Journal of Quality Technology

A Quarterly Journal of Methods, Applications and Related Topics

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/ujqt20

Federated generalized scalar-on-tensor regression

Elif Konyar & Mostafa Reisi Gahrooei

To cite this article: Elif Konyar & Mostafa Reisi Gahrooei (25 Sep 2023): Federated generalized scalar-on-tensor regression, Journal of Quality Technology, DOI: 10.1080/00224065.2023.2246600

To link to this article: https://doi.org/10.1080/00224065.2023.2246600

Published online: 25 Sep 2023.

JOURNAL OF QUALITY TECHNOLOGY
https://doi.org/10.1080/00224065.2023.2246600

Federated generalized scalar-on-tensor regression

Elif Konyar and Mostafa Reisi Gahrooei

Department of Industrial and Systems Engineering, University of Florida, Gainesville, Florida

ABSTRACT
Complex systems are generating more and more high-dimensional data for which tensor analysis showed promising results by capturing complex correlation structures of data. Such data is often distributed among various sites creating challenges for developing data-driven models. Specifically, data privacy and security concerns have been exacerbated in recent years and drove the demand to store and analyze data at the edge of networks rather than sharing it with a centralized server. Federated learning frameworks have been introduced as a solution to these concerns. These frameworks allow local clients to learn local models and collaborate with others to develop a more generalizable aggregated model while handling data privacy issues. In this article, we propose a federated generalized scalar-on-tensor regression framework where multiple local tensor models are learned at the edge, and their parameters are shared with and updated by an aggregator. Experiments on synthetic data sets and two real-world data sets from agriculture and manufacturing domains show the superiority of our approach over several benchmarks.

KEYWORDS
aggregated model; federated learning; personalized model; scalar-on-tensor regression

1. Introduction

Modern complex systems are equipped with numerous sensors, generating data in various forms such as waveform signals, images, and videos. Such data is often high-dimensional (HD) with complex structures that cannot be adequately represented by vectors. Due to the capability of tensors in preserving the structure and capturing the between and within-correlation structures of HD data, methods that use tensors for data representation have become popular (Zhou, Li, and Zhu 2013; Hoff 2015; Yu and Liu 2016; Fang, Paynabar, and Gebraeel 2019). For instance, scalar-on-tensor regression (SoTR) methods are developed to predict a scalar given an input tensor (Zhou, Li, and Zhu 2013; Guhaniyogi, Qamar, and Dunson 2017; Fang, Paynabar, and Gebraeel 2019; He and Zhang 2020). These tensor regression methods gained a significant traction in various applications, including healthcare (Zhou, Li, and Zhu 2013; Fan and Reimherr 2017), manufacturing (Fang, Paynabar, and Gebraeel 2019; Yan, Paynabar, and Pacella 2019; Gahrooei et al. 2021), and agriculture (Ogden et al. 2002; Li et al. 2020a) systems, where high dimensional data is common. For example, in agriculture, SoTR may be used to predict the health condition (e.g., healthy or not) of plants through imaging data collected by unmanned aerial vehicles (UAVs) equipped with hyperspectral cameras (Abdulridha et al. 2020). In another example, SoTR can be used in the detection of faulty vehicle engines using multichannel profiles that are generated by several sensors in engines and represented by tensors (Pacella 2018; Gahrooei et al. 2021). The main challenge of tensor models is that the number of parameters to be estimated is generally significantly larger than the sample size. To overcome this challenge, tensor decomposition techniques such as rank-R decomposition are incorporated into tensor regression models (Zhou, Li, and Zhu 2013; Lock 2018; Gahrooei et al. 2021).

In addition, complex systems are often decentralized. Therefore, the HD data they generate is generally distributed over multiple local sites, each with a small sample size. In the previously mentioned agriculture example, the hyperspectral images may be collected at multiple fields with potentially slightly different weather and soil conditions. Or, in the engine fault detection example, multiple factories may be involved in data collection for fault detection modeling. Unfortunately, often each local site creates a model in silo (at each local site) based on the data obtained from that site, ignoring the data generated in other

CONTACT Mostafa Reisi Gahrooei mreisigahrooei@ufl.edu Department of Industrial and Systems Engineering, University of Florida, Gainesville, Florida.

similar locations. This lack of collaboration results in models with limited generalization power. A potential solution to address this challenge is that each local site shares its local data with a server (e.g., cloud), which will construct potentially more accurate global models due to its access to the shared data. However, this framework creates huge communication and storage costs, makes the network susceptible to possible adversary attacks during communication of raw data, and leads to data privacy concerns since raw data may contain sensitive information that data owners are unwilling to share (McMahan et al. 2017; Yang et al. 2019; Li et al. 2020c). Therefore, there is a need to develop generalizable models for distributed HD data. The underlying question of this article is how local sites can collaborate with each other such that they benefit from HD data generated, stored, and processed in other sites without sharing their data. In an attempt to address this question, this article develops a federated scalar-on-tensor regression framework.

The idea of federated learning goes back to privacy-preserving methodologies (Du, Han, and Chen 2004). However, the term "federated learning" initiated with Federated Averaging (FedAvg) algorithm, which is an iterative stochastic gradient descent (SGD) based approach (McMahan et al. 2017). In this framework, each local client (site) stores local data, applies SGD to update its local model, and shares the gradient updates with a central server. This system is coordinated by a central server, which aggregates local gradients and sends the global update back to the local clients. These local and global update stages continue sequentially until global updates converge. This algorithm serves as the basis for many other federated learning studies (Kontar et al. 2021).

One of the main challenges in federated learning is that data sets stored in local sites are often statistically heterogeneous (i.e., they are not identically distributed). Therefore, aggregating local models from those sites may construct an aggregated model with reduced performance compared to models trained locally (Li et al. 2018) due to potentially negative transfer of information. Furthermore, the statistically heterogeneous characteristic may negatively affect the convergence of the aggregated model (Li et al. 2018). Some existing studies address the statistical heterogeneity by incorporating regularization on the parameter set in the local objective functions to improve generalization. One example is FedProx algorithm (Li et al. 2018) that adds a quadratic regularizer term to local objectives. Similarly, FedDyn (Acar et al. 2021) incorporates a dynamic regularizer into the local empirical risk functions, and SCAFFOLD (Karimireddy et al. 2020) tackles statistical heterogeneity, termed as client-drifts, by adding control variates to the local objective functions. In addition, some studies deal with statistical heterogeneity in terms of fairness to achieve more uniform solutions across local clients (Li et al. 2020b, 2021; Yue, Nouiehed, and Kontar 2021; Pillutla et al. 2022). For instance, Ditto regularizes local models such that they are close to a global model by jointly solving local and global models to achieve fairness and personalization (Li et al. 2021). Moreover, Pillutla et al. (2022) propose a partial personalization scheme to improve fairness, and Yue, Nouiehed, and Kontar (2021) designed GIFAIR-FL that weights local objective functions dynamically to attain a fair solution. The aforementioned frameworks are focused on generic optimization problems mostly suitable for models under low-dimensional settings such as logistic regression with the objective of achieving a reasonable aggregated model. Unfortunately, these studies do not consider high-dimensional settings, particularly when tensor data is available.

A handful of federated tensor modeling techniques have recently been introduced. For instance, Kim et al. (2017) propose a federated tensor factorization approach for computational phenotyping which preserves patient-level data privacy. Moreover, Ma et al. (2019) introduce a privacy-preserving tensor factorization approach that incorporates collaborative learning using heterogeneous electronic health records. Furthermore, Gao et al. (2021) propose a Tucker decomposition-based federated tensor decomposition approach for feature extraction. Nevertheless, the focus of these studies is on unsupervised tensor decomposition and therefore, are not suitable for tensor regression problems.

In this article, we propose a federated generalized scalar-on-tensor (FGSoT) regression framework where the input is a tensor and the output is a scalar. In this framework, multiple local sites collaborate in constructing an aggregated tensor model. In doing so, each site trains a tensor regression model at the edge and shares the model parameters with a central server. Then, the central server aggregates the shared parameters and broadcasts aggregated parameters back to each local site. These two steps continue sequentially, as it is illustrated in Figure 1. We estimate model parameters by minimizing novel local and global objective functions. To deal with the statistical heterogeneity of local data, we incorporate proximity regularization terms to control the deviation of local parameters from the global parameters. We further personalize local models by incorporating group lasso penalties on model

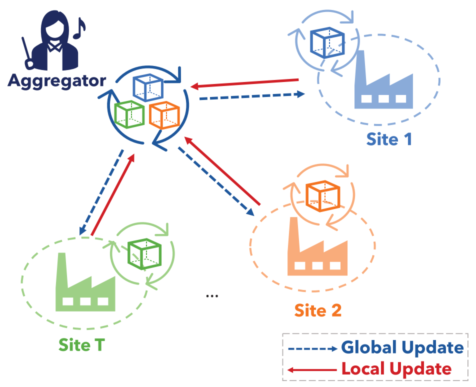

Figure 1. An overview of a federated network for tensor data.

parameters. Finally, to address the high-dimensionality of the problem and following the original SoTR (Zhou, Li, and Zhu 2013), we impose rank-R decomposition on the tensor of model parameters.

The rest of the article is organized as follows: In Section 2, we introduce our notation and review relevant tensor algebra. In Section 3, we review the generalized scalar-on-tensor regression framework. Next, we describe the proposed methodology in detail and discuss model parameter estimation algorithms in Section 4. In Section 5, we evaluate the performance of the proposed method with three different simulation scenarios. In Section 6, we explain two case studies and validate the performance of the proposed approach using real data sets related to agriculture and manufacturing applications. In Section 7, we provide our discussion on the proposed framework. Lastly, we conclude in Section 8.

2. Preliminaries and tensor notation

In this section, we introduce our notation and review some tensor algebra concepts (Kolda and Bader 2009). We use lowercase letter for scalars, e.g., x, bold lowercase letters for vectors, e.g., $\mathbf{x}$, bold capital letters for matrices, e.g., $\mathbf{X}$, and capital calligraphic letters for tensors, e.g., $\mathcal{X}$. For example, we denote an order-n tensor by $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \ldots \times I_n}$ where the dimensions of each mode i is I_i $(i = 1, \ldots, n)$. Furthermore, the mode-j matricization of a tensor $\mathcal{X}$, denoted by $\mathcal{X}_{(j)} \in \mathbb{R}^{I_j \times I_1 \ldots I_{j-1} I_{j+1} \ldots I_n}$, is a matrix whose columns are mode-j fibers (tensor subarrays with all but the j^{th} index fixed) of $\mathcal{X}$. Moreover, the trace of a matrix $\mathbf{X}$ is denoted as $\mathrm{Tr}(\mathbf{X})$. $\mathbf{X} \otimes \mathbf{Y} \in \mathbb{R}^{km \times ln}$ stands for *Kronecker product* of two matrices, $\mathbf{X} \in \mathbb{R}^{k \times l}$ and $\mathbf{Y} \in \mathbb{R}^{m \times n}$, and is obtained by:

$$\boldsymbol{X} \otimes \boldsymbol{Y} = \begin{bmatrix} x_{11}\boldsymbol{Y} & \ldots & x_{1l}\boldsymbol{Y} \\ \vdots & \ddots & \vdots \\ x_{k1}\boldsymbol{Y} & \ldots & x_{kl}\boldsymbol{Y} \end{bmatrix}.$$

Furthermore, if $l = n$, then *Khatri-Rao product* of $\mathbf{X}$ and $\mathbf{Y}$ is obtained by columnwise Kronecker products as $\mathbf{X} \odot \mathbf{Y} = [\mathbf{x}_1 \otimes \mathbf{y}_1 \ldots \mathbf{x}_l \otimes \mathbf{y}_l] \in \mathbb{R}^{km \times l}$. Besides, *outer product* of n vectors, $\mathbf{x}_1, \ldots, \mathbf{x}_n, \mathbf{x}_i \in \mathbb{R}^{I_i}$ $(i = 1, \ldots, n)$ is a tensor in $\mathbb{R}^{I_1 \times I_2 \ldots \times I_n}$, and is denoted as $\mathbf{x}_1 \circ \mathbf{x}_2 \circ \ldots \circ \mathbf{x}_n$. Each element of the outer product is calculated as $(\mathbf{x}_1 \circ \mathbf{x}_2 \circ \ldots \circ \mathbf{x}_n)_{j_1, j_2 \ldots j_n} = \prod_{k=1}^{n} x_{k, j_k}$, where x_{k, j_k} is the j_k^{th} element of $\mathbf{x}_k$. In addition, the *inner product* of two tensors, $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \ldots \times I_n}$, is calculated as $\langle \mathcal{X}, \mathcal{Y} \rangle = \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_n=1}^{I_n} x_{i_1 i_2 \cdots i_n} y_{i_1 i_2 \cdots i_n}$, where $x_{i_1 i_2 \cdots i_n}$ is an element $(i_1, i_2, \ldots, i_n)$ of $\mathcal{X}$.

Lastly, we explain *rank-R decomposition*, also known as CANDECOMP/PARAFAC (CP) decomposition. CP decomposes a tensor, $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_n}$, as $\mathcal{X} = \sum_{r=1}^{R} \mathbf{x}_1^{(r)} \circ \ldots \circ \mathbf{x}_n^{(r)}$, where $\mathbf{x}_i^{(r)} \in \mathbb{R}^{I_i}$, $(i = 1, \ldots, n)$. Alternatively, we represent this decomposition as $\mathcal{X} = [[\mathbf{X}_1, \ldots, \mathbf{X}_n]]$ where $\mathbf{X}_i \in \mathbb{R}^{I_i \times R}$ is a *factor matrix* whose columns are $\mathbf{x}_i^{(r)}$ $(r = 1, \ldots, R)$.

