7-2024

FedSTEM-ADL: A federated spatial-temporal episodic memory model for ADL prediction

Doudou WU
Singapore Management University, doudou.wu.2023@phdcs.smu.edu.sg

Shubham PATERIA
Singapore Management University, shubhamp@smu.edu.sg

Budhitama SUBAGDJA
Singapore Management University, budhitamas@smu.edu.sg

Ah-hwee TAN
Singapore Management University, ahtan@smu.edu.sg

Citation

WU, Doudou; PATERIA, Shubham; SUBAGDJA, Budhitama; and TAN, Ah-hwee. FedSTEM-ADL: A federated spatial-temporal episodic memory model for ADL prediction. (2024). *2024 International Joint Conference on Neural Networks (IJCNN), Yokohama, Japan, June 30 -July 5: Proceedings*. 1-8.
Available at: https://ink.library.smu.edu.sg/sis_research/9311

FedSTEM-ADL: A Federated Spatial-Temporal Episodic Memory Model for ADL Prediction

Doudou Wu*, Shubham Pateria, Budhitama Subagdja, Ah-Hwee Tan
School of Computing and Information Systems
Singapore Management University
Singapore
doudou.wu.2023@phdcs.smu.edu.sg, {shubhamp, budhitamas, ahtan}@smu.edu.sg

***Abstract*—Learning of Activities of Daily Living (ADLs) provides insights into an individual's habits, lifestyle, and well-being. However, it is crucial to address data privacy concerns in practical situations when learning the ADL routines of individuals. In this paper, we introduce FedSTEM-ADL, a federated spatial-temporal episodic memory model to address this privacy issue. FedSTEM-ADL utilizes a federation of Spatial-Temporal Episodic Memory for ADLs (STEM-ADL) for federated learning, wherein multiple local STEM-ADL models from individual users are combined into a global model while preserving the privacy of the original data. Specifically, each local model is designed to learn the spatio-temporal ADL routines of an individual user, representing them as ADL events and sequences of such events as episode patterns. The global model then integrates the local models without referring to the underlying individual data, thus addressing privacy concerns in multi-user ADL analysis. We conduct a series of experiments based on both pseudo and real-world multi-user ADL datasets. The results show that FedSTEM-ADL is able to learn global ADL models in an efficient manner and consistently outperforms the baseline models in the task of next ADL event prediction.**

***Index Terms*—Federated Learning, Activities of Daily Living (ADLs), Self-Organizing Model, Fusion ART, Next Event Prediction**

I. INTRODUCTION

Smart healthcare is a fast-growing field that can enhance people's lives by integrating technology into healthcare solutions [1]. One important aspect of this field is the study of people's daily self-care activities, for example, sleeping, eating, bathing, etc., referred to as Activities of Daily Living (ADLs) [2]. The study of ADLs involves ADL monitoring, recognition, and next activity prediction. However, ADL recognition and prediction involves the collection of personal data raising privacy concerns. Studying ADLs while preventing unauthorized access and misuse of personal data remains an open challenge. In this work, we adopt the federated learning approach [3] to address this challenge in the context of next activity prediction, which enables the learning of prediction models across multiple individuals without sharing private data.

One class of models suitable for ADL pattern learning and prediction is based on Adaptive Resonance Theory (ART) [4]–[6], a neural theory for self organization and adaptive learning. The self-organizing feature of ART-based models allows them to autonomously adjust their internal representations and create new categories to accommodate new patterns. For learning ADL routines in particular, STADLART [7] is an ART-based model that can integrate multimodal contextual information involving the time and space associated with the ADL performed. However, for tasks involving the prediction of subsequent activities, STADLART may not be directly applicable as the ADL patterns learned do not encode the order of the events in the episodes. A more recent model called Spatial-Temporal Episodic Memory for ADL (STEM-ADL) [8] addresses this issue by encoding ADL sequences with the gradient encoding scheme, which is initially introduced in EM-ART [9].

Combining the aforementioned strengths of federated learning for preserving data privacy and STEM-ADL's episodic memory encoding capability, we propose a novel federated learning approach designed to integrate the knowledge of STEM-ADL models trained across multiple individual clients for privacy-preserving ADL learning and prediction. This novel approach is named Federated STEM-ADL, or simply, FedSTEM-ADL. Specifically, FedSTEM-ADL extends existing fusion ART models [9], [10] with the federated learning methodology to merge multiple users' ADL routine models into a single global model without exposing private data. ADL routine learning is achieved through a gradient-based method proposed in EM-ART [9] that encodes the temporal order of people's ADL events and represents all the events in a day as episode patterns. This enables FedSTEM-ADL to perform *next ADL event* prediction tasks, which is useful in smart home-based healthcare applications.

Compared with popularly used federated learning models [11]–[13], FedSTEM-ADL is a lightweight model because it requires fewer parameters during training, which makes the model parameter transfer between the central server and individual participants cost-efficient. Thus, FedSTEM-ADL is especially suitable for applications where computational resources are constrained.

In summary, the main contributions of this work are as follows:

- We propose a novel model of federated spatial-temporal episodic memory named FedSTEM-ADL, which combines federated learning and the STEM-ADL model for ADL pattern learning and next event prediction.
- Following the federated learning framework, we present an method for transfering the knowledge encoded in the

local models to the global model without exposing the original data. Accordingly, the communication cost is much more reduced compared with existing models.
- We conduct extensive experiments to compare FedSTEM-ADL with alternative models based on the next event prediction task on real-world multi-user ADL datasets, which demonstrate the effectiveness and efficiency of FedSTEM-ADL.

The rest of this paper is organized as follows. Section II reviews the related work on federated learning and next event prediction. The preliminaries regarding the STEM-ADL model and algorithms are given in Section III. Our proposed approach and FedSTEM-ADL model are presented in Section IV. Section V reports and discusses the empirical experiments conducted. Lastly, section VI concludes the paper.

