning," *Mach. Learn.*, vol. 28, no. 1, pp. 41–75, 1997.

[15] P. Danaher, P. Wang, and D. M. Witten, "The joint graphical lasso for inverse covariance estimation across multiple classes," *J. Roy. Stat. Soc. Ser. B, Stat. Methodol.*, vol. 76, no. 2, p. 373, 2014.

[16] J. Chiquet, Y. Grandvalet, and C. Ambroise, "Inferring multiple graphical structures," *Statist. Comput.*, vol. 21, no. 4, pp. 537–553, Oct. 2011.

[17] J. Honorio and D. Samaras, "Multi-task learning of Gaussian graphical models," in *Proc. ICML*, 2010, pp. 1–8.

[18] J. Guo, E. Levina, G. Michailidis, and J. Zhu, "Joint estimation of multiple graphical models," *Biometrika*, vol. 98, no. 1, pp. 1–15, 2011.

[19] B. Zhang and Y. Wang, "Learning structural changes of Gaussian graphical models in controlled experiments," 2012, *arXiv:1203.3532*. [Online]. Available: http://arxiv.org/abs/1203.3532

[20] Y. Zhang and J. G. Schneider, "Learning multiple tasks with a sparse matrix-normal penalty," in *Proc. NIPS*, vol. 6, 2010, p. 2.

[21] Y. Zhu, X. Shen, and W. Pan, "Structural pursuit over multiple undirected graphs," *J. Amer. Statist. Assoc.*, vol. 109, no. 508, pp. 1683–1696, 2014.

[22] B. Wang, J. Gao, and Y. Qi, "A fast and scalable joint estimator for learning multiple related sparse Gaussian graphical models," in *Proc. Artif. Intell. Statist.*, 2017, pp. 1168–1177.

[23] B. Wang, A. Sekhon, and Y. Qi, "A fast and scalable joint estimator for integrating additional knowledge in learning multiple related sparse Gaussian graphical models," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 5161–5170.
[24] B. Wang and Y. Qi, "Fast and scalable learning of sparse changes in high-dimensional Gaussian graphical model structure," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2018, pp. 1691–1700.
[25] V. Smith, C.-K. Chiang, M. Sanjabi, and A. Talwalkar, "Federated multi-task learning," 2017, *arXiv:1705.10467*. [Online]. Available: http://arxiv.org/abs/1705.10467
[26] H. Liu, F. Han, M. Yuan, J. Lafferty, and L. Wasserman, "High-dimensional semiparametric Gaussian copula graphical models," *Ann. Statist.*, vol. 40, no. 4, pp. 2293–2326, Aug. 2012.
[27] S. Boyd, N. Parikh, and E. Chu, *Distributed Optimization and Statistical Learning Via the Alternating Direction Method of Multipliers*. Norwell, MA, USA: Now Publishers Inc, 2011.
[28] Z. Allen-Zhu and Y. Li, "Doubly accelerated methods for faster CCA and generalized eigendecomposition," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 98–106.
[29] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," 2016, *arXiv:1610.05492*. [Online]. Available: http://arxiv.org/abs/1610.05492
[30] R. Tibshirani, "Regression shrinkage and selection via the lasso: A retrospective," *J. Roy. Statist. Soc. B, Statist. Methodol.*, vol. 73, no. 3, pp. 273–282, 2011.
[31] K. Mohan, P. London, M. Fazel, D. Witten, and S. I. Lee, "Node-based learning of multiple Gaussian graphical models," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 445–488, Jan. 2014.
[32] S. U. Stich, "Local SGD converges fast and communicates little," 2018, *arXiv:1805.09767*. [Online]. Available: http://arxiv.org/abs/1805.09767
[33] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, "Robust and communication-efficient federated learning from Non-i.i.d. data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 9, pp. 3400–3413, Sep. 2020.
[34] Z. Yang, M. Chen, W. Saad, C. S. Hong, and M. Shikh-Bahaei, "Energy efficient federated learning over wireless communication networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 3, pp. 1935–1949, Mar. 2021.
[35] F. Niu, B. Recht, C. Re, and S. J. Wright, "HOGWILD!: A lock-free approach to parallelizing stochastic gradient descent," 2011, *arXiv:1106.5730*. [Online]. Available: http://arxiv.org/abs/1106.5730
[36] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 3, pp. 1205–1221, Jun. 2019.
[37] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-IID data," 2018, *arXiv:1806.00582*. [Online]. Available: http://arxiv.org/abs/1806.00582
[38] L. Lyu, H. Yu, and Q. Yang, "Threats to federated learning: A survey," 2020, *arXiv:2003.02133*. [Online]. Available: http://arxiv.org/abs/2003.02133
[39] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning," 2019, *arXiv:1905.10497*. [Online]. Available: http://arxiv.org/abs/1905.10497
[40] 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 *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2017, pp. 1175–1191.
[41] S. Truex, N. Baracaldo, A. Anwar, T. Steinke, H. Ludwig, R. Zhang, and Y. Zhou, "A hybrid approach to privacy-preserving federated learning," in *Proc. 12th ACM Workshop Artif. Intell. Secur. (AISec)*, 2019, pp. 1–11.
[42] A. Bhowmick, J. Duchi, J. Freudiger, G. Kapoor, and R. Rogers, "Protection against reconstruction and its applications in private federated learning," 2018, *arXiv:1812.00984*. [Online]. Available: http://arxiv.org/abs/1812.00984
[43] H. Chen and B. M. Sharp, "Content-rich biological network constructed by mining pubmed abstracts," *BMC Bioinf.*, vol. 5, no. 1, pp. 1–13, 2004.
[44] *Codecarbon [EB/OL]*. Accessed: 2021. [Online]. Available: https://github.com/mlco2/codecarbon#about-codecarbon-
[45] S. Zhang, A. Choromanska, and Y. LeCun, "Deep learning with elastic averaging SGD," 2014, *arXiv:1412.6651*. [Online]. Available: http://arxiv.org/abs/1412.6651
[46] J. Willcock and A. Lumsdaine, "Accelerating sparse matrix computations via data compression," in *Proc. 20th Annu. Int. Conf. Supercomput. (ICS)*, 2006, pp. 307–316.
[47] P. Jiang and G. Agrawal, "A linear speedup analysis of distributed deep learning with sparse and quantized communication," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 2530–2541.
[48] A. F. Aji and K. Heafield, "Sparse communication for distributed gradient descent," 2017, *arXiv:1704.05021*. [Online]. Available: http://arxiv.org/abs/1704.05021
[49] J. R. Bunch, L. Kaufman, and B. N. Parlett, "Decomposition of a symmetric matrix," *Numerische Math.*, vol. 27, no. 1, pp. 95–109, Mar. 1976.
[50] T. Hastie, R. Mazumder, J. D. Lee, and R. Zadeh, "Matrix completion and low-rank SVD via fast alternating least squares," *J. Mach. Learn. Res.*, vol. 16, pp. 3367–3402, Dec. 2015.

XIAO TAN is currently pursuing the degree with the School of Artificial Intelligence, Southeast University, China. Her research interests include federated learning, graphical model, and distribution optimization.

TIANYI MA is currently pursuing the degree with the School of Artificial Intelligence, Southeast University, China. His research interests include deep learning and model interpretability.

TONGTONG SU is currently pursuing the degree with the School of Artificial Intelligence, Southeast University, China. Her research interests include graphical models and federated learning.