3. Review of generalized scalar-on-tensor regression

In this section, we review the generalized scalar-on-tensor (GSoT) regression framework proposed by Zhou, Li, and Zhu (2013). Assume we have a scalar target, y_i, $(i = 1, \ldots, N)$, and associated with that, an input tensor, $\mathcal{X}_i \in \mathbb{R}^{I_1 \times I_2 \ldots \times I_m}$. Assume, given the input, y_i follows an exponential family distribution with canonical parameters θ_i and dispersion parameter ϕ, i.e., $y_i | \mathcal{X}_i \sim f(y_i; \theta_i, \phi) = \exp\left(\frac{y_i \theta_i - b(\theta_i)}{a(\phi)} + c(y_i, \phi)\right)$, where $b(.), a(.)$ and $c(.)$ are distribution-specific known functions. Then, the GSoT regression defines a generalized linear tensor model (GLM) as follows: $g(E[y_i | \mathcal{X}_i]) = \mu + \langle \mathcal{B}_0, \mathcal{X}_i \rangle + e_i$, where $\mathcal{B}_0 \in \mathbb{R}^{I_1 \times I_2 \ldots \times I_m}$ is the true parameter tensor, and e_i is independent and identically distributed random noise with zero mean and constant variance. One solution to estimating the model parameters given a set of training data is obtained by vectorizing the parameter and input tensors, and solving the GLM model using available vector-based methods. However, the huge number of parameters $(1 + \prod_{d=1}^{m} I_d)$ may lead to an overfitting problem. To address this challenge, Zhou, Li, and Zhu (2013) proposed imposing a rank-R decomposition constraint on the parameter tensor to reduce the dimensionality. The solution to this problem is obtained by maximizing the log-likelihood with low-rankness constraint as follows:

$$\max_{\mu, \mathcal{B}} l(\mu, \mathcal{B}) = \sum_{i=1}^{N} \frac{y_i \theta_i - b(\theta_i)}{a(\phi)} + \sum_{i=1}^{N} c(y_i, \phi) \quad [1]$$
$$s.t.\ \mathcal{B} = [[\boldsymbol{U}_1, \ldots, \boldsymbol{U}_m]],$$

where $\theta_i = \mu + \langle \mathcal{B}, \mathcal{X}_i \rangle$, $\mathbf{U}_i \in \mathbb{R}^{I_i \times R}$ is the factor matrix associated with mode i, and R is the rank of tensor $\mathcal{B}$. The low rank constraint reduces the number of parameters to $1 + \sum_{d=1}^{m} I_d$. The alternating least square (ALS) approach can be used to solve (1).

4. Federated generalized scalar-on-tensor regression

In this section, we describe our proposed federated generalized scalar-on-tensor regression framework. Our method consists of local and global update stages. The aggregator initiates the process by broadcasting the model structure and the model rank to each site. Next, in the local update stage, each site trains a generalized scalar-on-tensor regression model by minimizing its own local objective function and shares the model parameters with the aggregator. In the global update stage, the aggregator constructs an aggregated model by minimizing a global objective function. This updating procedure is illustrated in Figure 1. Let us assume T sites are collaborating in constructing models. Each site has access to a set of training data $\mathbb{D}_t = \{y_{i,t}, \mathcal{X}_{i,t}\}_{i=1}^{N_t}$, where N_t is the sample size of the local data in the t^{th} $(t = 1, \ldots, T)$ site.

4.1. Local update mechanism

We assume the data at each local site t follows an exponential family distribution with canonical parameters $\theta_{i,t}$ as follows:

$$y_{i,t} | \mathcal{X}_{i,t} \sim f(y_{i,t}; \theta_{i,t}, \phi_t) = \exp\left(\frac{y_{i,t}\theta_{i,t} - b(\theta_{i,t})}{a(\phi_t)} + c(y_{i,t}, \phi_t) \right), \qquad [2]$$

where $y_{i,t}$ is the scalar target of the i^{th} sample at local site t, $(i=1, \ldots, N_t, t=1, \ldots, T)$. Moreover, $\theta_{i,t} = \mu_t + \langle \mathcal{B}_t, \mathcal{X}_{i,t} \rangle$ is the canonical parameter of an exponential family distribution, $\mathcal{B}_t \in \mathbb{R}^{I_1 \times I_2 \ldots \times I_m}$ is the parameter tensor, $\mathcal{X}_{i,t} \in \mathbb{R}^{I_1 \times I_2 \ldots \times I_m}$ is the input tensor (with m modes), and μ_t is the intercept. Furthermore, $b(.)$, $a(.)$ and $c(.)$ are distribution-specific known functions. In addition, it is known that $b'(\theta_{i,t}) = E[y_{i,t} | \mathcal{X}_{i,t}]$, where $b'(.)$ is the first derivative of $b(.)$, and E[.] is the expected value. While estimating the mean of $y_{i,t}$, we assume dispersion parameter, ϕ_t, is constant. Since $a(.)$ and $c(.)$ depend on the dispersion parameter, we rewrite a simplified version of the distribution suitable for the mean estimation as $y_{i,t} | \mathcal{X}_{i,t} \sim f(y_{i,t}; \theta_{i,t}) = \exp(y_{i,t}\theta_{i,t} - b(\theta_{i,t}))$.

With these assumptions and given a training data, each site trains a local tensor model by solving the following minimization problem:

$$\begin{aligned} \min_{\mu_t, \mathcal{B}_t} l(\mu_t, \mathcal{B}_t) = \sum_{i=1}^{N_t} \big(& -y_{i,t}(\mu_t + \langle \mathcal{B}_t, \mathcal{X}_{i,t} \rangle) \\ & + b(\mu_t + \langle \mathcal{B}_t, \mathcal{X}_{i,t} \rangle) \big) + N_t \sum_{d=1}^{m} \sum_{r=1}^{R_t} \frac{\alpha_t}{2} \|\boldsymbol{u}_{d,t}^{(r)}\|_2 \\ & + N_t \sum_{d=1}^{m} \sum_{r=1}^{R_t} \frac{\psi}{2} \|\boldsymbol{u}_{d,t}^{(r)} - \boldsymbol{v}_d^{(r)}\|_2^2 + N_t \frac{\delta}{2} (\mu_t - \zeta)^2 \\ s.t. \quad \mathcal{B}_t = \sum_{r=1}^{R_t} & \boldsymbol{u}_{1,t}^{(r)} \circ \boldsymbol{u}_{2,t}^{(r)} \cdots \circ \boldsymbol{u}_{m,t}^{(r)}, \end{aligned} \qquad [3]$$

where local parameters are $\mathbf{u}_{d,t}^{(r)} \in \mathbb{R}^{I_d}$ and μ_t, which are the r^{th} column of the d^{th} factor matrix $\mathbf{U}_{d,t} \in \mathbb{R}^{I_d \times R_t}$ and the model intercept for site t, respectively. Furthermore, $\mathbf{v}_d^{(r)} \in \mathbb{R}^{I_d}$ and $\zeta \in \mathbb{R}$ are the global parameters estimated by the aggregator and are known at this stage, and R_t is the rank of $\mathcal{B}_t$. In this formulation, we impose rank-R decomposition to address the high dimensionality of the tensor of parameters. In the rest of the article, when the context is clear, we will use R instead of R_t to represent the rank of a site.

The first term of the objective function in (3) minimizes the negative log-likelihood of the training data. The second term is the group lasso penalty for the personalization of the local models. To be more specific, in GSoT regression models, one must choose the rank (R) of the tensor of parameters, which is often done by minimizing a criterion, such as the Bayesian Information Criterion (BIC). However, in a federated setting, multiple local sites are available, and each site trains its own tensor regression model and may have to choose a different rank from other sites. In that situation, aggregating local models remains a challenge as it requires one specific rank. To overcome this issue, one global rank is decided by the aggregator, then each local site personalizes that rank through the second term in the objective function that uses $l_{2,1}$ group sparsity-inducing penalties. Through this term, local sites eliminate some redundant patterns locally by setting some columns of $\mathbf{U}_{d,t}$ to zero vectors as illustrated in Figure 2. Note that when no prior knowledge of the initial rank is available, the aggregator can initialize it as a large value and allow it to decrease over iterations until it converges. An example of the effect of the initial rank and the convergence of the global rank is provided in Figure 1 in Supplementary Results (Section 1). Finally, the third and fourth terms control the deviations of local model parameters from global parameters. These proximity regularization terms prevent the divergence of local models from the aggregated model. Furthermore, they allow the construction of models between the fully aggregated and fully localized models. Adding these terms is essential, especially when data in

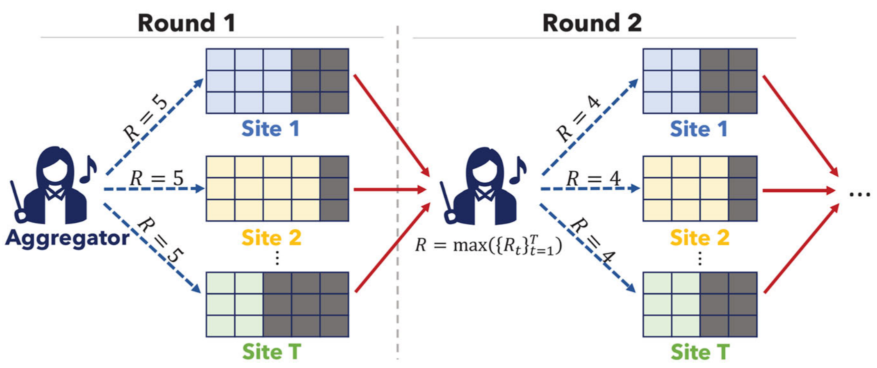

Figure 2. Rank selection mechanism. In factor matrices, gray shaded columns indicate zero vectors.

local sites are heterogeneous. Lastly, we impose a low rank constraint on the local parameter tensor, $\mathcal{B}_t$.

To solve this problem, we propose using the alternating least square (ALS) algorithm with the Newton's search method which has a quadratic convergence property. Note that other first/second order search methods can also be applied (See Section 7 for more details). First, we transform the inner product as $\langle \mathcal{B}_t, \mathcal{X}_{i,t}\rangle = \mathrm{Tr}(\mathbf{U}_{d,t}^\top \mathcal{X}_{i,(d),t}\mathbf{K}_{d,t})$, where $\mathbf{K}_{d,t} = \mathbf{U}_{m,t} \odot \ldots \mathbf{U}_{d+1,t} \odot \mathbf{U}_{d-1,t} \ldots \odot \mathbf{U}_{1,t}$. Furthermore, we denote $\mathbf{W}_{i,d,t} = [\mathbf{w}_{i,d,t}^{(1)}, \ldots, \mathbf{w}_{i,d,t}^{(R)}] := \mathcal{X}_{i,(d),t}\mathbf{K}_{d,t}, \mathbf{w}_{i,d,t}^{(r)} \in \mathbb{R}^{I_d}$. We then let $g_{i,d,t} := \mathrm{Tr}(\mathbf{U}_{d,t}^\top \mathbf{W}_{i,d,t})$, and $g_{i,d,t}^{(-r)} := \mathrm{Tr}(\mathbf{U}_{d,t}^{(-r)^\top} \mathbf{W}_{i,d,t}^{(-r)})$, where $\mathbf{U}_{d,t}^{(-r)}$ and $\mathbf{W}_{i,d,t}^{(-r)}$ are obtained by removing r^{th} columns of $\mathbf{U}_{d,t}$ and $\mathbf{W}_{i,d,t}$. Given these equations and definitions, we convert our objective function in (3) into:

$$\begin{aligned} f_t(\mu_t, \boldsymbol{u}_{d,t}^{(r)}) = &\sum_{i=1}^{N_t} (-y_{i,t}(\mu_t + g_{i,d,t}^{(-r)} + \boldsymbol{u}_{d,t}^{(r)^\top} \boldsymbol{w}_{i,d,t}^{(r)}) \\ &+ b(\mu_t + g_{i,d,t}^{(-r)} + \boldsymbol{u}_{d,t}^{(r)^\top} \boldsymbol{w}_{i,d,t}^{(r)})) \\ &+ N_t \sum_{d=1}^{m} \sum_{r=1}^{R} \frac{\alpha_t}{2} ||\boldsymbol{u}_{d,t}^{(r)}||_2 \\ &+ N_t \sum_{d=1}^{m} \sum_{r=1}^{R} \frac{\psi}{2} ||\boldsymbol{u}_{d,t}^{(r)} - \boldsymbol{v}_d^{(r)}||_2^2 \\ &+ N_t \frac{\delta}{2}(\mu_t - \zeta)^2. \end{aligned} \quad [4]$$

As the first step, we minimize (4) for $\mathbf{u}_{d,t}^{(r)}$ $(d = 1, \ldots, m;\ r = 1, \ldots, R)$. We fix the remaining parameters μ_t, $\{\mathbf{U}_{d',t}\}_{d'=1, d'\neq d}^m$, and $\{\mathbf{u}_{d,t}^{(r')}\}_{r'=1, r'\neq r}^R$, and we apply the Newton's search method. Given that $\mathbf{u}_{d,t}^{(r)} \neq \mathbf{0}$, we update $\mathbf{u}_{d,t}^{(r)}$ with ${}_*\mathbf{u}_{d,t}^{(r)} = \mathbf{u}_{d,t}^{(r)} - \gamma\ H(\mathbf{u}_{d,t}^{(r)})^{-1} G(\mathbf{u}_{d,t}^{(r)})$, where $G(.)$ is the gradient, $H(.)$ is the Hessian matrix, and γ is the step size. The gradient and the Hessian matrix can be found in Appendix A. On the other hand, if $|| \frac{1}{N_t} \sum_{i=1}^{N_t} (\mathbf{w}_{i,d,t}^{(r)} y_{i,t} - \mathbf{w}_{i,d,t}^{(r)} b'(\mu_t + g_{i,d,t}^{(-r)})) + \psi \mathbf{v}_d^{(r)}||_2 < \frac{\alpha_t}{2}$ holds, then $\mathbf{u}_{d,t}^{(r)} = \mathbf{0}$, which means the corresponding column of $\mathbf{U}_{d,t}$ becomes zero vector. Hence, this condition serves as rank selection for local models. Note that when one $\mathbf{u}_{d,t}^{(r)}$ in a factor matrix becomes zero, then the vectors in the r^{th} column of other factor matrices are equated to zero, i.e., $\mathbf{u}_{1,t}^{(r)} = \mathbf{u}_{2,t}^{(r)} = \ldots = \mathbf{u}_{m,t}^{(r)} = \mathbf{0}$. We repeat this procedure until convergence, i.e., when either $||{}_*\mathbf{u}_{d,t}^{(r)} - \mathbf{u}_{d,t}^{(r)}||_2$ becomes less than a threshold, or the maximum number of iterations is reached. To select γ, we employ the backtracking line search method. After updating $\mathbf{u}_{d,t}^{(r)}$ with this procedure, we repeat these steps for the remaining modes and factors for each t.