II. Related Work

A. Federated Learning

Early approaches to federated learning focused on methods for aggregating model updates from multiple devices. These methods included simple averaging and weighted averaging of model updates, as well as more sophisticated methods such as secure aggregation and differential privacy [11]. These methods aimed to provide accurate and secure aggregation of model updates while preserving data privacy.

Recent works have focused on distributed optimization algorithms for federated learning. These algorithms aim to optimize the global model by considering the local datasets and models on each device. Popular algorithms in this category include federated averaging, federated coordinate descent [12], and federated stochastic gradient descent. These algorithms have shown promising results in terms of accuracy and convergence, but they are limited by the communication overhead of aggregating model updates from multiple devices.

B. Next Event Prediction

Existing popular approaches to next event prediction of ADL include the use of rule-based systems such as decision trees, and machine learning methods such as Hidden Markov Models [14] and Support Vector Machines [15]. Such approaches are limited in their ability to handle complex and dynamic ADL patterns. Recently, deep learning approaches have been applied to the next event prediction of ADL. CNNs [16] have been used to extract spatial features from sensor data, while RNNs [17] have been used to capture temporal dependencies between ADL events. Long short-term memory (LSTM) [18] as an RNN-based model, is also widely used in next-event prediction. In [19], the authors proposed a prediction technique based on LSTM, the experiment's results were evaluated on a real dataset with sensor signals. Such deep learning approaches have demonstrated superior performance in comparison to traditional approaches, particularly in the prediction of complex and dynamic ADL patterns. Additionally, a federated version of LSTM has been proposed for ADL prediction [13]. Their results show that the centralized and federated approaches differ only slightly in terms of performance, showing the feasibility of federated learning for next-event prediction with enhanced privacy.

III. Preliminaries: STEM-ADL

Our proposed FedSTEM-ADL is composed of multiple local models and a global model, each of which is based on STEM-ADL [8]. As shown in Figure 1, the STEM-ADL network architecture comprises an episode field F_3, an event field F_2, and three input fields F_1^t, F_1^p, F_1^a for encoding input patterns representing the time, place, and ADL activity information, respectively. A summary of the STEM-ADL model is given below.

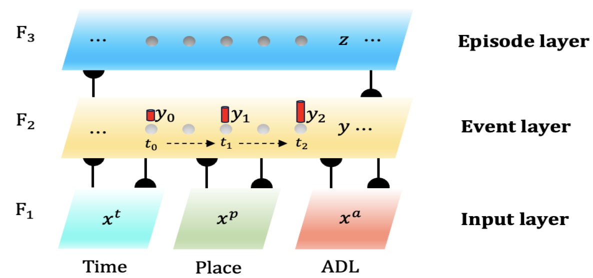

Fig. 1. The STEM-ADL model.

STEM-ADL performs learning of ADL episodes in two phases, namely event encoding and episode encoding. Each episode is a sequence of ADL events. Events are learned by the event nodes in the F_2 event field based on inputs received in the three input fields. Episode learning takes place in the F_3 field based on the events learned in the F_2 field. The key variables and the learning algorithm are presented as follows.

Input vectors: Let $\mathbf{I}^c = (I_1^c, I_2^c, \ldots, I_n^c)$ denote the input vector of an input field F_1^c, where $I_i^c \in [0,1]$ for $i \in [1,n]$ and $c \in \{t,p,a\}$. Using complement coding, each input vector is typically appended with a complementing vector $\overline{\mathbf{I}^c}$, where $\overline{I_i^c} = 1 - I_i^c$ for $i \in [1,n]$.

Firstly, the starting time and ending time are given as timestamps for each ADL. The time values are normalized by $\frac{(time\ in\ seconds)}{86400}$ since there are 86,400 seconds per day. Then, the normalized time input is encoded as

$$\mathbf{I}^t = (\mathbf{I_s}^t, \mathbf{I_e}^t, \overline{\mathbf{I_s}^t}, \overline{\mathbf{I_e}^t}), \tag{1}$$

where $\mathbf{I}_s$ and $\mathbf{I}_e$ denote the start time and end time of an ADL, respectively.

Secondly, each ADL also has an associated place vector $\mathbf{I}^p$ which represents the location of occurrence of the ADL, such as bedroom, kitchen, bathroom, etc. Formally, $\mathbf{I}^p$ is defined as follows

$$\mathbf{I}^p = (I_1^p, \ldots, I_P^p, \overline{I_1^p}, \ldots, \overline{I_P^p}), \tag{2}$$

where P is the number of places where the ADLs are performed.

Lastly, the type of ADL, such as eating, watching TV, sleeping, etc., is given as input vector $\mathbf{I}^a$ which is defined as follows

$$\mathbf{I}^a = (I_1^a, \ldots, I_A^a, \overline{I_1^a}, \ldots, \overline{I_A^a}), \tag{3}$$

where A is the number of ADL types.

Activation vectors: Let $\mathbf{x}^t = \mathbf{I}^t$, $\mathbf{x}^p = \mathbf{I}^p$, and $\mathbf{x}^a = \mathbf{I}^a$ denote the *activation vectors* of the input fields F_1^t, F_1^p, and F_1^a, respectively. Further, let $\mathbf{y} = (y_1, y_2, ..., y_M)$ denote the activation vector of the event field F_2, where M is the number of learned nodes in F_2 and $y_j \in [0, 1]$ is the activation value of an event node j. Finally, let $\mathbf{z} = (z_1, z_2, ..., z_S)$ denote the activation vector of the episode field F_3, where S is the number of learned episode nodes and $z_k \in [0, 1]$ is the activation value of an episode node k.

