Sparsity-Aware Intelligent Spatiotemporal Data Sensing for Energy Harvesting IoT System

Wen Zhang, Mimi Xie, *Member, IEEE*, Caleb Scott, and Chen Pan, *Member, IEEE*

***Abstract*—In this era of the Internet of Things (IoT), the increasing number of IoT devices benefit from energy harvesting (EH) technology which enables a sustainable data acquisition process, including data sensing, communication, and storing for promoting the well beings of the society. However, intermittent and low EH power confines the data acquisition process. Specifically, due to frequent harvesting power outages and depletion of energy for expensive data transmission to the IoT edge server, insufficient energy is allocated for data sensing resulting in the missing of key information. To address this issue, this article proposes a sparsity-aware spatiotemporal data sensing framework for EH IoT devices to minimize the data sensing rate/energy while acquiring comprehensive information and reserving sufficient energy. In this framework, the IoT devices sample critical sparse spatiotemporal data, and then the sparse data are sent to the edge server for reconstruction. To maximize the reconstruction accuracy subject to the limited power supply and intermittent work patterns of EH devices, we first propose the QR-based algorithm QR-ST to initiate a sensing scheduling for each EH device. Due to the unstable and intermittent work pattern, the schedule needs to be dynamically fine-tuned based on environmental inputs. Therefore, we further propose a multiagent deep reinforcement learning-based method named *S-Agents* for the IoT edge server to globally select the sensing devices at each time slot, where the spatial and temporal features of reconstructed data are guaranteed. Experimental results show that the proposed framework reduced the reconstruction error by 66.30% compared with baselines.**

***Index Terms*—Compressed sensing, energy harvesting (EH), Internet of Things (IoT), reinforcement learning.**

I. Introduction

IN THIS era of Internet of Things (IoT), a plethora of embedded devices are spatially deployed to continuously retrieve data of interest in specific regions. However, due to typically being powered by batteries, those embedded devices are subjected to limited lifetime and high maintenance costs. Energy harvesting (EH) technology that scavenges energy from the ambient environment becomes a viable substitute for the conventional battery by providing a sustainable power supply for embedded devices [1], [2]. Such upgrades can enable perpetual low-power sensing for various next generations of IoT applications. However, the weak and intermittent nature of EH power confines the performance of data sensing. Specifically, to capture the important spatiotemporal features, the EH devices at both specific locations and times need to conduct data sensing. Yet, due to the harvesting power shortage, a power outage may happen and the EH device will miss sensing key spatiotemporal data. Moreover, due to the expensive power consumption for communication, transmitting sensed data to the edge server will deplete the harvested energy rendering insufficient energy for sensing key spatiotemporal data [3], [4]. The situation will get worse when the spatiotemporal data are with huge size such as 4-D spatiotemporal weather data [5]. Therefore, it is a grand challenge to sense and transmit massive data with the EH IoT system.

To address this issue, we observed that most spatiotemporal data are compressible and the key features can be captured with sparsely sampling at both spatial and temporal domains. In this background, by applying compressed sensing theory [6], the aforementioned challenge of data sensing in the EH IoT system can be naturally addressed by inferring the missing data with the reconstruction values. To be more specific, temporally, for each EH IoT device, data sensing is only needed at sporadic moments. Spatially, at each moment, only a tiny subset of EH embedded devices are activated to obtain the sparse data. Once these pieces of sparse data are sent to the edge server, the whole map of observations can be reconstructed. Consequently, not only the data absences caused by the intermittent harvesting power can be compensated but also enough energy can be reserved for communication.

Despite the advantages of compressed sensing, it is extremely challenging to seek an optimal sensing activation schedule for EH devices. On the one hand, if we activate too many sensors frequently for improving reconstruction accuracy, the harvested energy is wasted for sensing redundant data. On the other hand, if the sensed data are too sparse, the reconstruction accuracy cannot be guaranteed. Therefore, it is crucial to determine the optimal amount of spatiotemporal data. Based on the compressed sensing, this problem is equivalent to searching the sparse measurement matrix through an l_0 norm optimization, which is a complex NP problem. However, for the heuristic solution, we can alternatively select the sensing strategy on sensing time and locations. Yet such optimization is a nonconvex problem that is still complex.

To address the aforementioned problem, this article leverages multiagent deep reinforcement learning (DRL) for developing a sparsity-aware spatiotemporal data sensing framework for EH systems. The sensing among spatially distributed EH

Manuscript received 21 July 2022; accepted 26 July 2022. Date of publication 9 August 2022; date of current version 24 October 2022. This work was supported by NSF under Grant 2153524. This article was presented at the International Conference on Embedded Software (EMSOFT) 2022 and appeared as part of the ESWEEK-TCAD special issue. This article was recommended by Associate Editor A. K. Coskun. *(Corresponding author: Chen Pan.)*

Wen Zhang and Chen Pan are with the Department of Computer Science, Texas A&M University at Corpus Christi, Corpus Christi, TX 78412 USA (e-mail: wzhang3@islander.tamucc.edu; chen.pan@tamucc.edu).

Mimi Xie and Caleb Scott are with the Department of Computer Science, University of Texas at San Antonio, San Antonio, TX 78249 USA (e-mail: mimi.xie@utsa.edu; caleb.scott@my.utsa.edu).

Digital Object Identifier 10.1109/TCAD.2022.3197543

devices can be formulated as the partially observable Markov decision process (POMDP) and the complicated intermittent feature of EH devices can be adapted by the strength of the neural network. This framework aims to schedule sensing operations for EH IoT devices in both spatial and temporal domains. Our key idea includes two phases, the *suggestion* phase and *runtime* phase. In the *suggestion* phase, motivated by [7], we first generate an initial schedule for each device by exploring the spatial and temporal features of historical data with the proposed QR-ST algorithm. After that, in the *runtime* phase, the developed DRL-based agent (*S-Agents*) deployed on each device will take the suggested schedule and EH status as inputs to determine the runtime sensing schedule dynamically to withstand the weak and intermittent harvesting power supply. To avoid stucking at local optimal, during the training period, we train DRL agents distributively with global rewards that reflect the reconstruction error. To precisely measure the reconstruction error between the reconstructed spatiotemporal data and the real spatiotemporal data, the multiresolution distance-based measurement is adopted. The main contributions of this article are summarized as follows.

1) We propose a comprehensive spatiotemporal data sensing framework for the EH IoT system, which consists of the *suggestion* and the *runtime* phases.
2) To ensure a small DRL model size while considering the limited computation capability of EH devices, we propose QR-ST to find determined scheduling of data sensing in the *suggestion* phase.
3) To distributively schedule sensing operations considering both spatial and temporal perspectives, we designed a multiagent DRL-based method named *S-Agents* to decide sensing operation dynamically subject to the limited power supply.

To the best of our knowledge, this article is the first work that proposes a sparsity-aware multiagent DRL-based spatiotemporal data sensing framework for EH systems.

II. Related Work

The existing works reconstruct data based on either only temporal or only spatial features.

Temporal: In [8], ADMM is adopted for activation scheduling of network sensors in periodical time to maximize the reconstruction accuracy given the constraints on the available number of activation time points for each device. Because we are reconstructing the sparse data from a set of devices in the network, the activated sensor can be scheduled to optimize the local reconstruction data or to optimize the global reconstruction through collaborative sensing. Jamali-Rad *et al.* [9] and Liu *et al.* [10] explored sensor selection in a distributed and collaborative manner, respectively. Intuitively, the distributed strategies may get trapped into local optima. But if we adopt the global strategies, while the network scale is large, finding a global optimal schedule is difficult and communication-intensive. Without the consideration of power supply restrictions, most researches [8], [9], [10] are targeting battery-powered devices. Calvo-Fullana *et al.* [11] investigated the joint optimization of sensor selection and power allocation for the "EH" sensing system. In [11], the nonconvex optimization problem is separated into a series of surrogate convex problems. Wang *et al.* [12] estimated the sparsity order of high-dimensional signals, where an illustration application is given by developing a two-step compressive spectrum sensing algorithm for cognitive radios. Wang *et al.* [13] also utilized the two-stage method to reduce the slow-time dimension of the signal.