Next, we minimize (4) for the model intercept, μ_t, and employ the Newton's search method by following a similar idea with $\mathbf{u}_{d,t}^{(r)}$ update. The details of the update procedure are provided in Appendix A.

4.2. Global update mechanism

In this step, local model parameters are collected from local sites and aggregated globally. To aggregate local parameters, the aggregator solves

$$\min_{\zeta, \mathbf{v}_d^{(r)}} \sum_{t=1}^{T} f_t(\zeta, \boldsymbol{v}_d^{(r)}) \quad [5]$$

where $f_t(.)$ is the local objective function (4) for t. This function is a mixture of local objective functions. First, we solve (5) for $\mathbf{v}_d^{(r)}$ $(d = 1, \ldots, m;\ r = 1, \ldots, R)$ and obtain a closed-form solution as $\mathbf{v}_d^{(r)} = \frac{\sum_{t=1}^T N_t \mathbf{u}_{d,t}^{(r)}}{\sum_{t=1}^T N_t}$. Second, we solve (5) for ζ, and we obtain $\zeta = \frac{\sum_{t=1}^T N_t \mu_t}{\sum_{t=1}^T N_t}$. Thus, global parameters $\mathbf{v}_d^{(r)}$ and ζ are weighted averages of the corresponding local parameters $\mathbf{u}_{d,t}^{(r)}$ and μ_t. Figure 3 illustrates and Algorithm 1 summarizes our framework.

4.3. Identifiability

In tensor models, two main sources of nonidentifiability are scaling and permutation indeterminacy (Zhou, Li, and Zhu 2013). Note that the objective function (4) is convex

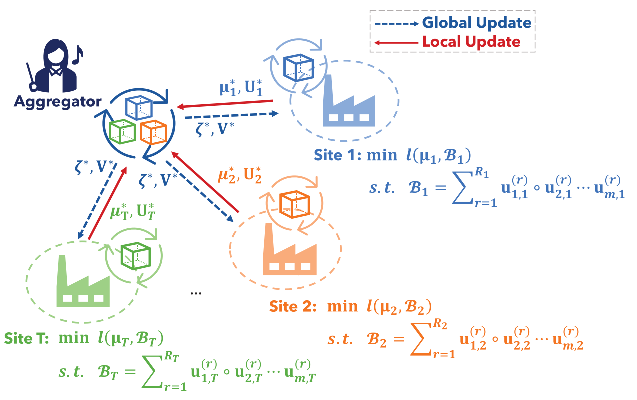
Global Update
Local Update
Aggregator
μ_1^*, U_1^*
ζ^*, V^*
Site 1: min $l(\mu_1, \mathcal{B}_1)$
s.t. $\mathcal{B}_1 = \sum_{r=1}^{R_1} u_{1,1}^{(r)} \circ u_{2,1}^{(r)} \cdots u_{m,1}^{(r)}$
ζ^*, V^*
ζ^*, V^*
μ_2^*, U_2^*
μ_T^*, U_T^*
Site 2: min $l(\mu_2, \mathcal{B}_2)$
s.t. $\mathcal{B}_2 = \sum_{r=1}^{R_2} u_{1,2}^{(r)} \circ u_{2,2}^{(r)} \cdots u_{m,2}^{(r)}$
Site T: min $l(\mu_T, \mathcal{B}_T)$
s.t. $\mathcal{B}_T = \sum_{r=1}^{R_T} u_{1,T}^{(r)} \circ u_{2,T}^{(r)} \cdots u_{m,T}^{(r)}$

Figure 3. Overview of federated generalized scalar-on-tensor (FGSoT) regression.

with respect to $\mathbf{u}_{d,t}^{(r)}$ given μ_t (and vice versa) in each estimation step. Hence, the solution to the minimization of (4) is unique, which addresses the scaling indeterminacy problem. The second source of nonidentifiability is the permutation indeterminacy: $\mathcal{B}_t = [[\mathbf{U}_{1,t}, \dots, \mathbf{U}_{m,t}]] = [[\mathbf{U}_{1,t}\Pi, \dots, \mathbf{U}_{m,t}\Pi]]$ where Π is a R-by-R permutation matrix. To solve the permutation indeterminacy of the tensor parametrization with $l_{2,1}$ norm penalty, we consider a permutation rule, Π, that rearranges the nonzero elements of the first row of $\mathbf{U}_{d,t}$ (i.e., $\mathbf{U}_{m,t,1}^{(r)} \neq 0$) in descending order such that $\mathbf{U}_{m,t,1}^{(1)} > \mathbf{U}_{m,t,1}^{(2)} > \dots > \mathbf{U}_{m,t,1}^{(R-y)}$ and leaves the zero elements at the end of the ordering. Here, y is the number of zero elements. After each local update stage for each site t, we employ this rule to permute the factor matrices. The details about the identifiability can be found in Appendix B.

Algorithm 1: Federated Generalized Scalar-on-Tensor (FGSoT) Regression

```
Input: $\psi_0$: initial $\psi$, $\delta_0$: initial $\delta$
Data: $\{\{y_{i,t}, \mathcal{X}_{i,t}\}_{i=1}^{N_t}\}_{t \in Sites}$ : target vectors and input tensors
1 for $t = 1, \dots, T$ do
2     Set $\psi := 0, \delta := 0$, and solve problem (3) for initial $\{\mathbf{u}_{d,t}^{(r)}\}_{\forall d,r}$ and $\mu_t$
3 end
4 Set $\psi := \psi_0$ and $\delta := \delta_0$
5 while convergence criterion not met do
6     Solve (5) for global parameters, $\{\mathbf{v}_d^{(r)}\}_{\forall d,r}$ and $\zeta$, and send them to sites
7     for $t = 1, \dots, T$ do
8         Solve (3) to update $\{\mathbf{u}_{d,t}^{(r)}\}_{\forall d,r}, \mu_t$, and send them to the aggregator
9     end
10 end
Output: $\{\mathbf{v}_d^{(r)}\}_{\forall d,r}$, $\zeta$, $\{\mathbf{u}_{d,t}^{(r)}\}_{\forall d,r,t}$, $\{\mu_t\}_{\forall t}$
```

4.4. Computational and space complexity

In this section, we study the computational and space complexities of the proposed method. Let us start with the computational complexity of the global update. Denote $\mathbf{V}_d = [\mathbf{v}_d^{(1)}, \dots, \mathbf{v}_d^{(R)}]$. The complexity of computing $\mathbf{V}_d$ is $O(RT\sum_{d=1}^m I_d + R(T-1)\sum_{d=1}^m I_d + R\sum_{d=1}^m I_d + T)$, which reduces to $O(RT\sum_{d=1}^m I_d)$. The complexity of computing ζ is $O(T + T + 1)$, which reduces to $O(T)$. Thus, the computational complexity of the global update step is $O(RT\sum_{d=1}^m I_d)$.

In the local step, the complexity of computing $\mathbf{K}_{d,t} = \mathbf{U}_{m,t} \odot \dots \mathbf{U}_{d+1,t} \odot \mathbf{U}_{d-1,t} \dots \odot \mathbf{U}_{1,t}$ is $O(R\Pi_{i=1, i\neq d}^m I_i)$. The complexity to compute $\mathbf{W}_{i,d,t} = \mathcal{X}_{i,(d),t}\mathbf{K}_{d,t}$ is $O(R\Pi_{i=1}^m I_i)$. The complexity of calculating $G(\mathbf{u}_{d,t}^{(r)})$ is $O(N_t((R-1)^2 I_d + R\Pi_{i=1}^m I_i))$. Furthermore, computing $H(\mathbf{u}_{d,t}^{(r)})$ requires $O(N_t((R-1)^2 I_d + R\Pi_{i=1}^m I_i + 2I_d^2))$ steps. The inverse calculation of $H(\mathbf{u}_{d,t}^{(r)}) \in \mathbb{R}^{I_d \times I_d}$ with standard Gauss-Jordan elimination requires $O(I_d^3)$ steps. This can be reduced to $O(\frac{1}{3}I_d^3)$ with Cholesky decomposition for $I_d > 1$. Therefore, the complexity of single Newton's step is $O(N_t((R-1)^2 I_d + R\Pi_{i=1}^m I_i + 2I_d^2) + \frac{1}{3}I_d^3)$. Assume we perform $n_{d,t}^{(r)}$ steps to update $\mathbf{u}_{d,t}^{(r)}$ for site t. Then, the complexity of local update for a single site, t, is $O(\sum_{r=1}^R \sum_{d=1}^m n_{d,t}^{(r)}(N_t((R-1)^2 I_d + R\Pi_{i=1}^m I_i + 2I_d^2) + \frac{1}{3}I_d^3))$. Note that the Newton's search method is not integral to the proposed method. Depending on the dimensionality of the model, other methods such as the quasi-Newton method might give better computational complexity (See Section 7 for more details).

Note that we are not sending the tensor $\mathcal{B}$ to the central server. Instead, we share the factor matrices, U_d $(d = 1, \dots, m)$, with the central server. Let us look at the space complexity of the transferred data to quantify how much data is transferred per iteration. The space complexity of decomposed $\mathcal{B}$ with rank-R decomposition is $O(R\sum_{d=1}^m I_d)$. Transferring $\mathcal{B}$ rather than the factor matrices would require $O(\Pi_{d=1}^m I_d)$ space. Therefore, using decomposition techniques such as rank-R decomposition reduces the data transfer and the space requirements substantially.

4.5. Selection of tuning parameters

Our proposed approach involves the selection of a few hyperparameters: ψ, δ, and $\{\alpha_t\}_{t=1}^T$. We set ψ and δ based on a sensitivity analysis as described in Section 5.2. To select $\alpha_t, (t = 1, \dots, T)$, which is the parameter associated with the group lasso penalty, we employ the Bayesian Information Criterion (BIC). Bayesian

Information Criterion (BIC) is commonly used for model selection in tensor analysis (Zhou, Li, and Zhu 2013; Fang, Paynabar, and Gebraeel 2019; Roy and Michailidis 2022). Especially, Zhou, Li, and Zhu (2013) showed the efficacy of BIC for rank selection in scalar-on-tensor regression models with rank-R decomposition. The BIC is calculated as $-2l(\alpha_t) + \log(N_t)p$, where $-l(.)$ is the loss in (4) and N_t is the number of samples in site t. Also, p denotes the effective number of parameters, which is computed as $p = R(I_1 + I_2) - R^2$, when $m=2$, and as $p = R(\sum_{d=1}^{m} I_d - m + 1)$, when $m > 2$. The terms $-R^2$ and $R(-m+1)$ are included to adjust the number of parameters for scaling indeterminacy for rank-R decomposition as discussed in (Zhou, Li, and Zhu 2013).

4.6. Implementation details

We adopt the termination criteria of *FedProx* (Li et al. 2018) to design the termination mechanisms for *FGSoT*, which include convergence check, divergence check, and the number of iterations check. The update procedure continues until one of these conditions holds: (i) $\left|\overline{l^{(k)}} - \overline{l^{(k-1)}}\right| < \epsilon$, where $\overline{l^{(k)}}$ is the average log-likelihood over all sites at k^{th} round (convergence check), (ii) $\overline{l^{(k)}} - \overline{l^{(k-z)}} > \tau$ (divergence check), or (iii) the maximum number of iterations, K_g, is reached. The parameters ϵ and K_g are user-specified and can be determined based on the user's computational resources. The parameters of the divergence check, τ and z, are set based on the approach in (Li et al. 2018). Note that we assume local sites share the log-likelihood information (computed in (4)) as well as the model parameters.