Weight vectors: Let $\mathbf{w}_j^c$ denote the weight vector of a node j in F_2 for learning the input received from F_1^c, where $c \in \{t, p, a\}$. Further, let $\mathbf{w}_k$ denote the weight vector of node k in F_3. Initially, both the F_2 and F_3 fields contain only one uncommitted node.

Parameters: Learning in F_2 layer corresponding to an input field c is governed by the choice parameter $\alpha^c > 1$, vigilance parameter $\rho^c \in [0, 1]$, learning rate parameter $\beta^c \in [0, 1]$, and contribution parameters $\gamma^c \in [0, 1]$. Similarly, in F_3 layer, the learning process is influenced by a set parameters: choice parameter $\alpha > 1$, vigilance parameter $\rho \in [0, 1]$, learning rate parameter $\beta \in [0, 1]$, and contribution parameters $\gamma \in [0, 1]$.

A. Encoding of Events

Given the input activation vectors $\mathbf{x}^c$ where $c \in \{t, p, a\}$, STEM-ADL performs a series of learning steps described as follows.

Code activation: The first step is to calculate a choice value T_j for each F_2 node j, as follows:

$$T_j = \Sigma_{c \in \{t,p,a\}}\, \gamma^c \frac{|\mathbf{x}^c \wedge \mathbf{w}_j^c|}{(\alpha_c + |\mathbf{w}_j^c|)}, \tag{4}$$

where the fuzzy AND operation $\wedge$ is defined by $(\mathbf{p} \wedge \mathbf{q})_i \equiv min(p_i, q_i)$, and the norm $|\cdot|$ is defined by $|\mathbf{p}| \equiv \Sigma_i |p_i|$ for vectors $\mathbf{p}$ and $\mathbf{q}$.

Code competition: In this step, the winner node J with the highest choice value is chosen as follows

$$T_J = max\{T_j : for\ all\ F_2\ node\ j\}. \tag{5}$$

When a winner node J is chosen, the F_2 activation values are set as $y_J = 1$ and $y_j = 0$, $\forall j \neq J$ (winner-take-all).

Template matching: The template matching process checks if the matching function of the winner node J satisfies the vigilance criterion set by the parameter ρ^c,

$$m_J^c = \frac{|\mathbf{x}^c \wedge \mathbf{w}_J^c|}{|\mathbf{x}^c|} \geq \rho^c. \tag{6}$$

If the vigilance constraint is not satisfied, a mismatch reset occurs such that T_J is set to zero and another F_2 node J is selected through code competition. This process is repeated until a node satisfies the vigilance constraints. If such a node is not found, a new category node is created in F_2.

Template learning: Once a selected node J satisfies the matching criteria and the resonance occurs, the weight vector $\mathbf{w}_J^c$ is modified by the following learning rule:

$$\mathbf{w}_J^{c'} = (1 - \beta^c)\mathbf{w}_J^c + \beta^c(\mathbf{x}^c \wedge \mathbf{w}_J^c), \tag{7}$$

where $\mathbf{w}_J^{c'}$ is the updated weight vector of node J in F_2 layer connecting to F_1^c field.

Activity readout: Any chosen F_2 node j can perform a top-down readout of weight vectors to the input field F_1^c such that $\mathbf{x}^{c'} = \mathbf{w}_j^{c'}$, where $\mathbf{x}^{c'}$ is the updated activation vector of F_1^c field.

B. Encoding of Episodes

An ADL episode (or daily routine) refers to a series of spatio-temporal events that happen in a day. Each node in the F_3 layer of STEM-ADL represents a unique episode and it is learned from the sequence of events encoded in the F_2 layer. The episode learning process in STEM-ADL is described below.

Let successive timestamps be represented as $t_0, t_1, t_2, \ldots, t_n$, in increasing order, and $y_j^{t_i}$ denote the activation value of the event node j in F_2 at time t_i. Then, $y_j^{t_0} < y_j^{t_1} < ... < y_j^{t_n}$, meaning that the activation value of an event is higher if it takes place later in time.

When a new node is activated in F_2, it has the maximum activation value equal to *one*, and the activation values of previously created nodes are decayed in each time step, so that

$$y_j^{(new)} = y_j^{(old)}(1 - \tau), \tag{8}$$

where y_j' is the updated activation value of node j in this step and τ denotes a predefined decay coefficient and $\tau \in (0, 1)$.

Given the activation vector $\mathbf{y} = (y_1, y_2, ..., y_M)$, assuming there are M nodes in F_2, STEM-ADL performs a series of learning steps for episode encoding similar to those described for event encoding in subsection III-A, as follows:

Code activation: Given a F_2 activation vector $\mathbf{y}$, for each F_3 node k, a choice value T_k is computed as follows:

$$T_k = \frac{|\mathbf{y} \wedge \mathbf{w}_k|}{(\alpha + |\mathbf{w}_k|)}. \tag{9}$$

Code competition: In the code competition procedure, the winner node K is the one with the highest choice value, as follows

$$T_K = max\{T_k : for\ all\ F_3\ node\ k\}. \tag{10}$$

When a category choice is made at node K, the F_3 activation values are set as $z_K = 1$ and $z_k = 0$, $\forall k \neq K$.

Template matching: The template matching process checks if the matching function of the winner node satisfies the vigilance requirement, as follows

$$m_K = \frac{|\mathbf{y} \wedge \mathbf{w}_K|}{|\mathbf{y}|} \geq \rho \tag{11}$$

If the vigilance constraint is not satisfied, a mismatch reset occurs so that T_K is set to zero and another F_3 node is selected. If the vigilance constraints are not satisfied by any selected node in F_3, a new category node is added in F_3.

Template learning: Once a node K is selected and the resonance occurs, the weight vector $\mathbf{w}_K$ is modified by the following learning rule:

$$\mathbf{w}_K^{new} = (1 - \beta^k)\mathbf{w}_K^{old} + \beta^k(\mathbf{y} \wedge \mathbf{w}_K^{old}). \tag{12}$$

Activity readout: The chosen F_3 node K may perform a top-down readout of weight vectors to the event field F_2 such that $\mathbf{y}^{new} = \mathbf{w}_K$.