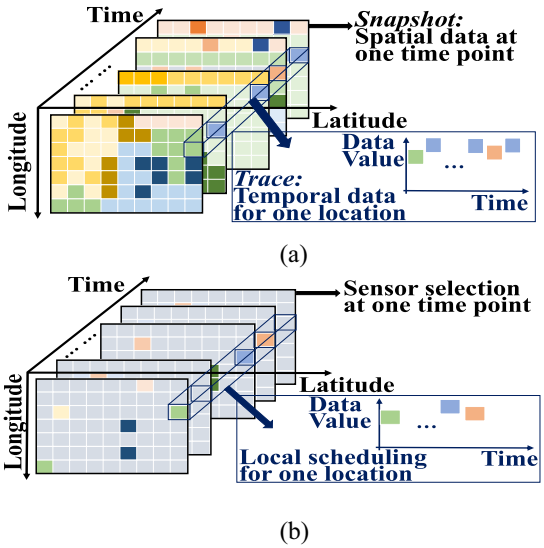

Fig. 1. Configuration of the (a) raw spatiotemporal data and (b) sparse data through compressed sensing.

Spatial: Masazade *et al.* [14] investigated the target tracking problem with an extended Kalman filter, where only a few sensors are activated and send their collected data to the data center to save energy. Clark *et al.* [15] proposed a column-pivoted QR decomposition to optimize the sensor selection scheme for full-state data reconstruction. Without consideration of cost constraints, Clark *et al.* [16] utilized the greedy algorithm to optimize the sensor placement while applying the sparsity-aware sensing to the facial recognition, climate science, and fluid mechanics. Instead of using the traditional universal basis in compressed sensing, Manohar *et al.* [7] investigated the full-state data reconstruction while using the tailored basis that is the eigenmodes extracted from the historical database. To explore EH-aware sparsity sensing, Calvo-Fullana *et al.* [17] proposed a novel sensor selection scheme for networks equipped with EH sensing devices. It minimizes the reconstruction distortion while selecting reduced but informative sensors.

Although great efforts have been devoted to sparsity-aware sensing, no existing work has investigated reconstructing the full-state data from both temporal and spatial features. Different from the existing work, this article proposes sparsity-aware spatiotemporal compressed sensing for EH IoT systems based on the multiagent DRL-based method.

III. Motivation of Spatiotemporal Sensing

Fig. 1(a) visualizes an example of the raw spatiotemporal data sensed by a 9×7 grid of devices at different latitudes and longitudes, where different snapshots represent the spatially distributed data at different time points. For each snapshot, the cell in the grid indicates the data of one device.

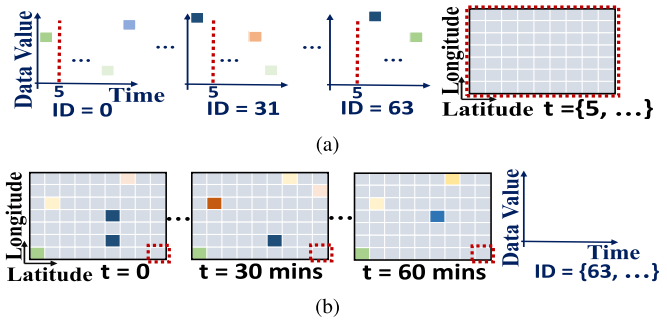

Fig. 2. Visualization of the sparsity-aware sensing (a) from temporal features and (b) from spatial features.

The data at the same location in different snapshots represent temporal data of one device named trace. Therefore, the spatiotemporal data are composed of a set of *snapshots* at continuous time points or a set of *traces* at spatially distributed locations, as shown in Fig. 1(a). The snapshots capture the spatial features at continuous time points and the traces capture the temporal features at spatially distributed locations. Sparsity-aware sensing aims to select the activated sensors and schedule the activation time to obtain the sparse data as shown in Fig. 1(b) to capture the critical spatial and temporal features for reconstructing Fig. 1(a) from Fig. 1(b) with the minimum reconstruction error. Specifically, a gray cell indicates that the corresponding device is inactivated, which results in that data's absence at a specific location and time point.

Spatiotemporal data have special properties since they consist of not only the spatial features in snapshots but also the temporal features in the traces. If we directly reconstruct the traces for each device or reconstruct the snapshots at each time point for the spatiotemporal data reconstruction, the spatial or temporal features cannot be captured, respectively. This problem is illustrated with the following two examples.

An example of data reconstruction from the trace is shown in Fig. 2(a), where the device needs to figure out what time the device should be activated to sense the sparse data of its local trace so as to reconstruct the local trace data. In this scenario, in order to maximize the trace reconstruction accuracy, the devices mainly target on the points of time when data collection has more contribution to reconstruction. Therefore, the devices are actually finding the local optimal sensing scheduler to collect the local temporal sparse data in a distributed manner while not considering the sensing data of other devices. After each device completes a reconstruction with maximum accuracy through temporal sparse data, it is highly possible that in a set of specific time points, all sensors are inactivated such as the time point 5 highlighted with a red dotted line in Fig. 2(a). If this case is applied in spatiotemporal data reconstruction, the spatial features of a snapshot at specific time points cannot be guaranteed in the sense that no data are measured for these snapshots, as indicated by the right square grid of Fig. 2(a).

An example of data reconstruction from the snapshot spatial features is shown in Fig. 2(b), where the networked devices mainly focus on which devices are activated for sensing at a given time point in a collaborative manner. Because the spatial data is location-related as shown in Fig. 2(b), a set of devices at certain locations that have larger contributions to spatial data reconstruction are frequently activated to minimize the reconstruction error, which results in the rare activation of certain devices [e.g., the device located at the cell highlighted with a dotted red square box in Fig. 2(b)]. Although each snapshot can be reconstructed with minimum error, the temporal features of those devices that are never activated cannot be captured, because those devices have never sensed any data, as indicated in the right subfigure of Fig. 2(b).

Therefore, the sparsity-aware sensing of spatiotemporal data requires consideration of not only the local sensing schedule of each device for temporal data reconstruction from a long-term perspective but also the selection of the activated devices at each time point for global spatial data sensing.