5. Performance evaluation using simulation

This section evaluates the performance of the proposed method by using several simulated data sets. In these simulation studies, we seek to answer the following two questions: First, *How well does an aggregated model, constructed by the collaboration of multiple sites, perform when applied to (i) test data in the sites involved in the collaboration and to (ii) data in new sites which did not participate in collaboration?* Second, *How does the performance of the aggregated model change when statistical heterogeneity exists in data?*

5.1. Data generation

For each local site t, we generate $\{(y_{i,t}, \mathcal{X}_{i,t})\}_{i=1}^{N_t}$ where $y_{i,t} \in \mathbb{R}$ is the target and $\mathcal{X}_{i,t} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ is an input tensor. The underlying data generation model is: $y_{i,t} = 1$ if $\pi(\mathcal{B}_t, \mathcal{X}_{i,t}) \geq 0.5$ and 0 otherwise, where $\pi(\mathcal{B}_t, \mathcal{X}_{i,t}) = \frac{1}{1+e^{-k\langle \mathcal{B}_t, \mathcal{X}_{i,t}\rangle + e_i}}$, e_i is independent and identically distributed random noise which follows $\mathcal{N}(0, \sigma_e^2)$, and k is a multiplier to scale $\langle \mathcal{B}_t, \mathcal{X}_{i,t}\rangle$.

Generation of $\mathcal{X}_{i,t}$: To generate an input data, $\mathcal{X}_{i,t}$, we simulate a vector, $\mathbf{x}_{i,t} \in \mathbb{R}^{I_1 I_2 I_3}$, from a multivariate normal distribution, $\mathbf{x}_{i,t} \sim \mathcal{N}(\mathbf{h}_t, \Sigma)$. Following the idea proposed in (Li et al. 2018), we simulate $\mathbf{h}_t$ for each site t from a normal distribution with mean zero and standard deviation of $\beta_{\mathcal{X}}$, i.e., $\mathbf{h}_t \sim \mathcal{N}(0, \beta_{\mathcal{X}}^2)$. Furthermore, we generate the $(i,j)^{th}$ entry of the covariance matrix Σ as follows: $\Sigma_{i,j} = \rho^{|i-j|}$. After simulating $\mathbf{x}_{i,t}$, we reshape it to obtain $\mathcal{X}_{i,t}$. An important remark is that $\beta_{\mathcal{X}}$ is a measure of statistical heterogeneity of data stored in local sites.

Generation of $\mathcal{B}_t$: To simulate the model parameters, $\mathcal{B}_t \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ for each site, we generate factor matrices $\mathbf{U}_{d,t} \in \mathbb{R}^{I_d \times R}$, where R is the rank of $\mathcal{B}_t$. Specifically, factor matrices have the following form: $\mathbf{U}_{d,t} = \mathbf{U}_d^{global} + \mathbf{U}_{d,t}^{site}$. In other words, each factor matrix has a global component and a site-specific component, which serves as statistical heterogeneity added to the local model. Each entry of $\mathbf{U}_d^{global}$ follows standard normal distribution, $\mathcal{N}(0,1)$. The site-specific component is simulated element-wise from a normal distribution, $\mathcal{N}(p_t, \sigma_u^2)$, where p_t is the mean corresponding to site t and follows $\mathcal{N}(0, \beta_{y\sim\mathcal{X}}^2)$. Therefore, $\beta_{y\sim\mathcal{X}}$ is a measure of statistical heterogeneity in local models. We alter $\beta_{y\sim\mathcal{X}}$ to change the level of statistical heterogeneity in local models. We also control the level of heterogeneity by deciding the number of factor matrices with a site-specific component. That is, we may include site-specific components to all factor matrices or to m_{het} of them. This approach simulates the situations where heterogeneity exists along a subset of modes of $\mathcal{B}_t$. To generate homogeneous local models, we let $\mathbf{U}_{d,t} = \mathbf{U}_d^{global}$ for all modes. The data generation procedure is illustrated in Figure 4. In this figure, $m_{het} = 2$.

In all the simulation scenarios, we assume the same level of heterogeneity exists in local data, i.e., $\beta_{\mathcal{X}} = 0.1$. We test the effect of heterogeneity in models by choosing m_{het} from $\{0,1,2,3\}$, and $\beta_{y\sim\mathcal{X}}$ from $\{0.05, 0.10, 0.15, 0.20, 0.25\}$. Note that $m_{het} = 0$ means that true models are homogeneous, and as m_{het} and $\beta_{y\sim\mathcal{X}}$ increase, the level of heterogeneity increases.

5.2. Simulation specifications

We evaluate the performance of the proposed model, *FGSoT*, using three different simulation studies. In addition to *FGSoT* with proximity regularization, we

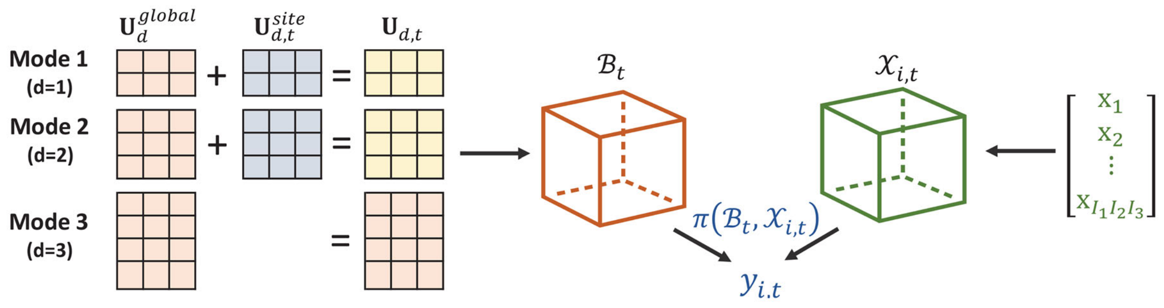

Figure 4. Data generation procedure.

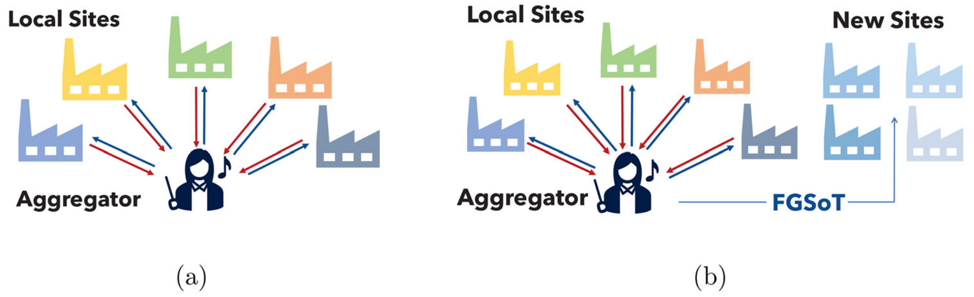

Figure 5. Simulation scenarios (a) Scenarios I & II, (b) Scenario III.

evaluate the performance of the proposed approach without this regularization, meaning that we do not penalize the deviations between local and global parameters ($\psi = \delta = 0$). This approach is denoted as $FGSoT - NR$.

In the first simulation scenario, we consider the situation where true local models show homogeneous behavior, i.e., $m_{het} = 0$. In the second and third simulation scenarios, we evaluate the performance of the model under the statistical heterogeneity of underlying models ($m_{het} > 0$). In Scenarios I and II, we divide data in each local site into training and test sets. The aggregator constructs an *aggregated model* by combining local models trained in local sites ($FGSoT_{agg}$ and $FGSoT - NR_{agg}$). Local sites further *personalize* the aggregated model by performing one more stage of local updates ($FGSoT_{per}$ and $FGSoT - NR_{per}$). Benchmarks in this setting are *local model*, where each local site learns its own model without any collaboration, and $FGSoT - NR$. Finally, we evaluate performances of all methods on the test data of local sites. In Scenario III, we further evaluate the performance of $FGSoT$ on new local sites which have not participated in collaboration. In this scenario, we *fine-tune* local models of new sites by initializing them with the aggregated model ($FGSoT_{ft}$ and $FGSoT - NR_{ft}$). In addition to $FGSoT - NR$ and the local model, we compare $FGSoT$ with *centralized model*, where we combine the data of train sites and learn a single global model. These scenarios are illustrated in Figure 5.

In the simulations, we set $\epsilon = 0.01$, $\tau = 2$, $z = 10$, and $K_g = 100$. Note that ϵ and K_g are the termination criteria parameters that users can specify based on their available computing resources. We determine the values of τ and z for the divergence criterion by following (Li et al. 2018).

We performed a sensitivity analysis to determine the number of local iterations, K_l. Table 1 reports the effect of the number of local iterations in each round in terms of mean (and standard deviation) of test accuracy under the heterogeneous model assumption ($R = 3$, $\sigma_e = 0.5$, $\beta_{y\sim\mathcal{X}} = 0.10$ and $m_{het} = 3$). For instance, the performance of $FGSoT_{agg}$ is 0.882 (0.045), 0.869 (0.047), and 0.867 (0.056) when the number of local iterations is 5, 10, and 25, respectively. Table 1 shows no significant difference in the performance of the personalized model, $FGSoT_{per}$, when the number of local iterations changes from 5 to 25. Furthermore, the site-level performance comparison for different K_l values is provided in Table 1 in Supplementary Results (Section 2). In addition, Table 2 reports the average number of communication rounds until convergence for different numbers of local iterations. The table shows that the number of communication rounds does not change significantly, but it is minimum when the number of local iterations is 5. Therefore, the number of local iterations is set to 5 because it requires less local computation in each round and fewer communication rounds until convergence without compromising the test performances of $FGSoT_{agg}$ and $FGSoT_{per}$.

Table 1. Performance comparison of $FGSoT_{agg}$ and $FGSoT_{per}$ in terms of accuracy for different number of local iterations.

	# of local iterations				
Model	5	8	10	15	25
FGSoT$_{agg}$	0.882 (0.045)	0.877 (0.049)	0.869 (0.047)	0.881 (0.041)	0.867 (0.056)
FGSoT$_{per}$	0.882 (0.052)	0.876 (0.044)	0.877 (0.047)	0.882 (0.050)	0.878 (0.045)

Table 2. Number of communication rounds until convergence.

# of local iterations				
5	8	10	15	25
70.5	77.4	76.4	82.7	75.6

Table 3. Performance comparison of $FGSoT_{agg}$ and $FGSoT_{per}$ in terms of accuracy for different ψ and δ values.

	ψ & δ							
Model	0.001	0.01	0.03	0.05	0.1	0.5	1	5
FGSoT$_{agg}$	0.872 (0.042)	0.875 (0.047)	0.872 (0.047)	0.883 (0.040)	0.878 (0.049)	0.883 (0.039)	0.880 (0.036)	0.854 (0.052)
FGSoT$_{per}$	0.865 (0.043)	0.877 (0.047)	0.876 (0.039)	0.884 (0.041)	0.881 (0.041)	0.892 (0.034)	0.885 (0.033)	0.858 (0.053)

In order to select ψ and δ, we performed a sensitivity analysis. Table 3 reports the mean and standard deviation (shown in parentheses) of the accuracy of $FGSoT_{agg}$ and $FGSoT_{per}$ applied to test data sets for different ψ and δ values: $\psi = \delta \in \{0.001, 0.01, 0.03, 0.05, 0.1, 0.5, 1, 5\}$ under the heterogeneous model assumption ($R = 3$, $\sigma_e = 0.5, \beta_{y\sim\mathcal{X}} = 0.10$ and $m_{het} = 3$). For example, the average performance of $FGSoT_{agg}$ is 0.872 (0.042), 0.883 (0.040), 0.883 (0.039), and 0.854 (0.052) when ψ and δ are 0.001, 0.05, 0.5 and 5, respectively. Furthermore, the site-level performance comparison for different values of ψ and δ can be found in Table 1 in Supplementary Results (Section 3). The table shows that when ψ and δ are not too small (below 0.05) or too large (above 1), the average performances of $FGSoT_{agg}$ and $FGSoT_{per}$ are not sensitive to the ψ and δ selection. Based on this sensitivity analysis, we empirically set ψ and δ to 0.5.

Furthermore, we randomly initialize local and centralized models. For fine-tuned, local, and centralized models, we set $\epsilon = 0.0005$. We consider three performance measures: accuracy, precision, and recall. We compute accuracy, precision and recall as following:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN},$$
$$Precision = \frac{TP}{TP + FP},$$
$$Recall = \frac{TP}{TP + FN},$$

where TP, TN, FP and FN are the numbers of true positives, true negatives, false positives and false negatives. Finally, we repeat the simulations 30 times to compute the mean and standard deviation of the performance measures. The following summarizes our simulation scenarios:

Scenario I: In the first simulation, we generate five local sites as illustrated in Figure 5(a), each containing 100 training and 100 test samples. In this scenario, true underlying local models are homogeneous. When generating data, we set $m = 3$, $I_1 = 3, I_2 = 4, I_3 = 5$, $k = 0.2$, $\beta_{\mathcal{X}} = 0.1, \rho = 0.01$ and $\sigma_u = 0.1$, and simulate it with $\sigma_e = \{0.2, 0.5, 1\}$ and $R = \{2, 3, 4\}$.

Scenario II: This scenario simulates heterogeneous true underlying local models. Similar to Scenario I, we consider five local sites with 100 training and 100 test samples. Parameters are set similar to Scenario I except that $\sigma_e = \{0.5, 1\}, \beta_{y\sim\mathcal{X}} = \{0.05, 0.10, 0.15, 0.20, 0.25\}, R = \{3, 4\}$ and $m_{het} = \{1, 2, 3\}$.

Scenario III: The third scenario generates five local sites under heterogeneous model assumption with 100 training samples and four new sites with 100 training and 100 test samples. This scenario is depicted in Figure 5(b), under heterogeneous model assumption. We follow the same procedure explained in the second scenario to generate data. We set $R = 3$ and $\sigma_e = 0.5$, and we test the effect of the heterogeneity level on the performance of the proposed approach.