IV. Proposed Method

As illustrated in Figure 2, FedSTEM-ADL consists of multiple local models on the client side and a single global model on the server side. Each local model in FedSTEM-ADL is a three-layered STEM-ADL network, as described in section III. The server-side global model learned using federated learning also consists of a similar three-layered STEM-ADL architecture. The main processes in FedSTEM-ADL are as follows:

1) The server initializes the global model and sends the local model configuration to all clients, which includes the number of epochs of training as well as choice parameters α, vigilance parameters ρ, learning rate parameters β, and contribution parameters γ. Hence, all client-side local models can be trained with the same configuration.
2) Each client performs local model training based on its local ADL data using the process described in section III. It then sends the weight templates of the learned F_2 and F_3 nodes through activity readout to the server.
3) The central server aggregates the weight templates of the F_2 and F_3 nodes submitted by local clients and updates the global model (refer to subsection IV-A).

A. Global Model Aggregation

After the local model training process is completed, the global model receives the weight vectors readout from F_2 nodes of local models one by one, with a flag that indicates the start and end of one episode, and performs the training process until it converges. The overall global model aggregation is described as follows.

a) Readout and Transfer of Event Nodes: We denote the five fields of a local model l as $F_1^{t,l}$, $F_1^{p,l}$, $F_1^{a,l}$, F_2^l, and F_3^l. Similarly, the fields of the global model G are denoted as $F_1^{t,G}$, $F_1^{p,G}$, $F_1^{a,G}$, F_2^G, and F_3^G, respectively. During the readout process, the activation vector $\mathbf{y}_k = (y_1, y_2, ..., y_M)$ is readout from F_3^l to F_2^l layer. While any component $y_j \in \mathbf{y}_k$ is greater than zero, the highest y_j is selected and the corresponding weight vector $\mathbf{w}_j$ is read out to the $F_1^{t,l}$, $F_1^{p,l}$, $F_1^{a,l}$ fields. Then, the event node with the second highest activation $y_j \in \mathbf{y}_k$ is readout, and so on. This process yields a sequence of abstract ADL event vectors ($\mathbf{w}_j$) which are transferred to the global model. The readout process is repeated for all the episode nodes k in the F_3^l field and for all local models $l \in [1, L]$.

b) Global Model Learning of Event Nodes: Upon the transfer of node k's weights to the F_1^G input layer, the global model initiates a series of learning steps for the encoding of an event node as elaborated in subsection III-A, including bottom-up code activation, code competition, and template matching followed by template learning when a resonance is achieved. This comprehensive sequence of learning steps culminates in the formation or updation of activation vector $\mathbf{y}^G = (y_1^G, y_2^G, ..., y_{M'}^G)$ in the F_G^2 event field, where M' is the number of learned event nodes in the global model and $y_j^G \in [0, 1]$ for event node j.

c) Global Model Learning of Episode Nodes: Given the activation vector $\mathbf{y}^G$, the global model performs a series of episode learning processes between F_2^G and F_3^G fields as elaborated in subsection III-B including bottom-up code activation, code competition, template matching followed by template learning when resonance is achieved, and finally the global model learns an episode node in F_3^G field.

Specifically, during the local training process, we try to preserve every unique ADL routine to the highest extent by using a relatively higher vigilance parameter value in both F_2^l and F_3^l field. However, during the global model aggregation process, since we try to obtain a global model that encompasses features from different clients, a lower vigilance is used in the F_3^G field to generalize the episode nodes from different local models. The detailed learning process is presented in Algorithm 1.

Algorithm 1 Global Model Aggregation

Require: Local models from 1 to L, initial global model G,
Ensure: Learning an aggregated global model G

for each local model l **do**
  for each episode node k in the F_3^l field **do**
    Readout $\mathbf{y}_k = (y_1, y_2, ..., y_M)$ to F_2^l layer
    while there is any $y_j > 0$ in $\mathbf{y}_k$ **do**
      Select event node j with highest activation y_j
      Readout $\mathbf{w}_j$ to the $F_1^{t,l}$, $F_1^{p,l}$, and $F_1^{a,l}$ fields
      Transfer $\mathbf{w}_j$ to input fields of global model
      Reset $y_j \leftarrow 0$
      compute the choice values of all event nodes in F_2^G (see Eq. (4))
      select node J in F_2^G with highest choice value as winner node (see Eq. (5))
      Compute activation vector $\mathbf{y}^G = (y_1^G, y_2^G, ..., y_{M'}^G)$ in the F_2^G layer
      Perform the template matching and template learning processes (see Eq. (6) and Eq. (7))
      Select the winning node J in F_2^G
      Update all the activated values of vector $\mathbf{y}^G$ of F_2^G such that $y_j^{(new)} = y_j^{(old)}(1-\tau)$ (see Eq. (8))
      Set $y_J \leftarrow 1$
    end while
    Given updated activation vector $\mathbf{y}^G$, compute the choice values of all nodes in F_3^G (see Eq. (9))
    Select winner node K with highest choice value in F_3^G (see Eq. (10)) as the global aggregated episode
    Perform the template matching and template learning processes (see Eq. (11) and Eq. (12)) for node K
  end for
end for

In Figure 3, a simplified example of global model aggregation is presented. This process involves the integration of two distinct local models (denoted as l_1 and l_2) into a global model. Specifically, l_1 is composed of two episodes: E_1 containing