IV. Fundamental of Compressed Sensing

Compressed sensing is a well-known technology for efficient data acquisition, which reconstructs the high-dimensional original signal data from a small number of sparse data. Most natural data, such as images or audio, can be written as a sparse vector in a transform coordinate system, where only a few coefficient parameters are active to record the large mode amplitudes of signals. Those coefficients describe the necessary features for accurate reconstruction. Mathematically, given an original signal data as $\mathbf{x} \in \mathbb{R}^n$ and a new coordinate basis as $\Psi \in \mathbb{R}^{n\times n}$, $\mathbf{x}$ can be identified as a sparse vector $\mathbf{s} \in \mathbb{R}^n$, given by (1). Specifically, Ψ is a transform basis such as Fourier transform basis or wavelet basis

$$\mathbf{x} = \Psi\mathbf{s}. \tag{1}$$

In traditional compressed sensing application such as image compression, the high-resolution data $\mathbf{x}$ is first obtained and then the corresponding sparse vector $\mathbf{s}$ is computed. After that, to alleviate the communication constraints, the transmitter sends $\mathbf{s}$ to the receiver. The high resolution $\mathbf{x}$ can be obtained by calculating (1) in receiver. Instead of measuring an original full-state data $\mathbf{x}$ and then discarding most of its information by calculating $\mathbf{s}$ to alleviate the constraints on communication resources, the sparsity-aware sampling directly measures a subset of $\mathbf{x}$, defined as $\mathbf{y} \in \mathbb{R}^m, m << n$. Assume the measurement matrix is $\mathbf{C} \in \mathbf{R}^{m\times n}$ to identify the measurement positions, the sparsity-aware sampling can be described with (2). Once we compute $\mathbf{s}$, we can reconstruct $\mathbf{x}$ by (1)

$$\mathbf{y} = \mathbf{C}\Psi\mathbf{s}. \tag{2}$$

While $m < n$, a set of $\mathbf{s}$ satisfies (2). Therefore, $\mathbf{s}$ is undetermined. To reduce the sampling cost, we aim to find the sparsest $\mathbf{s}$ denoted as $\mathbf{s}^*$, which is given in

$$\mathbf{s}^* = \underset{\mathbf{s}}{\operatorname{argmin}} \|\mathbf{s}\|_0, \quad \text{subject to: } \mathbf{y} = \mathbf{C}\Psi\mathbf{s} \tag{3}$$

where $\|\mathbf{s}\|_0$ is the l_0 norm of $\mathbf{s}$. The l_0 norm optimization is an NP problem; therefore, instead of optimizing the l_0 norm, Donoho [18] proposed to solve l_1 while the random sampling matrix $\mathbf{C}\Psi$ satisfies the so-called restricted isometry property (RIP) [6] conditions. In this case, we can also obtain the sparsity solution. Therefore, the l_0 minimization is transformed to the l_1 minimization problem as

$$\mathbf{s}^* = \underset{\mathbf{s}}{\operatorname{argmin}} \|\mathbf{s}\|_1, \quad \text{subject to: } \mathbf{y} = \mathbf{C}\Psi\mathbf{s}. \tag{4}$$

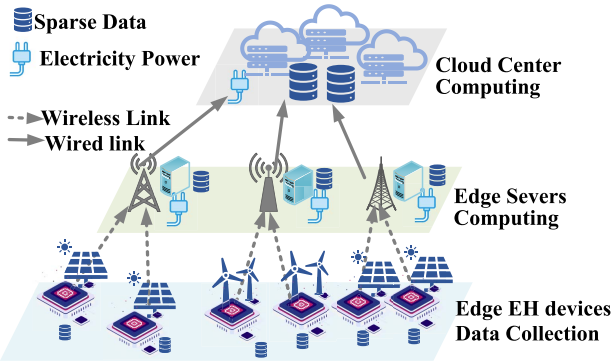

Fig. 3. System model.

From (2)–(4), if we know the measurement matrix, we can figure out the sampling locations and time points from both spatial and temporal perspectives. Following the sampling locations and time points, we can sense $\mathbf{y}$ to reconstruct $\mathbf{x}$. However, due to the difficulties in computing RIP parameters, it is hard to solve the exact measurement matrix $\mathbf{C}\Psi$.

V. System Model

A. Scenario Description

Fig. 3 shows a three-layer edge system for spatiotemporal data collection, which consists of three layers, including the cloud center on the top layer, multiple edge servers in the edge layer, and edge devices layer. In the bottom layer, the edge devices harvest energy from the environment to provide power supply for onboard data sensing, local computing, and communication. The edge devices sense and transmit the data to the edge server for preprocessing by the wireless link. In the middle layer and top layer, both the edge servers and the cloud center are powered by electricity. Also, the edge server is connected to the cloud center through a wired link. Specifically, in the middle layer, there is no communication among edge servers because of the geographic limitations and the limited transmission range.

This article considers a sparsity-aware edge data collection scenario, where the EH devices sense data with a much smaller measurement frequency than the typical full-state edge data collection scenario. Given a constant time interval as v seconds, per v seconds, each device decides if it will sense or not for the current time interval. Due to the limited energy and capability on computing, the EH devices transmit the collected data to the corresponding edge server for reconstruction. Because the edge servers lack the completed view of the data from all edge devices, the edge servers can only assist in reconstructing the local sparse data from the temporal dimension (local traces). To ensure the spatial features are captured, the sparse data is backed up from all edge servers to the cloud center, the cloud center will then reconstruct the final spatiotemporal data for each time interval and calculate the reconstruction errors.

B. Sensing System

In the bottom layer, the K EH devices are deployed with the index set of $\mathcal{K} = \{1, \ldots, k, \ldots, K\}$. The location of device k is notified with $[\text{lat}_k, \text{lon}_k]$. The system periodic operation time is slotted to T time slots labeled with $\mathcal{T} = \{1, \ldots, t, \ldots, T\}$. Therefore, the sensed data[1] of k are defined as $\mathbf{y}_k = [y_k[1], \ldots, y_k[t], \ldots, y_k[T]]', k \in \mathcal{K}, t \in \mathcal{T}, \mathbf{y}_k \in \mathbb{R}^T$. The spatiotemporal data for the whole system at the cloud center are represented by $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_K]$. In particular, since the device senses sparse data, most elements in the $\mathbf{Y}$ matrix are zeros.

C. Energy Model

The energy consumption of EH devices k consists of three components, including sensing energy consumption, communication energy consumption for transmitting data to edge servers, and computing energy consumption for S-Agent running to control the sensing. Thus, the total energy consumption of k is given by

$$E_{k,\text{cost}} = E_k^{\text{sense}} b^{\text{sense}} + p_k^{\text{trans}} \frac{b^{\text{trans}}}{\omega_{k,\text{trans}}} + \kappa_k b^{\text{compu}} c_k f_k^2 \quad (5)$$

where E_k^{sense} is the energy consumption for k to sense one bit data and b^{sense} represents the number of bits on sensing data. Denote p_k^{trans}, b^{trans}, and $\omega_{k,\text{trans}}$ as transmission power, the number of transmission bits, and transmission rate, respectively, where the transmission rate ω can be retrieved by the Shannon theory [19]. $\kappa_k b^{\text{compu}} c_k f_k^2$ is the energy consumption on computing b^{compu} bits data. κ_k is the hardware parameter of k. c_k and f_k represent the needed number of CPU cycles and the computing CPU frequency of k [20].

Given at t the power intensity of k as $p_{k,t}$, the harvested energy is equal to $E_{k,\text{harvest}} = \int p_{k,t} dt$. Therefore, the remaining energy of k at t time slot is (6). Note that $E_{k,\text{rest}}[t] \leq E_{k,\max} \ \ \forall t \in \mathcal{T}$ has to be satisfied, where $E_{k,\max}$ is the energy capacity of k

$$E_{k,\text{rest}}[t] = E_{k,\text{rest}}[t-1] + E_{k,\text{harvest}}[t] - E_{k,\text{cost}}[t]. \quad (6)$$

VI. Problem Formulation

In this section, we aim to minimize the errors in the spatiotemporal data reconstruction subject to the limited power supply and intermittent work pattern of EH devices in the EH IoT system. To formulate the reconstruction minimization problem, we first introduce the required notations. $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_K]$ is the reconstructed data matrix for the corresponding sparse data $\mathbf{Y}$. $\mathbf{L}$ is the label data of $\mathbf{X}$. It means while devices sense with the regular frequency, the collected data would be $\mathbf{L}$ (the real data). Therefore, this article aims to minimize the distance between the reconstructed data $\mathbf{X}$ and the real data $\mathbf{Y}$. However, the spatiotemporal data is a new data type. Using the Euclidean distance calculation cannot precisely capture the similarity/difference of spatial and temporal features between two spatiotemporal data scenarios proved by [5]. Therefore, instead of minimizing the Euclidean distance of $\mathbf{X}$ and $\mathbf{L}$, we employ the multiresolution distance [5], [21] to measure the similarity/difference.