5.3. Simulation results

Scenario I: Tables 4–6 summarize the results in terms of mean and standard deviation (shown in parenthesis) of accuracy, precision, and recall, respectively, when true models are homogeneous. For example, Table 4 shows that the mean accuracy (and standard deviation) for $FGSoT_{agg}, FGSoT_{per}, FGSoT-NR_{agg}, FGSoT-NR_{per}$, and local model is 0.934

Table 4. Performance comparison for Scenario I with homogeneous models in terms of accuracy.

R	σ_e	FGSoT_{agg}	FGSoT_{per}	$\mathrm{FGSoT-NR}_{agg}$	$\mathrm{FGSoT-NR}_{per}$	Local
2	0.2	**0.941** (0.028)	0.940 (0.026)	0.920 (0.032)	0.899 (0.028)	0.870 (0.027)
2	0.5	**0.887** (0.042)	0.886 (0.042)	0.866 (0.048)	0.848 (0.053)	0.826 (0.052)
2	1	**0.813** (0.065)	0.812 (0.070)	0.799 (0.069)	0.761 (0.068)	0.743 (0.069)
3	0.2	**0.934** (0.027)	0.933 (0.024)	0.891 (0.046)	0.883 (0.026)	0.845 (0.031)
3	0.5	**0.899** (0.033)	0.896 (0.034)	0.865 (0.046)	0.843 (0.039)	0.811 (0.041)
3	1	**0.836** (0.039)	0.833 (0.040)	0.795 (0.060)	0.787 (0.049)	0.764 (0.041)
4	0.2	0.913 (0.038)	**0.918** (0.030)	0.877 (0.045)	0.868 (0.025)	0.834 (0.028)
4	0.5	0.898 (0.030)	**0.903** (0.025)	0.860 (0.039)	0.857 (0.029)	0.824 (0.036)
4	1	**0.840** (0.049)	0.837 (0.050)	0.805 (0.043)	0.784 (0.046)	0.757 (0.047)

Table 5. Performance comparison for Scenario I with homogeneous models in terms of precision.

R	σ_e	FGSoT_{agg}	FGSoT_{per}	$\mathrm{FGSoT-NR}_{agg}$	$\mathrm{FGSoT-NR}_{per}$	Local
2	0.2	0.940 (0.033)	**0.942** (0.030)	0.922 (0.031)	0.901 (0.028)	0.872 (0.028)
2	0.5	0.889 (0.049)	**0.889** (0.044)	0.870 (0.053)	0.850 (0.054)	0.829 (0.055)
2	1	0.815 (0.062)	**0.816** (0.067)	0.804 (0.071)	0.767 (0.072)	0.748 (0.073)
3	0.2	0.932 (0.030)	**0.933** (0.028)	0.894 (0.045)	0.885 (0.029)	0.849 (0.031)
3	0.5	**0.893** (0.042)	0.893 (0.043)	0.863 (0.047)	0.840 (0.046)	0.807 (0.047)
3	1	**0.835** (0.050)	0.834 (0.048)	0.800 (0.065)	0.793 (0.052)	0.773 (0.048)
4	0.2	0.904 (0.046)	**0.910** (0.036)	0.872 (0.057)	0.860 (0.042)	0.831 (0.045)
4	0.5	0.903 (0.042)	**0.908** (0.038)	0.862 (0.046)	0.862 (0.037)	0.827 (0.044)
4	1	**0.840** (0.059)	0.838 (0.060)	0.808 (0.050)	0.783 (0.051)	0.756 (0.058)

Table 6. Performance comparison for Scenario I with homogeneous models in terms of recall.

R	σ_e	FGSoT_{agg}	FGSoT_{per}	$\mathrm{FGSoT-NR}_{agg}$	$\mathrm{FGSoT-NR}_{per}$	Local
2	0.2	**0.942** (0.031)	0.939 (0.027)	0.915 (0.040)	0.894 (0.032)	0.865 (0.034)
2	0.5	**0.883** (0.043)	0.881 (0.045)	0.860 (0.044)	0.844 (0.055)	0.820 (0.053)
2	1	**0.811** (0.084)	0.807 (0.087)	0.793 (0.077)	0.754 (0.073)	0.739 (0.075)
3	0.2	**0.936** (0.030)	0.933 (0.031)	0.887 (0.050)	0.880 (0.033)	0.838 (0.046)
3	0.5	**0.902** (0.035)	0.895 (0.036)	0.859 (0.055)	0.838 (0.045)	0.807 (0.048)
3	1	**0.846** (0.046)	0.841 (0.052)	0.800 (0.064)	0.787 (0.055)	0.761 (0.054)
4	0.2	0.919 (0.048)	**0.923** (0.040)	0.875 (0.049)	0.870 (0.030)	0.834 (0.033)
4	0.5	0.893 (0.038)	**0.898** (0.032)	0.856 (0.048)	0.849 (0.038)	0.821 (0.043)
4	1	**0.838** (0.053)	0.834 (0.053)	0.801 (0.048)	0.783 (0.046)	0.755 (0.052)

Table 7. Performance comparison for Scenario II with heterogeneous models in terms of accuracy.

R	σ_e	$\beta_{y\sim\mathcal{X}}$	m_{het}	FGSoT_{agg}	FGSoT_{per}	$\mathrm{FGSoT-NR}_{agg}$	$\mathrm{FGSoT-NR}_{per}$	Local
3	0.5	0.05	1	0.898 (0.027)	**0.901** (0.026)	0.854 (0.072)	0.853 (0.038)	0.824 (0.041)
3	0.5	0.10	1	0.892 (0.032)	**0.898** (0.028)	0.862 (0.053)	0.850 (0.034)	0.821 (0.030)
3	0.5	0.10	2	0.895 (0.024)	**0.904** (0.023)	0.839 (0.092)	0.860 (0.036)	0.833 (0.032)
3	0.5	0.10	3	0.880 (0.033)	**0.887** (0.036)	0.852 (0.050)	0.848 (0.036)	0.826 (0.033)
3	0.5	0.15	3	0.868 (0.030)	**0.877** (0.029)	0.813 (0.087)	0.845 (0.033)	0.821 (0.035)
3	0.5	0.20	3	0.869 (0.036)	**0.886** (0.030)	0.833 (0.060)	0.843 (0.035)	0.830 (0.034)
3	0.5	0.25	3	0.858 (0.040)	**0.877** (0.035)	0.809 (0.098)	0.845 (0.035)	0.833 (0.031)
3	1	0.10	3	0.832 (0.046)	**0.837** (0.051)	0.802 (0.061)	0.787 (0.049)	0.772 (0.049)
4	0.5	0.10	3	0.870 (0.029)	**0.879** (0.030)	0.816 (0.077)	0.833 (0.029)	0.813 (0.032)
4	1	0.10	3	0.834 (0.034)	**0.839** (0.031)	0.772 (0.103)	0.797 (0.033)	0.780 (0.031)

(0.027), 0.993 (0.024), 0.891 (0.046), 0.883 (0.026) and 0.845 (0.031), respectively, when data is simulated with R = 3 and $\sigma_e = 0.2$. Tables 4–6 report that in the homogeneous case, both $FGSoT$ and $FGSoT-NR$ outperform local models for all rank and noise levels in terms of accuracy, precision, and recall. More specifically, in most rank and noise combinations, $FGSoT_{agg}$ gives the highest accuracy on test sets.

Scenario II: Tables 7–9 show the performance of the models in terms of accuracy, precision, and recall under different heterogeneity (in $\beta_{y\sim\mathcal{X}}$ and m_{het} columns), rank, and noise levels (in R and σ_e columns). In this scenario, $FGSoT_{agg}$ and $FGSoT_{per}$ outperform the benchmarks in all cases reaching the highest accuracy, precision, and recall under the heterogeneous model assumption. Moreover, $FGSoT_{per}$ further improves the performance with personalization.

Scenario III: Tables 10–12 report that, under different levels of heterogeneity, $FGSoT$ outperforms the local model and provides a comparable or better result compared to the centralized model in terms of accuracy, precision, and recall when applied to new sites. For instance, the accuracy level of $FGSoT_{agg}$, $FGSoT_{ft}$, $FGSoT-NR_{agg}$, $FGSoT-NR_{ft}$, local, and centralized models are

Table 8. Performance comparison for Scenario II with heterogeneous models in terms of precision.

R	σ_e	$\beta_{y\sim\mathcal{X}}$	m_{het}	$FGSoT_{agg}$	$FGSoT_{per}$	$FGSoT-NR_{agg}$	$FGSoT-NR_{per}$	Local
3	0.5	0.05	1	0.900 (0.036)	**0.904** (0.032)	0.854 (0.081)	0.856 (0.047)	0.829 (0.048)
3	0.5	0.10	1	0.892 (0.039)	**0.898** (0.037)	0.859 (0.060)	0.847 (0.042)	0.823 (0.040)
3	0.5	0.10	2	0.898 (0.027)	**0.906** (0.022)	0.840 (0.096)	0.864 (0.036)	0.836 (0.035)
3	0.5	0.10	3	0.881 (0.034)	**0.888** (0.039)	0.856 (0.056)	0.849 (0.037)	0.828 (0.048)
3	0.5	0.15	3	0.865 (0.035)	**0.875** (0.033)	0.810 (0.095)	0.841 (0.040)	0.815 (0.045)
3	0.5	0.20	3	0.874 (0.042)	**0.889** (0.039)	0.835 (0.063)	0.844 (0.038)	0.836 (0.040)
3	0.5	0.25	3	0.858 (0.056)	**0.879** (0.045)	0.810 (0.110)	0.826 (0.033)	0.849 (0.044)
3	1	0.10	3	0.822 (0.055)	**0.828** (0.057)	0.791 (0.070)	0.779 (0.055)	0.764 (0.055)
4	0.5	0.10	3	0.877 (0.037)	**0.886** (0.038)	0.819 (0.078)	0.839 (0.034)	0.820 (0.042)
4	1	0.10	3	0.834 (0.044)	**0.841** (0.042)	0.767 (0.101)	0.793 (0.040)	0.778 (0.036)

Table 9. Performance comparison for Scenario II with heterogeneous models in terms of recall.

R	σ_e	$\beta_{y\sim\mathcal{X}}$	m_{het}	$FGSoT_{agg}$	$FGSoT_{per}$	$FGSoT-NR_{agg}$	$FGSoT-NR_{per}$	Local
3	0.5	0.05	1	0.897 (0.030)	**0.898** (0.034)	0.859 (0.059)	0.848 (0.042)	0.819 (0.045)
3	0.5	0.10	1	0.890 (0.039)	**0.895** (0.032)	0.866 (0.050)	0.852 (0.039)	0.816 (0.037)
3	0.5	0.10	2	0.891 (0.033)	**0.900** (0.032)	0.835 (0.099)	0.856 (0.046)	0.829 (0.045)
3	0.5	0.10	3	0.877 (0.042)	**0.887** (0.040)	0.848 (0.052)	0.849 (0.041)	0.826 (0.035)
3	0.5	0.15	3	0.866 (0.041)	**0.876** (0.039)	0.812 (0.087)	0.844 (0.034)	0.823 (0.033)
3	0.5	0.20	3	0.863 (0.045)	**0.882** (0.035)	0.831 (0.070)	0.841 (0.046)	0.824 (0.043)
3	0.5	0.25	3	0.854 (0.043)	**0.871** (0.041)	0.804 (0.104)	0.834 (0.040)	0.828 (0.041)
3	1	0.10	3	0.837 (0.064)	**0.838** (0.069)	0.808 (0.071)	0.789 (0.056)	0.772 (0.061)
4	0.5	0.10	3	0.861 (0.041)	**0.871** (0.038)	0.813 (0.081)	0.825 (0.041)	0.805 (0.039)
4	1	0.10	3	0.827 (0.045)	**0.828** (0.041)	0.770 (0.105)	0.791 (0.037)	0.774 (0.034)

Table 10. Performance comparison for Scenario III with heterogeneous models and new sites in terms of accuracy.

$\beta_{y\sim\mathcal{X}}$	m_{het}	$FGSoT_{agg}$	$FGSoT_{ft}$	$FGSoT-NR_{agg}$	$FGSoT-NR_{ft}$	Local	Centralized
0.05	1	0.895 (0.030)	0.855 (0.030)	0.854 (0.065)	0.841 (0.037)	0.829 (0.028)	**0.896** (0.026)
0.10	1	0.886 (0.032)	0.843 (0.032)	0.855 (0.047)	0.830 (0.038)	0.823 (0.031)	**0.887** (0.023)
0.10	2	**0.892** (0.033)	0.864 (0.032)	0.847 (0.087)	0.851 (0.040)	0.837 (0.031)	0.884 (0.031)
0.10	3	0.874 (0.036)	0.848 (0.039)	0.840 (0.059)	0.832 (0.049)	0.818 (0.038)	**0.874** (0.030)
0.15	3	**0.857** (0.039)	0.843 (0.043)	0.823 (0.060)	0.833 (0.046)	0.822 (0.045)	0.847 (0.041)
0.20	3	**0.858** (0.031)	0.842 (0.040)	0.825 (0.056)	0.839 (0.040)	0.829 (0.036)	0.852 (0.029)
0.25	3	0.833 (0.056)	**0.849** (0.041)	0.800 (0.078)	0.843 (0.046)	0.827 (0.043)	0.833 (0.057)

Table 11. Performance comparison for Scenario III with heterogeneous models and new sites in terms of precision.