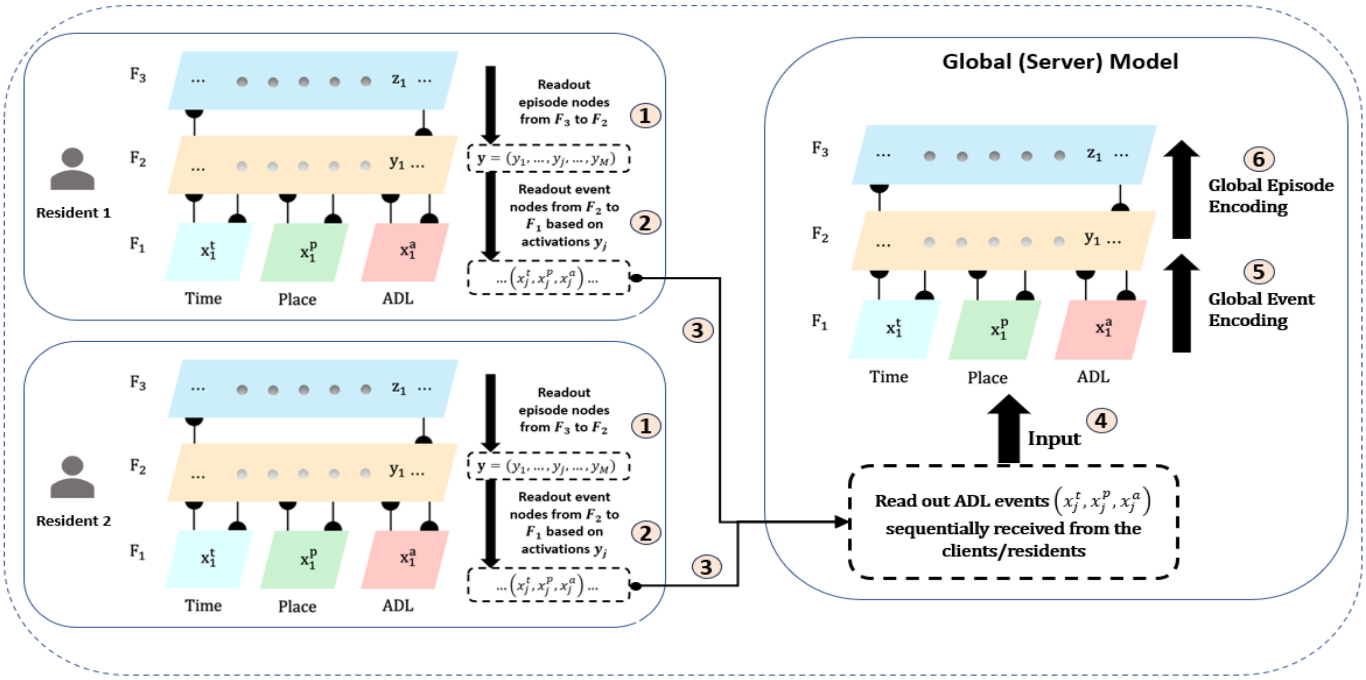

Fig. 2. The framework of FedSTEM-ADL. The global model aggregation process is shown using the sequence of steps marked from 1 to 6.

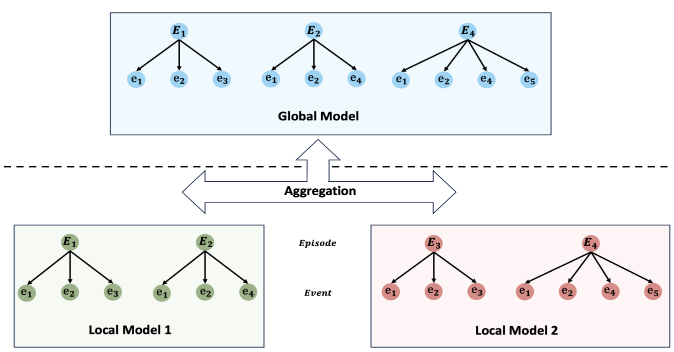

Fig. 3. An illustration on how the ADL sequences learned by multiple local models can be transferred and aggregated by the global model.

three events (e_1, e_2 and e_3) and E_2 containing three events (e_1, e_2 and e_4). Similarly, l_2 is composed of two episodes: E_3 containing three events (e_1, e_2 and e_3) and E_4 containing four events (e_1, e_2, e_4 and e_5). A notable observation is the congruence between episode E_1 in l_1 and episode E_3 in l_2. During the aggregation phase, our methodology focuses on combining similar episodes while retaining distinct ones, and the preservation of event sequences within each episode. Consequently, the aggregated global model is composed of E_1, E_2, and E_4.

B. Next Event Prediction

After the global model is trained to aggregate the event and episode nodes from the local models, it can be used to perform the next event prediction task. In the three-layer global model, each event has an associated pair of starting and ending time stamps. Consequently, a temporal sequence of events encoded in an episode can be used to predict the next event from the previous ones. For such next event prediction, we need to represent the temporal order of events encoded in the F_2^G layer of the global model with gradient encoding [9]. The procedure for next event prediction is as follows:

1) The trained FedSTEM-ADL global model takes $\mathbf{x}_{cur} = (\mathbf{x}^t, \mathbf{x}^p, \mathbf{x}^a)$ as the current input ADL.
2) The event node J in the F_2^G layer that best matches $\mathbf{x}_{cur}$ is selected based on a winner-take-all strategy (Eq (5)). Based on the single node selected in F_2^G as the cue for retrieving the episodes, the choice value of episode nodes in F_3^G can then be calculated based on Eq (9). However, multiple episode nodes in F_3^G might have the same highest choice value since only one event (matching $\mathbf{x}_{cur}$) is used as the cue and it might be contained in multiple encoded episodes. Thus, multiple episodes matching $\mathbf{x}_{cur}$ might be selected.
3) The readout vector of a selected episode node in F_3^G may contain the actual activation value of the current event. Let y_J^G be the activation value corresponding to the current event. Based on Eq (8), the activation value y_{j*}^G of the next event within each episode can be calculated as

$$y_{j*}^G = \frac{y_J^G}{1-\tau}. \tag{13}$$

All nodes j^* in the readout vector(s) of the selected node(s) in F_3^G with an activation value equal to y_{j*}^G form a candidate next event set $\mathbb{B} = \{j_1^*, j_2^*, ..., j_B^*\}$. Then, event vectors $(\mathbf{x}_b^t, \mathbf{x}_b^a, \mathbf{x}_b^p)$ can be readout corresponding to each $j_b^* \in \mathbb{B}$.