[1]$'$denotes vector or matrix transpose operation.

A. Distance Measurement on Spatiotemporal Data

The multiresolution distance-based measurement is developed in [5] for the difference/similarity measurement on spatiotemporal scenario data, whose effectiveness is ensured by [21] and [22]. It calculates the difference of two spatiotemporal data in multiresolution, where the spatiotemporal data are scanned and compared with a set of size varying windows. Given $\mathbf{L}$ and $\mathbf{X}$ as the real and reconstructed spatiotemporal scenario data consisting of a set of snapshots at the continuous time points as Fig. 1(a) and (b) indicated. Define G as the sensing area and $g_k \in G$ as the cell located in $[\text{lat}_k, \text{lon}_k]$. $\phi_{k,w} \in \Phi_w$ is the square spatial window centered at g_k with size w and Φ_w is the set of spatial windows with size w in the space G. $\phi_{t,h} \in \Phi_h$ is the temporal window *starting* from the instant t with size h and Φ_h is the set of temporal windows with size h in the time space $\mathcal{T}$. Therefore, while the size of spatiotemporal moving window is w and h, the distance of two scenarios $d_{\mathbf{X},\mathbf{L},w,h}$ is given by the following:

$$d_{\mathbf{X},\mathbf{L},w,h} = \sum_{\phi_{k,w}\in\Phi_w} \sum_{\phi_{t,h}\in\Phi_h} \frac{1}{|\phi_{k,w}||\phi_{t,h}||\Phi_h|} \Delta_{\mathbf{XL}} \tag{7}$$

$$\Delta_{\mathbf{XL}} = \left| \sum_{g_z\in\phi_{k,w}} \sum_{q\in\phi_{t,h}} \frac{x_z[q]}{\lambda_{z,w}\tau_{q,h}} - \sum_{g_z\in\phi_{k,w}} \sum_{q\in\phi_{t,h}} \frac{l_z[q]}{\lambda_{z,w}\tau_{q,h}} \right| \tag{8}$$

where $|\cdot|$ is the cardinality of the corresponding set. $\lambda_{z,w}$ and $\tau_{q,h}$ are the contribution factors to eliminate the border effect on the distance calculation. The cells or time points at the boundary are counted less than those at the center area. The distance of $\mathbf{X}$ and $\mathbf{L}$ is given by (9) that is the weighted sum of $d_{\mathbf{X},\mathbf{L},w,h}$ obtained at different spatiotemporal window size

$$D_{\mathbf{X},\mathbf{L}} = \sum_{h=1}^{h_{\max}} \sum_{w=1}^{w_{\max}} d_{\mathbf{X},\mathbf{L},w,h} \frac{\delta_w \alpha_h}{\sum_{h=1}^{h_{\max}} \sum_{w=1}^{w_{\max}} \delta_w \alpha_h} \tag{9}$$

where $w_{\max}$ and $h_{\max}$ are the maximum spatial and temporal window sizes, respectively. δ_w and α_h are weighting factors that control the contributions of different spatial and temporal resolutions to the distance calculation.

B. Optimization Goal

For the EH IoT system, each device k senses spatiotemporal data in area G with the given power intensity $p_{k,t}$. Let $\mathbf{a}_k$ be the vector that represents if the device k performs sensing or not at each time slot. The scheduler of device k is defined as

$$\mathbf{a}_k = \begin{bmatrix} a_k[1] \\ \cdots \\ a_k[t] \\ \cdots \\ a_k[T] \end{bmatrix} k \in \mathcal{K}, t \in \mathcal{T}, \mathbf{a}_k \in \{0, 1\}. \tag{10}$$

Our objective is to seek the device scheduler $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_K]$ that minimizes the reconstruction error on the spatiotemporal data $D_{\mathbf{X},\mathbf{L}}$. Mathematically

$$\begin{aligned} \underset{\mathbf{A}}{\text{argmin}} \quad & \mathrm{D}_{\mathbf{X},\mathbf{L}} \\ \text{subject to} \quad & E_{k,\text{rest}} \geq 0 \\ & E_{k,\text{rest}} \leq E_{k,\max}. \end{aligned}$$

VII. Algorithm Design

A. Architecture of Spatiotemporal Data Sensing

To address the challenge of limited computing, communication, and storage resources of EH devices, sparsity-aware spatiotemporal data sensing in EH IoT system is developed that address the conflicts of high demand on high-dimensional data, the limited capability of EH devices, and limited lifetime of battery-powered devices. The framework of sparsity-aware spatiotemporal data sensing for the EH IoT system is shown in Fig. 4, which is composed of two phases, including the *suggestion* phase and the runtime phase.

In the *suggestion* phase, motivated by [7], we propose the QR-ST algorithm, which adopts QR pivot factorization [7], [16], [23] as foundation to find the global sensor selection from the *Spatial* features of the historical snapshots at each time point. Meanwhile, we also conduct the local sensing scheduling from the Temporal features of the historical data traces for each device. There are suggestions from both spatial and temporal perspectives. When either *global* sensor selection or *local* scheduling requests the sensor to perform sensing, the final suggestion that is input into S-Agent is "Yes." Recall that the spatiotemporal data consist of either a set of the snapshots at different time points (spatial features) or a set of data traces from different devices (temporal features), whose examples are described in Section III and Fig. 1. Specifically, with more and more data stored in the historical database, we can update the coherent spatial and temporal modes (Ψ_{spl} and Ψ_{tpl}) per T_U time with new historical data. Correspondingly, the determining schedules are updated for each T_U time periodically. The mathematical explanation of Ψ_{spl} and Ψ_{tpl} is elaborated in Section VII-B.

Given the historical spatiotemporal data, the spatial modes of snapshots and the temporal modes of data traces are determined. Therefore, the sensor selections and local schedules calculated by the QR-ST algorithm are fixed. However, without consideration of the variation in the complicated EH environment, the fixed sensing operations are not environment adaptive and might lead to poor reconstruction performance. For example, the selected sensors cannot perform the sensing task as expected caused by that the selected sensors are sleeping due to the weak harvesting power, which leads to the necessary reconstruction data being missing. To dynamically fine-tune the scheduling of sparsity-aware sensing, we further develop the DRL-based method for the EH IoT system.