$\beta_{y\sim\mathcal{X}}$	m_{het}	$FGSoT_{agg}$	$FGSoT_{ft}$	$FGSoT-NR_{agg}$	$FGSoT-NR_{ft}$	Local	Centralized
0.05	1	0.895 (0.037)	0.858 (0.034)	0.853 (0.072)	0.843 (0.044)	0.833 (0.033)	**0.897** (0.029)
0.10	1	**0.886** (0.037)	0.841 (0.034)	0.856 (0.051)	0.829 (0.040)	0.824 (0.033)	0.885 (0.029)
0.10	2	**0.894** (0.038)	0.858 (0.039)	0.848 (0.093)	0.840 (0.051)	0.834 (0.038)	0.884 (0.040)
0.10	3	0.871 (0.050)	0.851 (0.050)	0.837 (0.067)	0.834 (0.058)	0.832 (0.047)	**0.871** (0.039)
0.15	3	**0.850** (0.046)	0.839 (0.046)	0.816 (0.067)	0.832 (0.046)	0.822 (0.048)	0.840 (0.052)
0.20	3	**0.859** (0.040)	0.038 (0.049)	0.829 (0.065)	0.840 (0.042)	0.825 (0.044)	0.849 (0.036)
0.25	3	0.833 (0.052)	**0.853** (0.042)	0.804 (0.075)	0.848 (0.045)	0.831 (0.043)	0.831 (0.059)

Table 12. Performance comparison for Scenario III with heterogeneous models and new sites in terms of recall.

$\beta_{y\sim\mathcal{X}}$	m_{het}	$FGSoT_{agg}$	$FGSoT_{ft}$	$FGSoT-NR_{agg}$	$FGSoT-NR_{ft}$	Local	Centralized
0.05	1	**0.894** (0.037)	0.847 (0.043)	0.855 (0.060)	0.834 (0.041)	0.818 (0.036)	0.892 (0.036)
0.10	1	0.886 (0.035)	0.847 (0.035)	0.852 (0.048)	0.832 (0.043)	0.820 (0.042)	**0.888** (0.033)
0.10	2	**0.884** (0.042)	0.866 (0.038)	0.833 (0.097)	0.856 (0.047)	0.834 (0.040)	0.876 (0.037)
0.10	3	**0.876** (0.037)	0.840 (0.052)	0.843 (0.054)	0.826 (0.057)	0.806 (0.059)	0.876 (0.038)
0.15	3	**0.860** (0.049)	0.840 (0.061)	0.824 (0.065)	0.823 (0.063)	0.808 (0.066)	0.848 (0.046)
0.20	3	**0.856** (0.035)	0.849 (0.040)	0.819 (0.065)	0.843 (0.048)	0.840 (0.040)	0.853 (0.038)
0.25	3	0.836 (0.072)	**0.843** (0.050)	0.801 (0.080)	0.837 (0.057)	0.821 (0.057)	0.839 (0.061)

0.892 (0.033), 0.864 (0.032), 0.847 (0.087), 0.851 (0.040), 0.837 (0.031) and 0.884 (0.031), respectively, when $\beta_{y\sim\mathcal{X}} = 0.1$ and $m_{het} = 2$. Note that as the heterogeneity level increases (i.e., as the sites become more distinct), the performance difference between *FGSoT* and the local model, when applied to new test sites, is expected to diminish in this scenario. The main reason is that the test sites have not participated in the collaborative model learning procedure. However, even in this setting, *FGSoT* outperforms the local model. Furthermore, the centralized model represents the ideal case where all data is shared across sites. Using data from all local sites potentially results in a more generalizable model due to the access to a larger sample size and shared knowledge across sites. Therefore, having better or comparable performance with the centralized model shows the efficacy of *FGSoT*, which only has access to the local models trained based on limited data from heterogeneous sites.

Effect of regularization: In all simulation scenarios, *FGSoT* outperforms $FGSoT-NR$ in terms of accuracy, precision and recall. The results show that regularization provides a more stable model with a higher mean and lower standard deviation of accuracy in all scenarios. During the training process under the federated scheme, regularization limits the deviation of the locally trained models from the aggregated model in heterogeneous model settings (Scenarios II and III).

Effect of personalization: In the heterogeneous setting, personalized models outperform the aggregated models. Personalization allows local sites to benefit from collaboration and site-specific information by performing one more stage of local updates. On the other hand, when true models are homogeneous, $FGSoT_{agg}$ is comparable to $FGSoT_{per}$ since $FGSoT_{agg}$ ensembles the local models of homogeneous sites which are similar in distribution (For more details, see Section 7). Furthermore, the site-level results are provided in Supplementary Results (Section 4).

6. Case study

This section introduces two case studies. The objective is to validate the performance of *FGSoT* with real data sets. In the first study, we apply the proposed approach to detect a disease in tomato plant leaves by using hyperspectral image data collected by unmanned aerial vehicle (UAV). In the second study, we evaluate the performance of *FGSoT* to detect fault events in vehicle engines, using sensor data.

The case study experiments follow the same procedure to tune α_t, $(t = 1, \ldots, T)$ by minimizing the Bayesian Information Criterion (BIC) as described in Section 4.5. We follow the sensitivity analysis approach to select ψ and δ, which resulted in $\psi = \delta = 0.5$. Similar to the simulation studies, these case studies use convergence check, divergence check, and the maximum number of iterations as stopping criteria. We set the convergence threshold, ϵ, to 0.0005, the divergence parameters, τ and z, to 2 and 10, respectively, and the maximum number of iterations, K_g, to 100. Finally, we set the number of local iterations (K_l) to 5.

6.1. Plant disease detection using hyperspectral images in tomato farm

Fast and accurate detection of diseases in an agricultural farm is critical for farm owners and growers. This process often requires lab experiments, which are often expensive and time-consuming. Therefore, constructing models that can identify diseased plants based on images obtained from a farm is extremely beneficial. For this purpose, unmanned aerial vehicles (UAVs) equipped with remote sensing systems collect spectral image data from different regions of interest (RoIs) on a farm (Abdulridha et al. 2020; Costa et al. 2022). These images are then converted to reflectance curves over a wide wavelength range. This data is then labeled using lab results to create an often small training data set to develop a classification model for disease detection. This model is finally used for future predictions based on spectral images obtained rapidly from the farm.

This case study considers six sites of tomato seedlings at the Southwest Florida Research and Education Center, some of which are inoculated with a black spot disease. Hyperspectral images are obtained from each site and converted into reflectance curves over different wavelengths, which resulted in a small set of high dimensional data. These signals may serve as an indicator of healthy/diseased conditions of a plant. Our objective is to construct a federated classification model to detect diseased plants based on reflectance profiles, located at different sites. Figure 6 shows the average reflectance profiles of three sites, as examples, where solid lines indicate diseased and dotted lines indicate healthy instances. In the figure, the relationship between the healthy/diseased condition and the reflectance levels are different in different sites, indicating heterogeneity in models. Given the reflectance profiles

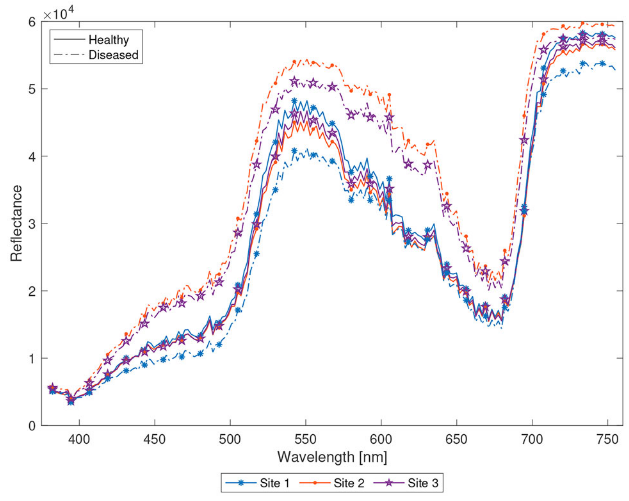

Figure 6. Average hyperspectral reflectance curves.

defined over 180 wavelengths, we generate tensors $\mathcal{X}_{i,t} \in \mathbb{R}^{6\times6\times5}$, $(i = 1, ..., 60;\ t = 1, ..., 6)$ by reshaping each profile. Reshaping the tensor balances the dimensions and will improve the computational time (See Section 7 for more details). To evaluate the performance of the proposed approach, we consider five training sites and one testing site, each with 30 healthy and 30 diseased instances. We assume five training sites collaborate to construct an aggregated model to be tested on the testing site data. That is, a new site joins the network and uses the aggregated model constructed based on the shared knowledge of other sites.

Figure 7 shows the boxplots of each performance metric (accuracy in Figure 7(a), precision in Figure 7(b), and recall in Figure 7(c)) of $FGSoT_{agg}$, $FGSoT_{per}$, $FGSoT-NR_{agg}$, $FGSoT-NR_{per}$, local model, and centralized model. In the boxplots, red diamond dots show the mean of the corresponding performance measure. Note that the reported values for the local model benchmark are obtained by averaging the performance of local models constructed by each of the five training sites and applied to the testing site. Figure 7 demonstrates that the mean and the median of each performance measure of $FGSoT_{agg}$ is superior to that of the local models with lower variance. Furthermore, $FGSoT_{agg}$ provides comparable performance to the centralized model, which is expected to perform well as it has access to all the data from different sites. These results show the benefit of collaborative modeling frameworks in agricultural applications, where data sharing is not feasible due to competition and privacy concerns.

6.2. Vehicle engine fault detection using sensor data

NO_x Storage Catalyst (NSC) is a control system for gas emissions used for combustion engines (Pacella 2018). In this system, exhaust gas is processed in two stages: (i) adsorption, where NO_x molecules are catched by an adsorber, and (ii) regeneration, where NO_x molecules are reduced as the adsorber becomes saturated. However, fault events happening during the regeneration phase diminish the efficiency of NO_x reduction. Therefore, it is critical to identify those fault events and find their root causes.

Combustion engines have multiple sensors which collect real-time profile data. Figure 8 shows signals of example sensors that measure air/fuel ratio (λ), actual air mass, and injection quantity. The signals are color-coded with respect to two fault event types. The figure shows that these signals have different patterns in regard to these fault events. Hence, the multichannel profile data can be exploited to help identify fault events in the regeneration phase of NSC. Detecting fault events and associating them with their root causes will boost the efficiency of NSC, which is necessary for gas emission regulations. In this study, we use signals of lambda (λ), actual air mass (mg/s), injection quantity (mg/s), boost pressure actual value (mbar), and low-pressure exhaust gas recirculation (EGR) valve position (%) sensors. Each sensor is a data channel. The original signals contain 203 observations obtained in an interval of 2 s. The signals are noisy, as shown in Figure 8. To slightly smoothen the signals, we first perform a moving average using a sliding window of size 3. Hence, the smoothed signals have 200 observations. Based on our experience, reshaping the signals into a 3D tensor by slicing them into pieces is beneficial as it improves the computational time. Furthermore, tensors can capture the correlations within and between slices (See Section 7 for more details). Therefore, we reshape the smoothed signals to generate tensors, $\mathcal{X}_{i,t} \in \mathbb{R}^{20\times10\times5}$ $(i = 1, ..., 19\ or\ 20; t = 1, ..., 4)$. Here, 20 is the slice length, 10 is the number of slices, and 5 is the number of channels (i.e., lambda, actual air

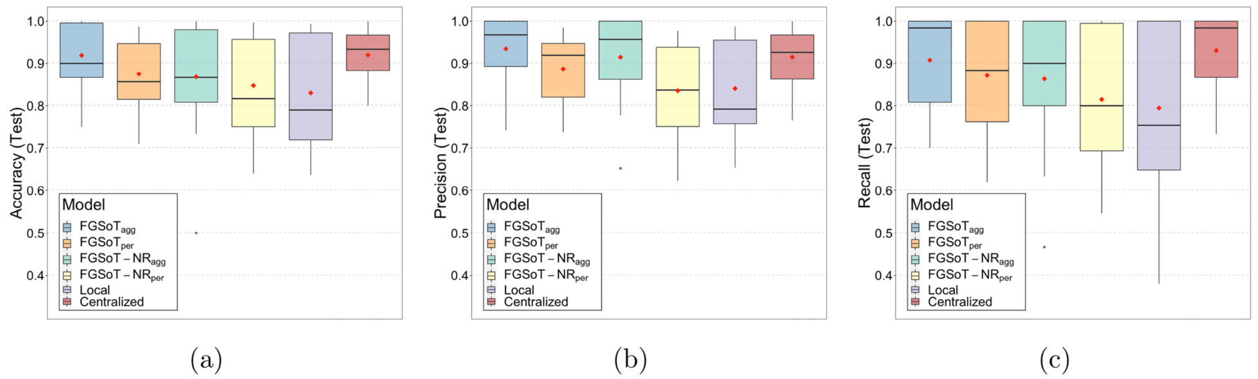

Figure 7. Performance comparison for tomato plants classification case study in terms of (a) accuracy, (b) precision, and (c) recall. Red diamond dots show the mean of the corresponding performance measure.