4) Among the multiple candidate ADL events, the winning one is selected as follows. Firstly, the ADL type $\mathbf{x}_b^a$ that occurs most frequently in $\mathbb{B}$ (time and place might vary) is identified and all the events containing $\mathbf{x}_b^a$ are shortlisted. Subsequently, among the shortlisted events, the one whose start time has the least difference from the end time of the current event $\mathbf{x}_{cur}$ is selected as the single winning next event.

V. Experiments

Our first set of experiments is conducted based on a public domain ADL dataset called Orange4Home[1] which contains daily ADL sequences performed by a single user. We simulated three users from this data set by splitting the dataset into three roughly equal segments. As the Orange4Home dataset was collected based on a single subject, who adhered to a prescribed lifestyle pattern, it is deemed as a clean ADL dataset. The experiments conducted on this dataset thus effectively evaluate the various federated models' ability to integrate the daily routines learned by the local models and perform the next event prediction.

The second experiment is conducted on a real-world dataset called TWOR[2] that records the daily ADL sequences of two users. This is to evaluate the capacity of FedSTEM-ADL to predict the next event using noisy ADL data produced by different users. We specifically opted for these two datasets since they provide ADL records over an extended duration, ranging from dozens to hundreds of days. The choices in the ADL domain are notably limited as such datasets are rare, underscoring the challenging nature of ADL research.

Federated versions of LSTM [18] and RNN [17], namely Fed-LSTM and Fed-RNN, are chosen as the baseline methods for comparison with FedSTEM-ADL. The recently introduced FedMA algorithm [20] is used for global model aggregation for both methods since it has shown better performance than traditional FedAvg [11] owing to its matched averaging approach. We implement the LSTM and RNN models underlying Fed-LSTM and Fed-RNN, respectively, to learn to predict both the next ADL and its start time (dual output) compared to the previous work using federated LSTM that only predicts the next ADL [13]. The hyper-parameter settings for these models are done similarly to that reported in STEM-ADL [8] work. These include setting the number of neurons to 128, training for 300 epochs, using a batch size of 64, employing "MSE" as the loss function, and "Adam" as the optimizer. Other parameters of both the baseline models were iteratively tuned.

A. Experiments on Orange4Home

a) Dataset: The Orange4Home dataset records the ADLs performed by a single user over 20 successive workdays with full annotations of ADLs, comprising the starting timestamp, ending timestamp, the place information, and the associated activity type. Table I lists the segmentation and distribution of the data in the Orange4Home dataset. As mentioned in section III, the time vector is normalized and complement coded.

[1]https://amiqual4home.inria.fr/orange4home/

[2]https://casas.wsu.edu/datasets/twor.2010.zip

TABLE I
Orange4Home dataset with three simulated users.

	Resident 1	**Resident 2**	**Resident 3**	**Total**
Days	7	7	6	20
Events	183	178	132	493
Place Type	Entrance, Living room, Kitchen, Bedroom, Office, Bathroom, Toilet, Staircase			
ADL Type	Entering, Reading, Napping, Cleaning, Eating, Watching TV, Computing, Going up, Going down, Dressing, Leaving, Using the sink, Using the toilet, Showering, Preparing, Cooking, Washing the dishes			

TABLE II
Next event prediction performance of various non-federated and federated learning models on Orange4Home.

		Accuracy	**F1 score**	**MAE**
Local Model (w/o FL)	LSTM	0.887±0.082	**0.889**±0.018	**9.67**
	RNN	0.817±0.071	0.832±0.103	12.31
	STEM-ADL	**0.892**±0.037	0.881±0.052	10.89
Global Model (with FL)	Fed-LSTM	0.898±0.054	0.901±0.094	**5.53**
	Fed-RNN	0.884±0.070	0.912±0.067	10.02
	FedSTEM-ADL	**0.917**±0.034	**0.916**±0.072	7.58

b) Experiment Settings: From each local dataset, two days are randomly selected as the test data. In total, 14 days of ADL data are used for training and 6 days are used for testing. For fair comparison, all local models and global models in our experiments use the same train-test split in the experiments.

As a key learning parameter of STEM-ADL, the vigilance parameters ρ in the F_2 and F_3 layers specify the minimum degree of match required for resonance and learning. For ease of discussion, we denote the vigilance parameter values for event encoding and episode encoding in the global model as $\rho^{e(G)}$ and $\rho^{s(G)}$ respectively and in a similar manner, as $\rho^{e(l)}$ for event encoding and $\rho^{s(l)}$ for episode encoding in the local models. For the other parameters, a standard set of parameter values was used as follows for both local model training and global model aggregation:

1) For event encoding between F_1 and F_2, the parameters for each of the three input fields were set as choice parameters α = 0.001 to maximize code compression, contribution parameters γ = 0.333 for equal input contributions, and learning rate β = 1 for fast learning.
2) For episode encoding between F_2 and F_3, the parameters for the only input channel were set as α = 0.001, γ = 1, β = 1. The delay coefficient τ was set to 0.1.

c) Performance Measures: As discussed in section IV-B, FedSTEM-ADL is designed to predict both the ADL type and the duration of the ADL performed. For ADL-type prediction, accuracy and F-score are used as the performance measures. Besides ADL type prediction, we also compare the performance of the predicted starting time of the next events by computing mean average error (MAE) [8], which is the mean absolute difference between the ground-truth starting time and predicted starting time of next ADL events.