In the *training/runtime* phase, with the determined global sensor selection and local schedule as the suggestions, the developed DRL-based agent named S-Agent is deployed on each device to dynamically control the sensing operation at each time point subject to the limited power supply of EH devices. After that, the sensed sparse data are transmitted to the corresponding edge server and backed up at the cloud center. Once all edge servers complete data back-up, the cloud center utilizing the transform basis $\Psi_{\text{spl}}[t]$ reconstructs the snapshot at the current time point. The spatiotemporal reconstruction error is broadcasted to all edge servers. It is worth noting that, in the training stage, we have the historical full-state real data to calculate the reconstruction errors. However, in the runtime stage, if we perform sparsity-aware sensing all the time, we do

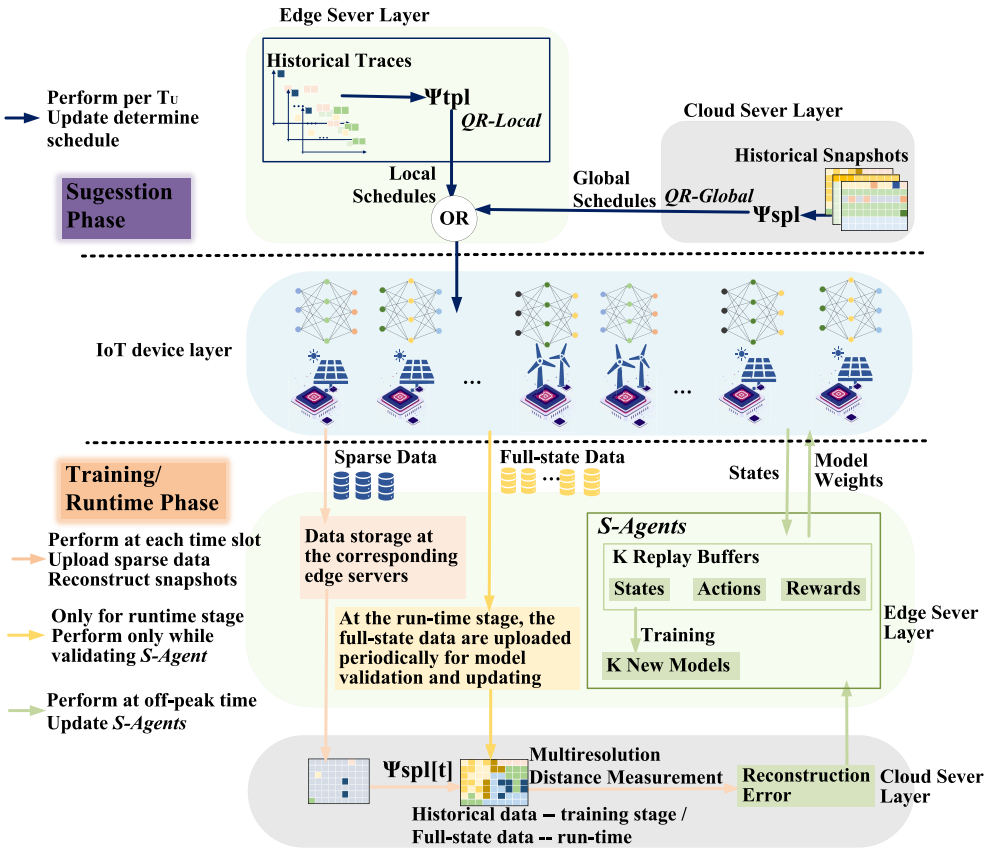

Fig. 4. Framework of sparsity-aware spatiotemporal compressed data sensing in the EH IoT system.

not have the full-state real data to calculate the reconstruction errors. Consequently, we cannot evaluate the performance during runtime. Therefore, during runtime, the full-state data are collected and uploaded periodically for the trained model validation and updating. In particular, to train S-Agent and fully utilize the computation resource at the edge, the experiences of S-Agent are uploaded to the corresponding edge server for learning at the off-peak time.

B. Design on the Suggestion Phase: QR-ST

The compressed sensing technology is widely used to recover the data of unknown content with random measurement on a universal basis Ψ. When the data type information is accessible such that we are measuring the soil humility or the data is a solar power intensity, the main features can be extracted from the historical data of representative cases to a tailored basis. Therefore, the proposed QR-ST algorithm utilizes *QR* pivot factorization to find the fixed global sensor selection and local sensing scheduling by extracting *Spatial* modes (Ψ_{spl}) and *Temporal* modes (Ψ_{tpl}) of historical spatiotemporal data. From the spatial dimension, we can extract the spatial features from the historical snapshots collected at a certain time point t that indicate the coherent spatial modes of the snapshots at the t time points (line 4 in Algorithm 1). Thereby, the global sensor selection at each time point is determined based on the spatial modes of historical snapshots, where we can activate sensors that have more contributions to the snapshot information (line 5 in Algorithm 1). Similarly, by analyzing the temporal features and the temporal modes for the historical data traces of each device k (line 10 in Algorithm 1), we can obtain the determined local schedule for each device k (line 11 in Algorithm 1).

Algorithm 1: QR-ST

Input: $\mathbf{L}^i, \forall i \in \mathcal{I} = \{1, 2, \ldots, I\}, r_s, r_t, T, K$; /* $\mathbf{L}^i$The historical spatiotemporal data */

Output: $\Psi_{spl}[t], \forall t \in \mathcal{T}, \Psi_{tpl,k}, \forall k \in \mathcal{K}, \Xi^*, \Gamma^*$;

1 // From Spatial dimension
2 **for** $t \leftarrow 1$ **to** T **do**
3 Retrieve $\mathbf{L}[t]$ from $\mathbf{L}^i, \forall i \in \mathcal{I}$;
4 $\Psi_{spl}[t], S, V = svd(\mathbf{L}[t])$; /* SVD and qr are the existing packages in MATLAB or Python */
5 $\mathbf{Q}, \mathbf{R}, \Xi[t]^* = qr(\Psi_{spl}(:, 1 : r_s)')$;
6 **end**
7 // From Temporal dimension
8 **for** $k \leftarrow 1$ **to** K **do**
9 Retrieve $\mathbf{L}_k$ from $\mathbf{L}^i, \forall i \in \mathcal{I}$;
10 $\Psi_{tpl,k}, S, V = svd(\mathbf{L}_k)$;
11 $\mathbf{Q}, \mathbf{R}, \Gamma_k^* = qr(\Psi_{tpl}(:, 1 : r_t)')$;
12 **end**
13 **return** $\langle \Psi_{spl}[t] \rangle$, $\langle \Psi_{tpl,k} \rangle$,$\langle \Xi[t]^* \rangle$, $\langle \Gamma_k^* \rangle$

Global Sensor Selection From Spatial Modes: Proper orthogonal decomposition (POD) is a typical method used for dimension reduction, which proposes to represent the high-dimensional data as linear combinations of a small number of orthonormal eigenmodes. Singular value decomposition

(SVD) proposed in [24] is an efficient algorithm to retrieve the dominant modes from data. Define the flatted snapshots as $\mathbf{l}[t] = [l_1[t], \ldots, l_k[t], \ldots, l_K[t]', \mathbf{l}[t] \in \mathbb{R}^K, t \in \mathcal{T} \quad \forall k \in \mathcal{K}$. Let $\mathbf{l}_i[t]$ be the ith snapshot of t time point in the historical snapshots database, so the matrix of historical snapshots at t time point is $\mathbf{L}[t] = [\mathbf{l}_1[t], \mathbf{l}_2[t], \ldots, \mathbf{l}_I[t]]$. Then, the eigenmodes of $\mathbf{L}[t]$ denoted as $\Psi_{\text{spl}}[t]$ can be the orthonormal left singular vectors, which are obtained efficiently by SVD and implemented in most computing software packages (line 4 in Algorithm 1). Given the tailored basis as $\Psi_{\text{spl}}[t] \in \mathbb{R}^{K \times r_s}$ as the orthonormal eigenmodes of snapshots at t time point, according to the compressed sensing technology (1), we have

$$\mathbf{l}[t] = \Psi_{\text{spl}}[t]\mathbf{z}_s \tag{11}$$

where $\mathbf{z}_s \in \mathbb{R}^{r_s}$, $r_s << K$ is the POD coefficients that is used for representing $\mathbf{l}[t]$ in $\Psi_{\text{spl}}[t]$ coordinator.