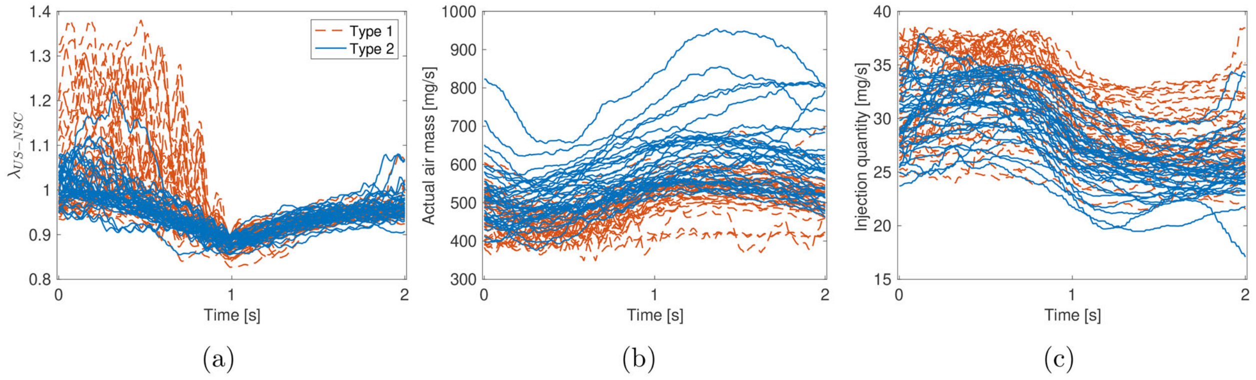

Figure 8. Signals of (a) lambda (λ), (b) actual air mass, and (c) injection quantity sensors.

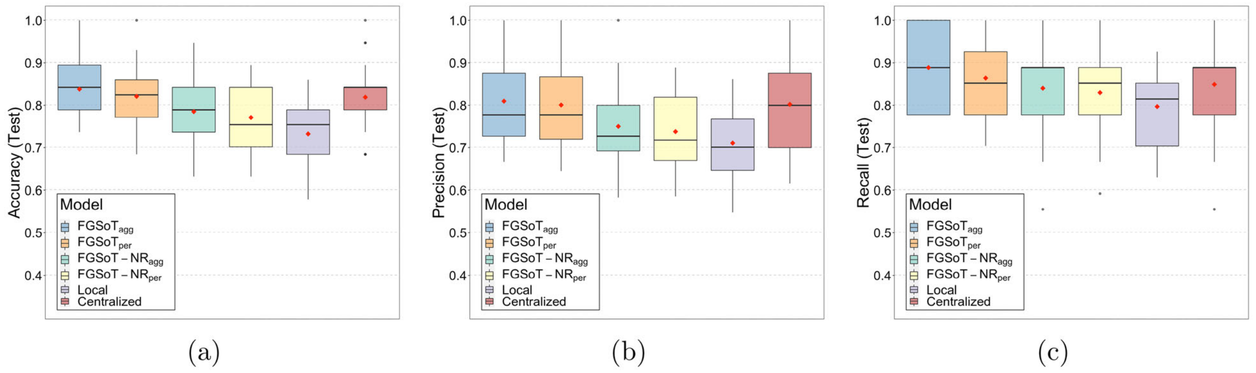

Figure 9. Performance comparison for vehicle engine case study in terms of (a) accuracy, (b) precision, and (c) recall. Red diamond dots show the mean of the corresponding performance measure.

mass, injection quantity, boost pressure actual value, and low-pressure gas recirculation). We consider three training sites and one testing site, each with 19 or 20 samples, and follow the same procedure explained in the methodology to train models. Thus, three training sites collaborate to construct an aggregated model to be tested on the testing site data. Hence, the objective in this case study is to classify fault events based on localized and high-dimensional data by using the proposed approach.

Figure 9 shows the boxplots of performance measures of *FGSoT* and the benchmarks. Figure 9 shows that the mean and the median of each performance measure for $FGSoT_{agg}$ is superior to that of the local model with lower variance. Furthermore, $FGSoT_{agg}$ achieves a comparable performance to the centralized model. These results present the benefit of collaboration in a federated setting in a network of factories.

7. Discussion

The proposed tensor regression framework is general and can handle functional data as well. Particularly, the proposed approach is beneficial when multi-channel functional data is available (For example, Case Study II). In this setting, each sample in the data can be arranged as a matrix of *channels* $\times$ *profile lengths*. The benefit of our approach compared to existing functional data analysis methods is that by using a tensor, the correlation structures between and within channels can effectively be modeled. Often, the length of signals may be much larger than the number of channels. Under this scenario, it is beneficial to slice the signal into pieces and convert the data into a 3D tensor (*channels* $\times$ *slice lengths* $\times$ *number of slices*). This slicing approach balances the dimensions of the tensor and will improve the computational time (Lee et al. 2023). Please note that because tensors can model correlations between modes, the relationship between slices (sub-profiles) is still preserved in the reshaped tensor data. Our results in the case studies further support the effectiveness of generating tensors from functional data.

We use CP decomposition in the proposed tensor regression model. CP decomposition has commonly been used in tensor regression, and its effectiveness has been shown in (Zhou, Li, and Zhu 2013; Fang, Paynabar, and Gebraeel 2019). The main advantage of CP decomposition over Tucker decomposition is that it has a fewer tuning parameters. More specifically, CP decomposition requires selecting a single rank. In comparison, the Tucker decomposition decomposes the parameter tensor into a core tensor and a set of factor matrices, and requires selecting a rank for each mode of the core tensor. In the federated settings, selecting a rank for each mode should be done at each site and globally, exacerbating the problem. Furthermore, when using CP decomposition, each site only shares the low-dimensional basis matrices with the central server to obtain an aggregated model. However, Tucker decomposition will require each site to share the core tensor and the factor matrices, increasing the communication with the central server. Alternatively, if each site sends only the factor matrices and keeps the core tensor to reduce the communication, the Tucker method can only construct personalized local models as there will be no aggregated core tensor to construct an aggregated model. Given these differences, Tucker decomposition should be studied separately for federated tensor regression.

Furthermore, Newton's search method used in parameter estimation is not an integral part of the proposed method. The search method can be determined by the user depending on the resources they have. One advantage of Newton's search method is its quadratic convergence property. For larger-scale problems where the inverse calculation of the Hessian matrix is too costly, the proposed framework can easily be adjusted to use other optimization techniques, such as quasi-Newton, and first- and zero-order techniques.

Finally, in the proposed framework, personalization allows local sites to benefit from collaboration and site-specific information by performing one more stage of local updates. Our results show that personalization further improves the performance of the aggregated model under the heterogeneous model assumption. Especially as the heterogeneity level increases, the performance gain with additional local updates increases. In real applications, it may not be possible to know if the local sites are homogeneous or heterogeneous. In that situation, training both $FGSoT_{agg}$ and $FGSoT_{per}$ and testing on a validation set is possible. However, our results show that $FGSoT_{per}$ performs almost equivalently as $FGSoT_{agg}$ under the homogeneous model assumption and outperforms $FGSoT_{agg}$ under the heterogeneous model assumption. Therefore, a personalized model can be selected in all cases.

8. Conclusion

This article proposes a federated generalized scalar-on-tensor regression (FGSoT) framework to model high-dimensional and distributed data. In this framework, multiple local tensor regression models are trained at the edge, and their parameters are shared with a central server, which aggregates the models by solving an optimization problem. This framework incorporates proximity regularization terms in the local objective functions to tackle statistical heterogeneity among local sites. In addition, group lasso penalties are included to personalize local models. To estimate parameters, the alternating least square (ALS) approach with the Newton's search method is applied. The results from the simulation studies show that under both homogeneous and heterogeneous settings, the proposed *FGSoT* outperforms the benchmarks. Furthermore, the case studies from agriculture and manufacturing domains validated the superiority of the proposed framework compared to several benchmarks, indicating the benefits of collaboration between local sites. As a future work, differential privacy approaches can be employed to generate a federated system that is more secure and robust against adversarial attacks.

About the authors

Elif Konyar is a doctoral student in the Department of Industrial and Systems Engineering at University of Florida. Her email address is elif.konyar@ufl.edu.

Dr. Mostafa Reisi Gahrooei is an Assistant Professor in the Department of Industrial and Systems Engineering at University of Florida. His email address is mreisigahrooei@ufl.edu. He is the corresponding author.

Acknowledgements

We would like to thank Ioannis Ampatzidis, Lucas Fideles Costa and Vitor Gontijo da Cunha for providing hyperspectral image data collected at the Southwest Florida Research and Education Center. Also, we would like to thank Massimo Pacella for providing access to the vehicle engine sensor data.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This work has been partially supported by the National Science Foundation (NSF) award 2212878.

ORCID

Elif Konyar http://orcid.org/0000-0001-5491-069X
Mostafa Reisi Gahrooei http://orcid.org/0000-0002-7633-9575

Data availability statement

The data used in this article are not publicly available. To request access to the data used in Case Study I (Section 6.1) and Case Study II (Section 6.2), one may contact the corresponding authors of (Costa et al. 2022) and (Pacella 2018), respectively.

References

Abdulridha, J., Y. Ampatzidis, S. C. Kakarla, and P. Roberts. 2020. Detection of target spot and bacterial spot diseases in tomato using UAV-based and benchtop-based hyperspectral imaging techniques. *Precision Agriculture* 21 (5): 955–78. doi: 10.1007/s11119-019-09703-4.

Acar, D. A. E., Y. Zhao, R. M. Navarro, M. Mattina, P. N. Whatmough, and V. Saligrama. 2021. Federated learning based on dynamic regularization. *arXiv preprint arXiv: 2111.04263.*

Costa, L., J. McBreen, Y. Ampatzidis, J. Guo, M. R. Gahrooei, and M. A. Babar. 2022. Using UAV-based hyperspectral imaging and functional regression to assist in predicting grain yield and related traits in wheat under heat-related stress environments for the purpose of stable yielding genotypes. *Precision Agriculture* 23 (2):622–42. doi: 10.1007/s11119-021-09852-5.

Du, W., Y. S. Han, and S. Chen. 2004. Privacy-preserving multivariate statistical analysis: Linear regression and classification. In *Proceedings of the 2004 SIAM International Conference on Data Mining*, 222–233. SIAM. doi: 10.1137/1.9781611972740.21.

Fan, Z., and M. Reimherr. 2017. High-dimensional adaptive function-on-scalar regression. *Econometrics and Statistics* 1:167–83. doi: 10.1016/j.ecosta.2016.08.001.

Fang, X., K. Paynabar, and N. Gebraeel. 2019. Image-based prognostics using penalized tensor regression. *Technometrics* 61 (3):369–84. doi: 10.1080/00401706.2018.1527727.

Gahrooei, M. R., H. Yan, K. Paynabar, and J. Shi. 2021. Multiple tensor-on-tensor regression: An approach for modeling processes with heterogeneous sources of data. *Technometrics* 63 (2):147–59. doi: 10.1080/00401706.2019.1708463.

Gao, Y., G. Zhang, C. Zhang, J. Wang, L. T. Yang, and Y. Zhao. 2021. Federated tensor decomposition-based feature extraction approach for industrial IOT. *IEEE Transactions on Industrial Informatics* 17 (12):8541–9. doi: 10.1109/TII.2021.3074152.

Guhaniyogi, R., S. Qamar, and D. B. Dunson. 2017. Bayesian tensor regression. *Journal of Machine Learning Research* 18 (1):2733–63.

He, Z., and Z. Zhang. 2020. High-dimensional uncertainty quantification via active and rank-adaptive tensor regression. In *2020 IEEE 29th Conference on Electrical Performance of Electronic Packaging and Systems (EPEPS)*, 1–3. IEEE. doi: 10.1109/EPEPS48591.2020.9231388.

Hoff, P. D. 2015. Multilinear tensor regression for longitudinal relational data. *Annals of Applied Statistics* 9 (3): 1169–93. doi: 10.1214/15-AOAS839.

Karimireddy, S. P., S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh. 2020. SCAFFOLD: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, 5132–43. PMLR.

Kim, Y., J. Sun, H. Yu, and X. Jiang. 2017. Federated tensor factorization for computational phenotyping. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 887–895. doi: 10.1145/3097983.3098118.

Kolda, T. G., and B. W. Bader. 2009. Tensor decompositions and applications. *SIAM Review* 51 (3):455–500. doi: 10.1137/07070111X.

Kontar, R., N. Shi, X. Yue, S. Chung, E. Byon, M. Chowdhury, J. Jin, W. Kontar, N. Masoud, M. Noueihed, et al. 2021. The internet of federated things (IOFT): A vision for the future and in-depth survey of data-driven approaches for federated learning. *arXiv preprint arXiv: 2111.05326.*

Lee, H. Y., M. Reisi Gahrooei, H. Liu, and M. Pacella. 2023. Robust tensor-on-tensor regression for multidimensional data modeling. *IISE Transactions* (just-accepted):1–11. doi: 10.1080/24725854.2023.2183440.

Li, B., X. Xu, L. Zhang, J. Han, C. Bian, G. Li, J. Liu, and L. Jin. 2020a. Above-ground biomass estimation and yield prediction in potato by using UAV-based RGB and hyperspectral imaging. *ISPRS Journal of Photogrammetry and Remote Sensing* 162:161–72. doi: 10.1016/j.isprsjprs.2020.02.013.

Li, T., A. Beirami, M. Sanjabi, and V. Smith. 2020b. Tilted empirical risk minimization. *arXiv preprint arXiv: 2007.01162.*

Li, T., S. Hu, A. Beirami, and V. Smith. 2021. Ditto: Fair and robust federated learning through personalization. In *International Conference on Machine Learning*, 6357–68. PMLR.

Li, T., A. K. Sahu, A. Talwalkar, and V. Smith. 2020c. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine* 37 (3):50–60. doi: 10.1109/MSP.2020.2975749.