Through the utilization of fuzzy AND operations and complement coding, FedSTEM-ADL learns generalized event nodes which can provide the start time intervals during which users initiate specific activities. However, for comparison with the Fed-LSTM and Fed-RNN models that produce a single timestamp prediction, we take the middle of the predicted starting time interval in our model as the single predicted time value. Specifically, let t_{gi} denote the ground-truth starting time of the ith next event, and $t_i = [t_{s1i}, t_{s2i}]$ denote the predicted starting time interval of the ith next event, the MAE is computed as

$$MAE = \frac{1}{H}\sum_{i=1}^{H} |t_{gi} - \text{mean}(t_{s1i}, t_{s2i})|, \tag{14}$$

where H denotes the number of next event predictions made.

d) Results & Analysis: The performance of FedSTEM-ADL compared with those of Fed-LSTM and Fed-RNN is shown in Table II. For non-federated scenarios, the LSTM model produced a slightly better F1 score and lower MAE compared to STEM-ADL. However, in the federated setting, the FedSTEM-ADL model produced the best accuracy and F1 scores in predicting the type of the next events, while Fed-LSTM incurred the least MAE in predicting the starting time of the next events. Therefore, we can say that FedSTEM-ADL is at least as competitive as Fed-LSTM in next event prediction task for this data set. For FedSTEM-ADL, the best result is observed when the local models are trained with $\rho^{e(l)}$ = 0.99 and $\rho^{s(l)}$ = 1.0, in conjunction with $\rho^{e(G)}$ = 0.99 and $\rho^{s(G)}$ = 1.0 used in the global model. This indicates that the FedSTEM-ADL model achieves better performance when allowing a small degree of generalization on the event nodes encoded in the F_2 field of both the local models and global model.

B. Experiments using Real Multi-user Dataset

a) Dataset: The TWOR dataset is one of the CASAS data sets, which consists of a total of 38 data sets that contain sensory data collected from well-equipped test beds in several cities across the world. The TWOR data set records the ADLs performed by two users over 250 consecutive workdays with full annotations.

Table III summarizes the data characteristics and distribution of the TWOR data set. The TWOR dataset includes seven place labels and twelve ADL types. The number of days covered in the dataset as well as the total length of the ADLs sequences are much longer compared to the Orange4Home dataset. Also, the number of ADLs sequences contained in each day varies significantly, making it difficult to learning an ADL pattern. This is the most notable difference between TWOR and Orange4Home.

b) Experiment Settings: Similarly, the test set is also randomly selected from each of the two local user datasets. The training to test ratio is fixed at 0.7 to 0.3. For all experiments, the global model and local models use the same test data. The learning parameters of all models, including local model training and global model aggregation, were set as the same as those used for Orange4Home.

TABLE III
KEY STATISTICS OF TWOR DATA SET.

	Resident 1	**Resident 2**
Days	250	250
Events	1813	1764
Place Type	kitchen, living room, toilet, staircase, walkway, bathroom, bedroom	
ADL Type	Bathing, Bed Toilet Transition, Eating, Enter Home, Leave Home, Meal Preparation, Personal Hygiene, Sleep, Sleeping Not in Bed, Wandering in room, Watch TV, Work	

TABLE IV
NEXT EVENT PREDICTION PERFORMANCE OF VARIOUS NON-FEDERATED AND FEDERATED LEARNING MODELS ON TWOR.

		Accuracy	**F1 score**	**MAE**
Local Model (w/o FL)	LSTM	**0.516**±0.049	**0.523**±0.012	**45.24**
	RNN	0.482±0.092	0.478±0.075	58.66
	STEM-ADL	0.502±0.030	0.481±0.098	48.45
Global Model (with FL)	Fed-LSTM	0.533±0.107	0.529±0.074	41.26
	Fed-RNN	0.496±0.118	0.488±0.02	52.85
	FedSTEM-ADL	**0.547**±0.013	**0.536**±0.025	**40.47**

c) Results & Analysis: Based on the TWOR dataset, we evaluate the performance of FedSTEM-ADL against Fed-RNN and Fed-LSTM in predicting the type and starting time of the next events. Again, the best performance of FedSTEM-ADL is observed when the local models are trained with $\rho^{e(l)}$ = 0.99 and $\rho^{s(l)}$ = 1.0 in local models, in conjunction with $\rho^{e(G)}$ = 0.99 and $\rho^{s(G)}$ = 1.0 in the global model. This outcome aligns with the findings from the prior experiment, indicating that FedSTEM-ADL performs best when it applies moderate generalization in encoding event nodes in the local and global models.

As shown in Table IV, though LSTM, as the local model, produced the best performance across all measures, the FedSTEM-ADL model can significantly improve upon the performance of STEM-ADL (as the local model) and produced the best performance in predicting both the type and the starting time. The results show that the unique structure and learning dynamics of the ART models brings strong robustness in global model aggregation, which enables FedSTEM-ADL to achieve superior performance and stability in this prediction task. Comparing with the results obtained for Orange4Home, the impact of our proposed federated learning approach appears to be most profound in integrating the diverse ADL patterns provided by distinct users into the global model.

C. Comparison of Learning Efficiency

In terms of efficiency, FedSTEM-ADL requires significantly less computing time for training compared with Fed-RNN and Fed-LSTM, as shown in Table V. Specifically, due to

TABLE V

COMPARISON OF TRAINING TIME (IN MINUTES) TAKEN BY VARIOUS MODELS ON ORANGE4HOME AND TWOR.