From spatial (global) perspective, we aim to select a portion of sensors from $\mathcal{K}$ at time point t. When we only measure at the selected sensors, we can recover $\mathbf{l}[t]$. Mathematically, define the set of selected sensors at t as $\Xi[t] = \{\xi_1, \xi_2, \ldots, \xi_\kappa\}$, $\Xi[t] \in \mathcal{K} \quad \forall t \in \mathcal{T}$. Therefore, we aim to reconstruct the snapshot $\mathbf{l}[t]$ from the sensed sparse data $\mathbf{y}[t] = [l_{\xi_1}, l_{\xi_2}, \ldots, l_{\xi_\kappa}]'$. Define the measurement matrix as $\mathbf{C}$. Based on (11), we have

$$\mathbf{y}[t] = \mathbf{C}\mathbf{l}[t] = \mathbf{C}\Psi_{\text{spl}}[t]\mathbf{z}_s \tag{12}$$

where each row of $\mathbf{C}$ is the canonical basis vector for $\mathbb{R}^K$ with the unit entry index at ξ column and zeros at all other entries. Recall that $\mathbf{x}[t]$ denotes the reconstructed data of $\mathbf{l}[t]$. Thus, we can reconstruct $\mathbf{l}[t]$ by (13)[2]

$$\mathbf{x}[t] = \Psi_{\text{spl}}[t]\mathbf{z}_s, \text{ where } \mathbf{z}_s = \begin{cases} (\mathbf{C}\Psi_{\text{spl}}[t])^{-1}\mathbf{y}[t], & \kappa = r_s \\ (\mathbf{C}\Psi_{\text{spl}}[t])^{\dagger}\mathbf{y}[t], & \kappa > r_s \end{cases} \tag{13}$$

where κ is the cardinality of $\Xi[t]$. Because r_s affects the structure of $\mathbf{C}\Psi_{\text{spl}}[t]$, only when $\kappa = r_s$, $(\mathbf{C}\Psi_{\text{spl}}[t])^{-1}$ exists. Thus, if $\kappa > r_s$, we adopt the Moore–Penrose pseudoinverse.

Therefore, to capture more features in $\Psi_{\text{spl}}[t]$, the global sensor selection optimizes $\Xi[t]$ $(\mathbf{C}\Psi_{\text{spl}}[t])$ to maximize the product of its eigenvalue as

$$\Xi[t]^* = \underset{\Xi[t] \quad \forall t \in \mathcal{T}}{\text{argmax}} |\det \mathbf{C}\Psi_{\text{spl}}[t]|. \tag{14}$$

Through the matrix QR factorization with column decomposition (line 5 in Algorithm 1), r sensors locations are conducted from the basis mode $\Psi_{\text{spl}}[t]$ by

$$\Psi'_{\text{spl}}[t]\mathbf{C}' = \mathbf{Q}\mathbf{R} \tag{15}$$

where $\mathbf{Q}$, $\mathbf{R}$, and $\mathbf{C}$ are a unitary matrix, an upper-triangular matrix, and a column permutation matrix, respectively, that are decomposed from matrix $\Psi'_{\text{spl}}[t]$. Through the obtained measurement matrix $\mathbf{C}$, Ξ can be retrieved. Repeat to compute $\Xi[t]$ for each time slot, the determined global sensor selection $\Xi^* = \{\Xi[t]^* \quad \forall t \in \mathcal{T}\}$ for each time slot is calculated.

Local Sensing Scheduling From Temporal Modes: To capture the features in temporal dimension, we also train the $\Psi_{tpl,k} \in \mathbb{R}^{T \times r_t}$ as the orthonormal eigenmodes of the traces at location k with the historical temporal traces $\mathbf{L}_k = [\mathbf{l}_{1,k}, \mathbf{l}_{2,k}, \ldots, \mathbf{l}_{I,k}]$ (line 10 in Algorithm 1). Similarly, define $\mathbf{x}_k = [x_k[1], x_k[2], \ldots, x_k[T]]', k \in \mathcal{K}$ as the reconstructed traces at k and $\mathbf{l}_k$ being the real trace of k. By seeking the optimal local schedule $\Gamma_k = \{\gamma_1, \gamma_2, \ldots, \gamma_\zeta\}, \Gamma_k \in \mathcal{T}$ to maximize (16) for each device k (line 11 in Algorithm 1), we then can reconstruct $\mathbf{x}_k \quad \forall k \in \mathcal{K}$ via (17)

$$\Gamma_k^* = \underset{\Gamma_k \quad \forall k \in \mathcal{K}}{\text{argmax}} |\det \mathbf{C}\Psi_{\text{tpl}}| \tag{16}$$

$$\mathbf{x}_k = \Psi_{\text{tpl}}\mathbf{z}_t, \text{ where } \mathbf{z}_t = \begin{cases} (\mathbf{C}\Psi_{\text{tpl}})^{-1}\mathbf{y}_k, & \zeta = r_t \\ (\mathbf{C}\Psi_{\text{tpl}})^{\dagger}\mathbf{y}_k, & \zeta > r_t. \end{cases} \tag{17}$$

Therefore, the determined local schedule of each device k, $\Gamma^* = \{\Gamma_k^* \quad \forall k \in \mathcal{K}\}$, is calculated.

C. Design on the Run-Time Phase: S-Agents

As shown in Fig. 4, we deploy a unique DRL model named S-Agent on each EH device to control the run-time sensing operation. This article adopts deep-Q-network (DQN) [25], [26] as underlying methods. We define $s_{k,t}$ as the environment states of device k at t time slot. During the learning period, based on the state $s_{k,t}$, the neural networks of the S-Agent compute a control action $a_{k,t}$. Following $a_{k,t}$, the device k executes the corresponding action that leads to the environment transit to the next state $s_{k,t+1}$. Meanwhile, a feedback reward $r_{k,t+1}$ is returned to the agent for the action evaluation. The agent aims to select action $a_{k,t} \quad \forall t \in \mathcal{T}$ to maximize the expected long-term reward named q-value $Q(s,a)^\pi = \mathbb{E}[R_t^\pi | s_t = s, a_t = a]$, where π identifies the policy that maps the state to the action and $R_{k,t} = \sum_{t'=t}^{\infty} \gamma'^{t'-t} r_{t'+1}$, $\gamma' \in (0, 1]$. γ' is the factor to discount future reward.

Learning: During the learning period, the S-Agent k has two neural networks, including the target network and local network parameterized by θ_k and θ'_k. At each time slot, based on the state $s_{k,t}$, the local network θ_k will output $a_{k,t}$ to control the sensing operation. The target network will output the corresponding "label" data simultaneously. The S-Agent collects its own experiences $\text{experience} = (s_{i,t}, a_{i,t}, r_{i,t+1}, s_{i,t+1})$ and uploads the *experience* set to the relay buffer on the edge server at off-peak time. The minimum batch size *experiences* is randomly sampled from the replay buffer to optimize the local network and target network (line 31 in Algorithm 2). To stabilize the optimization, the target network will be forbidden from updating weights for m interactions. After the m interactions, the target network updates the weight by copying weights from the local network (lines 34–36 in Algorithm 2). The loss function of the local network is given by (18). To avoid local minimum and learn from the environment, DRL agents are expected to take more actions on exploration at an early age. After getting familiar with the environment, agents should exploit based on their experience. Thus, this article adopts the decaying epsilon-greedy exploration policy [25] (line 13–23 in Algorithm 2). The algorithm details are described in Algorithm 2. Specifically, all *S-Agents* have the same architecture with different weights

$$L(\theta_k) = \sum_{\substack{s_{k,t}, a_{k,t}, \\ r_{k,t+1}, s_{k,t+1} \in G_{i,t}}} \Big(r_{i,t+1} + \gamma \max_a Q\big(s_{k,t+1}, a_{k,t+1}; \theta'\big) - Q\big(s_{k,t}, a_{k,t}; \theta\big) \Big)^2. \tag{18}$$

[2]†denotes the Moore–Penrose pseudoinverse.