Li, T., A. K. Sahu Zaheer, M. Sanjabi, M. Talwalkar, and A. Smith. 2018. V. Federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127.*

Lock, E. F. 2018. Tensor-on-tensor regression. *Journal of Computational and Graphical Statistics: A Joint Publication of American Statistical Association, Institute of Mathematical Statistics, Interface Foundation of North America* 27 (3):638–47. doi: 10.1080/10618600.2017.1401544.

Ma, J., Q. Zhang, J. Lou, J. C. Ho, L. Xiong, and X. Jiang. 2019. Privacy-preserving tensor factorization for collaborative health data analysis. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management, 1291–1300. doi: 10.1145/3357384.3357878.

McMahan, B., E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In Artificial Intelligence and Statistics, 1273–82. PMLR.

Ogden, R. T., C. E. Miller, K. Takezawa, and S. Ninomiya. 2002. Functional regression in crop lodging assessment with digital images. *Journal of Agricultural, Biological, and Environmental Statistics* 7 (3):389–402. doi: 10.1198/108571102339.

Pacella, M. 2018. Unsupervised classification of multichannel profile data using PCA: An application to an emission control system. *Computers & Industrial Engineering* 122:161–9. doi: 10.1016/j.cie.2018.05.029.

Pillutla, K., K. Malik, A. Mohamed, M. Rabbat, M. Sanjabi, and L. Xiao. 2022. Federated learning with partial model personalization. *arXiv preprint arXiv:2204.03809.*

Roy, S., and G. Michailidis. 2022. Regularized high dimension low tubal-rank tensor regression. *Electronic Journal of Statistics* 16 (1):2683–723. doi: 10.1214/22-EJS2004.

Yan, H., K. Paynabar, and M. Pacella. 2019. Structured point cloud data analysis via regularized tensor regression for process modeling and optimization. *Technometrics* 61 (3):385–95. doi: 10.1080/00401706.2018.1529628.

Yang, Q., Y. Liu, T. Chen, and Y. Tong. 2019. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology*)10 (2):1–19. doi: 10.1145/3298981.

Yu, R., and Y. Liu. 2016. Learning from multiway data: Simple and efficient tensor regression. In *International Conference on Machine Learning*, 373–81. PMLR.

Yue, X., M. Nouiehed, and R. A. Kontar. 2021. GIFAIR-FL: An approach for group and individual fairness in federated learning. *arXiv preprint arXiv:2108.02741.*

Zhou, H., L. Li, and H. Zhu. 2013. Tensor regression with applications in neuroimaging data analysis. *Journal of the American Statistical Association* 108 (502):540–52. doi: 10.1080/01621459.2013.776499.

Appendix A. Parameter estimation details

The gradient and the Hessian of (4) with respect to $\mathbf{u}_{d,t}^{(r)}$ are computed as:

$$G(\boldsymbol{u}_{d,t}^{(r)}) = \sum_{i=1}^{N_t}\Big(-\boldsymbol{w}_{i,d,t}^{(r)}y_{i,t} + \boldsymbol{w}_{i,d,t}^{(r)}b'\Big(\mu_t + g_{i,d,t}^{(-r)} + \boldsymbol{u}_{d,t}^{(r)^\top}\boldsymbol{w}_{i,d,t}^{(r)}\Big)\Big) + N_t\frac{\alpha_t}{2}\frac{\boldsymbol{u}_{d,t}^{(r)}}{||\boldsymbol{u}_{d,t}^{(r)}||_2} + N_t\psi\Big(\boldsymbol{u}_{d,t}^{(r)} - \boldsymbol{v}_d^{(r)}\Big), \tag{6}$$

$$H(\boldsymbol{u}_{d,t}^{(r)}) = \sum_{i=1}^{N_t}\Big(\boldsymbol{w}_{i,d,t}^{(r)}\boldsymbol{w}_{i,d,t}^{(r)^\top}b''\Big(\mu_t + g_{i,d,t}^{(-r)} + \boldsymbol{u}_{d,t}^{(r)^\top}\boldsymbol{w}_{i,d,t}^{(r)}\Big)\Big) + N_t\frac{\alpha_t}{2}\left(\frac{\boldsymbol{I}}{||\boldsymbol{u}_{d,t}^{(r)}||_2} - \frac{\boldsymbol{u}_{d,t}^{(r)}\boldsymbol{u}_{d,t}^{(r)^\top}}{||\boldsymbol{u}_{d,t}^{(r)}||_2^3}\right) + N_t\ \psi\ I, \tag{7}$$

where $G(.)$ is the gradient, $H(.)$ is the Hessian matrix, $b'(.)$ is the first derivative of $b(.)$, and $b''(.)$ is the second derivative of $b(.)$.

Moreover, the first and the second derivatives of (4) with respect to μ_t, $f'(\mu_t)$ and $f''(\mu_t)$, are computed as:

$$\frac{\partial f}{\partial \mu_t} = f'(\mu_t) = \sum_{i=1}^{N_t}(-y_{i,t} + b'(\mu_t + \mathrm{Tr}(\boldsymbol{U}_{d,t}^\top\mathcal{X}_{i,(d),t}\boldsymbol{K}_{d,t}))) + N_t\delta(\mu_t - \zeta), \tag{8}$$

$$\frac{\partial^2 f}{\partial \mu_t^2} = f''(\mu_t) = \sum_{i=1}^{N_t}b''(\mu_t + \mathrm{Tr}(\boldsymbol{U}_{d,t}^\top\mathcal{X}_{i,(d),t}\boldsymbol{K}_{d,t})) + N_t\delta. \tag{9}$$

Then, we update μ_t with the Newton's search method as ${}_*\mu_t = \mu_t - \gamma\ f''(\mu_t)^{-1}f'(\mu_t)$, where γ is the step size.

Appendix B. Identifiability

Two main sources of nonidentifiability are scaling and permutation indeterminacy (Zhou, Li, and Zhu 2013). First, let us show that the estimates of parameters $\mathbf{u}_{d,t}^{(r)}$ and μ_t, are unique in each local step. Recall that the objective function for the local model is:

$$\begin{aligned} f_t(\mu_t, \boldsymbol{u}_{d,t}^{(r)}) = &\sum_{i=1}^{N_t}(-y_{i,t}(\mu_t + g_{i,d,t}^{(-r)} + \boldsymbol{u}_{d,t}^{(r)^\top}\mathbf{w}_{i,d,t}^{(r)}) + b(\mu_t + g_{i,d,t}^{(-r)} + \boldsymbol{u}_{d,t}^{(r)^\top}\mathbf{w}_{i,d,t}^{(r)})) \\ &+N_t\sum_{d=1}^{m}\sum_{r=1}^{R}\frac{\alpha_t}{2}||\boldsymbol{u}_{d,t}^{(r)}||_2 + N_t\sum_{d=1}^{m}\sum_{r=1}^{R}\frac{\psi}{2}||\boldsymbol{u}_{d,t}^{(r)} - \boldsymbol{v}_d^{(r)}||_2^2 + N_t\frac{\delta}{2}(\mu_t - \zeta)^2. \end{aligned} \tag{10}$$

In each step, we fix one parameter (e.g., μ_t) and estimate the other one. We show that this objective function (10) is convex with respect to both $\mathbf{u}_{d,t}^{(r)}$ and μ_t in each step. The first term, i.e., $-y_{i,t}(\mu_t + g_{i,d,t}^{(-r)} + \mathbf{u}_{d,t}^{(r)^\top}\mathbf{w}_{i,d,t}^{(r)})$ is linear in $\mathbf{u}_{d,t}^{(r)}$ and μ_t, and thus is convex. The second term is convex, as its second derivative is positive semi-definite. To see this, let us set $\theta_{i,t} = \mu_t + g_{i,d,t}^{(-r)} + \mathbf{u}_{d,t}^{(r)^\top}\mathbf{w}_{i,d,t}^{(r)}$. The second derivative of the second term is $\frac{\partial^2 b(\mu_t + g_{i,d,t}^{(-r)} + \mathbf{u}_{d,t}^{(r)^\top}\mathbf{w}_{i,d,t}^{(r)})}{\partial \mathbf{u}_{d,t}^{(r)^2}} = \mathbf{w}_{i,d,t}^{(r)}\mathbf{w}_{i,d,t}^{(r)^\top}b''(\theta_{i,t})$. Here, $\mathbf{w}_{i,d,t}^{(r)}\mathbf{w}_{i,d,t}^{(r)^\top}$ is a rank-1 positive semi-definite matrix, and $b''(\theta_{i,t}) > 0$. Thus, the second term is convex with respect to $\mathbf{u}_{d,t}^{(r)}$. Similarly, one can show the convexity with respect to μ_t. By definition, norms are convex. Thus, the last three terms in the objective function are convex. As a result, the objective function (10) is convex with respect to both $\mathbf{u}_{d,t}^{(r)}$ and μ_t. Hence, the solution to (10) is unique, addressing the scaling indeterminacy problem in the estimation procedure.

The second source of nonidentifiability is the permutation indeterminacy: $\mathcal{B}_t = [[\mathbf{U}_{1,t}, \ldots, \mathbf{U}_{m,t}]] = [[\mathbf{U}_{1,t}\mathbf{\Pi}, \ldots, \mathbf{U}_{m,t}\mathbf{\Pi}]]$ where Π is a R-by-R permutation matrix. To solve the permutation indeterminacy of the tensor parametrization with $l_{2,1}$ norm penalty, we consider a specific permutation that rearranges the nonzero elements of the first row of $\mathbf{U}_{d,t}$ (i.e., $\mathbf{U}^{(r)}_{m,t,1} \neq 0$) in descending order such that $\mathbf{U}^{(1)}_{m,t,1} > \mathbf{U}^{(2)}_{m,t,1} > \ldots > \mathbf{U}^{(R-y)}_{m,t,1}$ and leaves the zero elements to the end of the ordering. Here, y is the number of zero elements. After each local update stage for each t, we employ this rule to permute the factor matrices.

Now, denote $\mathcal{V}$ as the aggregated model tensor with $\mathcal{V} = [[\mathbf{V}_1, \ldots, \mathbf{V}_m]]$ and $\mathbf{V}_d = [\mathbf{v}^{(1)}_d, \mathbf{v}^{(2)}_d, \ldots, \mathbf{v}^{(R)}_d] \in \mathbb{R}^{I_d \times R}, d = 1, \ldots, m$. Each $\mathbf{V}_d$ is obtained by $\mathbf{V}_d = \frac{\sum_{t=1}^{T} N_t \mathbf{U}_{d,t}}{\sum_{t=1}^{T} N_t}$. Suppose we permute $\mathbf{U}_{d,t}$ with a permutation matrix Π and obtain $\tilde{\mathbf{U}}_{d,t} = \mathbf{U}_{d,t}\mathbf{\Pi}$. Note that the permutation rule, $\mathbf{\Pi}$, is the same over all sites and determined by the aggregator. Suppose the factor matrices of the permuted aggregated model are obtained as:

$$\tilde{\mathbf{V}}_d = \frac{\sum_{t=1}^{T} N_t \tilde{\mathbf{U}}_{d,t}}{\sum_{t=1}^{T} N_t} = \frac{\sum_{t=1}^{T} N_t \boldsymbol{U}_{d,t}\mathbf{\Pi}}{\sum_{t=1}^{T} N_t} = \boldsymbol{V}_d\mathbf{\Pi}.$$

Let $\tilde{\mathcal{V}}$ denote the aggregated model tensor, where $\tilde{\mathcal{V}} = [[\tilde{\mathbf{V}}_1, \ldots, \tilde{\mathbf{V}}_m]]$. We show that $\langle \mathcal{V}, \mathcal{X}_{i,t} \rangle = \langle \tilde{\mathcal{V}}, \mathcal{X}_{i,t} \rangle$:

$$\begin{aligned}
\langle \tilde{\mathcal{V}}, \mathcal{X} \rangle &= \langle \tilde{\mathbf{V}}_d (\tilde{\mathbf{V}}_m \odot \ldots \tilde{\mathbf{V}}_{d+1} \odot \tilde{\mathbf{V}}_{d-1} \ldots \odot \tilde{\mathbf{V}}_1)^\top, \mathcal{X}_{i,(d),t} \rangle \\
&= \langle \boldsymbol{V}_d \Pi (\boldsymbol{V}_m \Pi \odot \ldots \boldsymbol{V}_{d+1} \Pi \odot \boldsymbol{V}_{d-1} \Pi \ldots \odot \boldsymbol{V}_1 \Pi)^\top, \mathcal{X}_{i,(d),t} \rangle \\
&= \langle \boldsymbol{V}_d \Pi (\boldsymbol{K}_{V_d} \Pi)^\top, \mathcal{X}_{i,(d),t} \rangle \\
&= \langle \boldsymbol{V}_d \Pi \Pi^\top \boldsymbol{K}_{V_d}^\top, \mathcal{X}_{i,(d),t} \rangle \\
&= \langle \boldsymbol{V}_d \boldsymbol{K}_{V_d}^\top, \mathcal{X}_{i,(d),t} \rangle \\
&= \langle \mathcal{V}, \mathcal{X}_{i,t} \rangle,
\end{aligned} \qquad [11]$$

where $\mathbf{K}_{V_d} = \mathbf{V}_m \odot \ldots \mathbf{V}_{d+1} \odot \mathbf{V}_{d-1} \ldots \odot \mathbf{V}_1$ and $\Pi\Pi^\top = \mathbf{I}_{R\times R}$. This concludes that the proposed tensor model is identifiable up to permutation and has a valid solution under a specific permutation rule.