	Local Model (w/o FL)			Global Model (with FL)		
	LSTM	RNN	STEM-ADL	Fed-LSTM	Fed-RNN	FedSTEM-ADL
Orange4Home	17.74	11.32	**5.65**	29.13	21.77	**12.19**
TWOR	29.41	25.01	**9.38**	39.45	31.12	**17.67**

the fast and stable learning characteristics of the ART-based models, FedSTEM-ADL converges quickly on the spatial-temporal ADL data. While the computational time advantage of FedSTEM-ADL is not as significant for smaller data sets such as Orange4Home, it is more evident on the TWOR dataset. This is a huge advantage of FedSTEM-ADL to dealing with very large data sets.

VI. CONCLUSION

In this study, we propose FedSTEM-ADL, a novel approach for aggregating daily behavior patterns of individual users to a global model through federated learning. Specifically, FedSTEM-ADL enables the transfer of the model parameters between local models and the global model without sharing the original data. Additionally, its intrinsic structural characteristics significantly reduce the communication cost compared with the existing approaches. Our extensive experiments have demonstrated that FedSTEM-ADL can achieve superior ADL event prediction performance and protect users' data privacy both on clean and noisy datasets. Those abilities make FedSTEM-ADL an ideal choice for ADL modeling applications in smart home healthcare and health-assistive services.

Regarding future work, a potential direction for enhancing the FedSTEM-ADL framework involves incorporating federated personalization. By leveraging techniques such as user-specific fine-tuning, context-aware feature selection, and personalized aggregation strategies, the FedSTEM-ADL framework can be extended to adapt to individual preferences and behaviors, leading to more accurate and tailored results for personalized healthcare applications. Additionally, exploring the integration of fusion ART and reinforcement learning algorithms could further enhance the framework's ability to adapt and optimize its performance over time, especially in dynamic healthcare environments.

ACKNOWLEDGMENT

This research was supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG2-RP-2020-019), the SMU-A*STAR Joint Lab in Social and Human-Centered Computing (Grant No. SAJL-2022-HAS001), and the Jubilee Technology Fellowship awarded to Ah-Hwee Tan by Singapore Management University.

REFERENCES

[1] S. Tian, W. Yang, J. M. Le Grange, P. Wang, W. Huang, and Z. Ye, "Smart healthcare: making medical care more intelligent," Global Health Journal, vol. 3, no. 3, pp. 62–65, 2019.

[2] S. Katz, A. B. Ford, R. W. Moskowitz, B. A. Jackson, and M. W. Jaffe, "Studies of illness in the aged. the index of ADL: a standardized measure of biological and psychosocial function," Journal of American Medical Association(JAMA), vol. 185, no. 12, pp. 914–919, 1963.

[3] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," IEEE Signal Processing Magazine, vol. 37, no. 3, pp. 50–60, 2020.

[4] S. Grossberg, "Adaptive resonance theory: How a brain learns to consciously attend, learn, and recognize a changing world," Neural Networks, vol. 37, pp. 1–47, 2013.

[5] G. Carpenter, S. Grossberg, and D. Rosen, "Fuzzy art: an adaptive resonance algorithm for rapid, stable classification of analog patterns," in IJCNN-91-Seattle International Joint Conference on Neural Networks, vol. ii, pp. 411–416 vol.2, 1991.

[6] A.-H. Tan and H.-S. V. Soon, "Concept hierarchy memory model: A neural architecture for conceptual knowledge representation, learning, and commonsense reasoning," International Journal of Neural Systems, vol. 7, no. 03, pp. 305–319, 1996.

[7] S. Gao, A.-H. Tan, and R. Setchi, "Learning ADL daily routines with spatiotemporal neural networks," IEEE Transactions on Knowledge and Data Engineering, vol. 33, no. 1, pp. 143–153, 2019.

[8] X. Song, D. Wang, C. Quek, A.-H. Tan, and Y. Wang, "Spatial-temporal episodic memory modeling for ADLs: encoding, retrieval, and prediction," Complex & Intelligent Systems, pp. 1–18, 2023.

[9] W. Wang, B. Subagdja, A.-H. Tan, and J. A. Starzyk, "Neural modeling of episodic memory: Encoding, retrieval, and forgetting," IEEE Transactions on Neural Networks and Learning Systems, vol. 23, no. 10, pp. 1574–1586, 2012.

[10] A.-H. Tan, G. A. Carpenter, and S. Grossberg, "Intelligence through interaction: Towards a unified theory for learning," in International Symposium on Neural Networks, pp. 1094–1103, Springer, 2007.

[11] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA, vol. 54, pp. 1273–1282, PMLR, 2017.

[12] X. Leng, C. Li, W. Xu, Y. Sun, and H. Wang, "Federated coordinate descent for privacy-preserving multiparty linear regression," arXiv preprint arXiv:2209.07702, 2022.

[13] S. Zehtabian, S. Khodadadeh, L. Bölöni, and D. Turgut, "Privacy-preserving learning of human activity predictors in smart environments," in IEEE INFOCOM 2021-IEEE Conference on Computer Communications, pp. 1–10, IEEE, 2021.

[14] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," Proceedings of the IEEE, vol. 77, no. 2, pp. 257–286, 1989.

[15] W.-Y. Lin, C.-H. Chen, Y.-J. Tseng, Y.-T. Tsai, C.-Y. Chang, H.-Y. Wang, and C.-K. Chen, "Predicting post-stroke activities of daily living through a machine learning-based approach on initiating rehabilitation," International Journal of Medical Informatics, vol. 111, pp. 159–164, 2018.

[16] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.

[17] J. L. Elman, "Finding structure in time," Cognitive science, vol. 14, no. 2, pp. 179–211, 1990.

[18] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 2010.

[19] C. Wang and Z. Peng, "Deep learning model for human activity recognition and prediction in smart homes," in 2020 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS), pp. 741–744, IEEE, 2020.

[20] H. Wang, M. Yurochkin, Y. Sun, D. S. Papailiopoulos, and Y. Khazaeni, "Federated learning with matched averaging," in 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020, OpenReview.net, 2020.