Algorithm 2: Learning Procedure for *S-Agents*

Input: $max\_episodes, m, n\_node, > \{\gamma', \epsilon_{start}, \epsilon_{end}$, decay_steps, $\delta\}$ (for learning and exploration)
$\{E_{k,sense}\kappa_k, p_k^{\text{trans}}, \omega_{k,trans}c_k, f_k, \forall k \in \mathcal{K}\}$
Output: optimized *S-Agents* $\theta_k, \forall k \in \mathcal{K}$

Initialize n_node local neural networks, $\theta_k, \forall k$;
Initialize n_node target neural networks, $\theta_k' \leftarrow \theta_k, \forall k$;
Initialize the corresponding replay buffer, $G_k, \forall k$;
$n\_episode \leftarrow 0$;
Epsilons = 0.01 / np.logspace(-2, 0, decay_steps,False)-0.01;
Epsilons = Epsilons * (ϵ_{start} - ϵ_{end}) + ϵ_{start};
while $n\_episode < max\_episodes$ **do**
  Reset EH IoT system environment;
  $Step_k \leftarrow 0 \quad \forall k$;
  for $t \leftarrow 1$ ***to*** T **do**
    for $k \leftarrow 1$ ***to*** n_node **do**
      // Update ϵ
      if $t >= decay\_steps$ **then**
        $\epsilon = \epsilon_{end}$;
      else
        ϵ = Epsilons[t];
      end
      // Epsilon-greedy exploration policy
      if *np.random.rand()* $> \epsilon$ **then**
        $a_{k,t} = \underset{\forall a_{k,t} \in \{0,1\}}{\text{argmax}} \; Q(s_{k,t}, a_{k,t})$;
      else
        $a_{k,t}$ = np.random.randint(len($Q(s_{k,t})$));
      end
      if $a_{k,t} == 1$ *and* $E_{k,rest} > E_k^{sense} b^{sense}$ **then**
        Perform sensing; /* $a_{k,t} \in \{0, 1\}$; $a_{k,t} == 1$ `Sensing; Otherwise, Sleeping` */
        Calculate $E_{k,rest}$ based on (6)
      end
      Environment transit to $s_{k,t+1}$; Feedback $r_{k,t+1}$
      $experience_k = (s_{k,t}, a_{k,t}, r_{k,t+1}, s_{k,t+1})$;
      Save experience to buffer G_k;
      if $|G_k| \geq mini\_batchsize$ **then**
        Randomly sample mini-batch of experiences;
        Optimize θ_k with sampled experiences;
        if $Step_k \% m == 0$ **then**
          Update target network θ_k' from θ_k;
        end
      end
      $Step_k = Step_k + 1$
    end
  end
  $n\_episode = n\_episode + 1$
end

MDP Settings: In a view of the architecture indicated in Fig. 4 and the discussions above, we define the distributed *S-Agents* POMDP as follows.

1) $s_{k,t} \in \mathbf{S}$: The state of environment is captured by the tuple of $\{E_{k,\text{rest}}, t, \mathcal{Q}\}$, where $E_{k,\text{rest}}$ is the current onboard energy to indicate if the EH device has the residual energy to perform sensing. The real time t gives the information to agent so as to select if sensing to capture the temporal features. $\mathcal{Q} \in \{0, 1\}$ is the suggestion from QR-ST.
2) $a_{k,t} \in \mathcal{A}$: The action $a_{k,t} \in \{0, 1\}$ that indicates if the EH device performs sensing operation or not at t time slot. Note that in Section VI we describe $a_{k,t}$ as $a_k[t]$.
3) $P_{ss'}(a_{k,t}) \in \mathcal{P}$: $P_{ss'}(a_t)$ is the state transition probability from state s into state s' while taking action $a_{k,t}$.
4) $r_{k,t} \in \mathcal{R}$: The reward function is defined as (19), which reflects the global reconstruction errors before t. $D_{\mathbf{X},\mathbf{L}}$ is calculated by (9), where $\mathbf{X}[t] = [\mathbf{x}[1], \mathbf{x}[2], \ldots, \mathbf{x}[t]]$.

TABLE I
PARAMETERS ON EH DEVICE AND DISTANCE MEASUREMENT

Notation	Definition	Value/Range
E_k^{sense}	Unit sensing energy	1e-5 J/bits
p_k^{trans}	Power of transmission	0.05w
f_k	CPU frequency	16MHz
h_{max}	maximum window size	24
w_{max}	maximum window size	24
δ_w	weighted factor	$e^{-0.8(w-1)}$
α_h	weighted factor	$e^{-0.8(w-1)}$
$E_{k,max}$	Energy capacity	1J

Therefore, $\mathbf{X}[t]$ has a different size on snapshots at different time slots. To ensure fairness, (19) is the reconstruction errors divided by the number of snapshots. Recall that the snapshot $\mathbf{x}[t]$ is reconstructed by (13) and $\mathbf{L}[t]$ is the real full-state spatiotemporal data

$$r_{k,t+1} = \min\left(0, \log\left(\frac{t}{\sum_{\tau=0}^{t} D_{\mathbf{X}[t],\mathbf{L}[t]}}\right)\right). \quad (19)$$

VIII. Evaluation

We developed a solar energy-powered IoT data collection simulator to evaluate the *S-Agents*. The device power traces are downloaded from [27].

A. Experiment Settings

EH IoT Sensing: We simulate 343-device EH IoT sensing system by following the similar model architecture in Fig. 3, where the devices begin operating from 8:00 A.M. to 18:00 P.M. to sense the luminosity data. The devices are expected to transmit the collected data to the edge server per 5 min. For the sensed data of devices, we adopt the real luminosity data sensed by 343 devices in the United States in 2006 downloaded from the database [28]. Overall, we have spatiotemporal data about luminosity for 365 days. The spatiotemporal data for each day consist of either 343 data traces or 156 snapshots. We randomly select the luminosity traces on 150 days from 343 devices trained Ψ_{spl} for 156 time slots and Ψ_{tpl} for 343 devices. After that, the luminosity traces of 115 days are for the S-Agent training. The leftover 100 days of data are used to evaluate the reconstruction errors. Table I lists the parameters used in our EH IoT device simulation [29] and multiresolution distance measurement [21].

S-Agent: We train S-Agent 115 episodes, which matches our EH IoT system simulation for 115 days. For POMDP, we set discount factor $\gamma = 0.9$ and the learning rate $\delta = 5e^{-4}$. The architecture of the target neural network model and local neural network model are both 3-64-256-2. At the beginning of each day, the environment will be reset, and all measured parameters will be reinitialized. This is for a fair comparison which prevents the system from being influenced by residual effects such as remaining energy from the previous day.

Baselines: We compare our S-Agent algorithm with the following four benchmarks to show the influence of each phase in our framework.

1) QR-ST baseline provides the determined sensing schedule with QR pivot factorization based method; compared

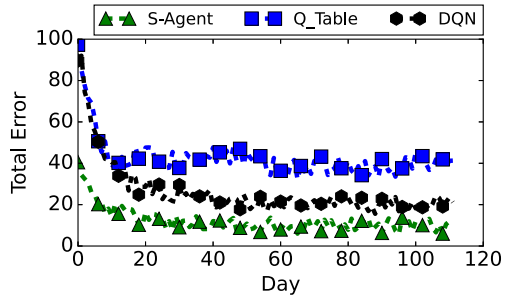

Fig. 5. S-Agent reconstruction error versus with training day.

Fig. 6. Average reconstruction errors of spatiotemporal data for 100 test days.

with the proposed S-Agents, the QR-ST algorithm provides a fixed schedule without consideration for dynamic fine-tuning of the sensing operation with DRL for the environmental variation.

2) *DQN* baseline adopts the DQN algorithm to schedule sensing; when compared with the proposed *S-Agents*, the preliminary suggestion from QR-ST is not provided in the DQN baseline.
3) *CS* baseline provides the sensing scheduling with *S-Agents* that reconstructs data with a universal basis, Fourier basis, where the *S-Agents* reconstruct data with the tailored basis Ψ_{spl} and Ψ_{tpl}.
4) *Q-table* baseline provides the sensing scheduling with the *Q*-table deployed on each device; This baseline aims to explore if we can save more computing resources by replacing the neural network with *Q*-table without sacrificing the performance of sparsity-aware sensing.

B. Overall Performance

We first evaluate the overall performance with measured average reconstruction error during the training period and test sets. Fig. 5 indicates the reconstruction error versus 115 training days. In the training period, the S-Agent does not only converges faster than the typical DQN but also achieves a much lower reconstruction error. The only difference between the S-Agent and the typical DQN is the S-Agent with the suggestions from QR-ST and the DQN without that. Although the distributed *Q*-table has the fastest convergent speed, its reconstruction error is almost 3× that of the *S-Agents*.

Fig. 6 overviews the average spatiotemporal data reconstruction error for 100 test days with the multiresolution distance-based measurement and Euclidean distance measurement. No matter which measurement we adopt, the S-Agent achieves

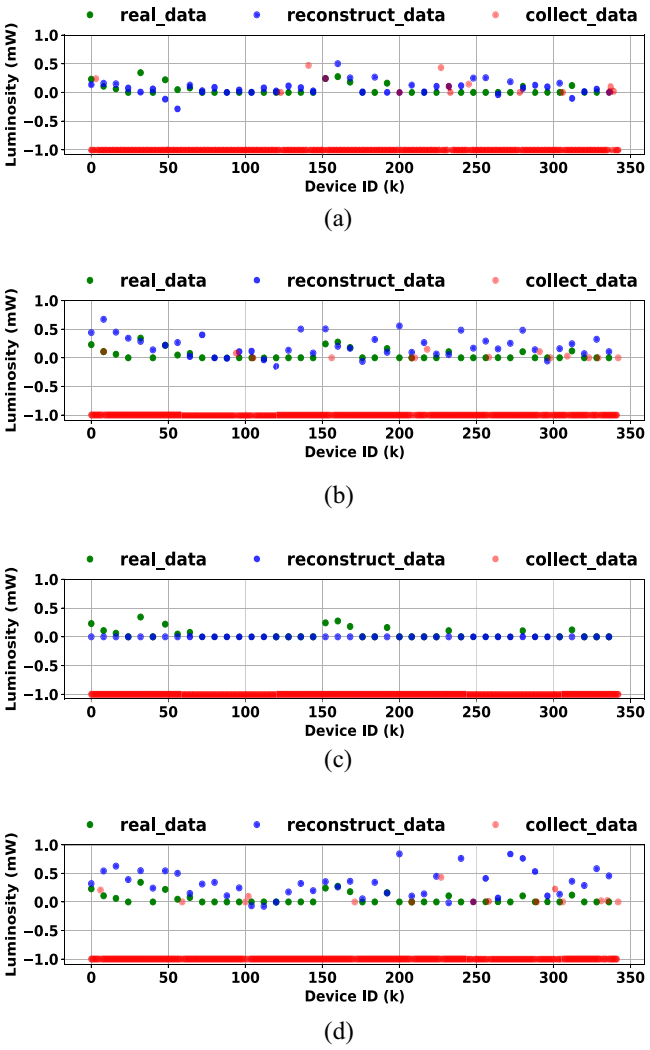

Fig. 7. Reconstructed snapshot at 10:00 by (a) S-Agent, (b) DQN, (c) QR-ST, and (d) CS.

the best performance on the reconstruction error. With the multiresolution distance-based measurement, the performance for different algorithms is distinguished, where the S-Agent decreases the reconstruction error to 6.78, reduced by 66.30% compared with DQN which has the best performance among the baselines. Although there is a much smaller difference among algorithms under Euclidean distance measurement, the S-Agent still outperforms all baselines.

C. Reconstruction Visualization

We randomly select the snapshots at 10:00 and 16:00 on one day to visualize the reconstruction details of three baselines. Since the reconstruction error completed by *Q*-table is obviously larger than other methods, we do not show it. Figs. 7 and 8 visualize the real data, the reconstructed data, and the collected sparse data of snapshots at 10:00 and 16:00, respectively. The green indicates the real data. The blue is reconstructed by the red collected data that is sparse. Ideally, the reconstructed data are expected to exactly coincide with the real data. Meanwhile, the fewer data required to be collected, the better is. Specifically, the luminosity value of the collected

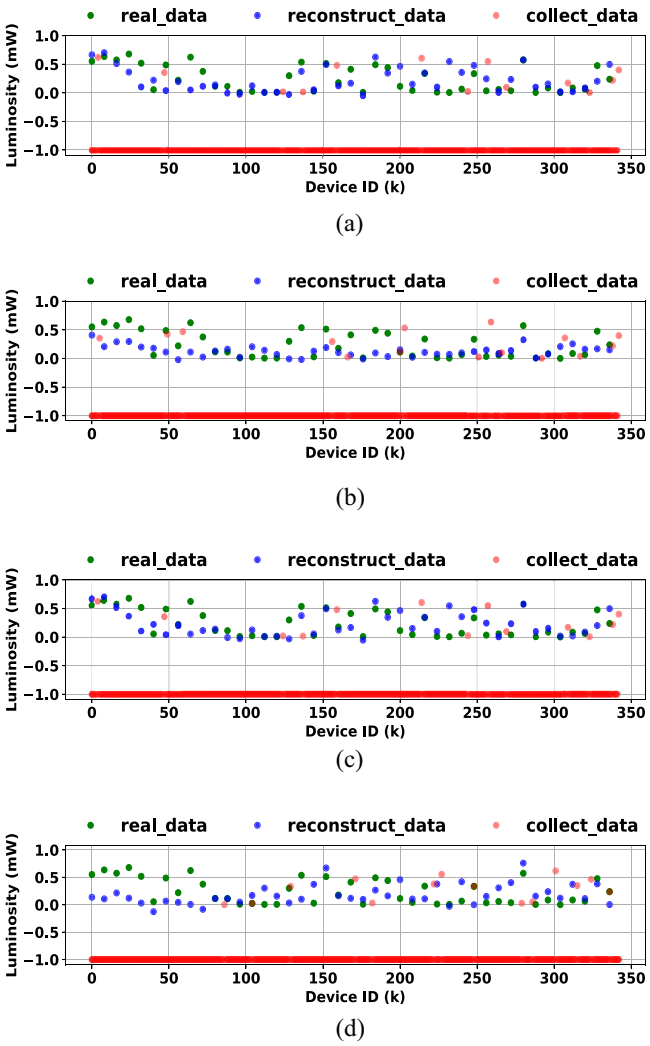

Fig. 8. Reconstructed snapshot at 16:00 by (a) S-Agent, (b) DQN, (c) QR-ST, and (d) CS.

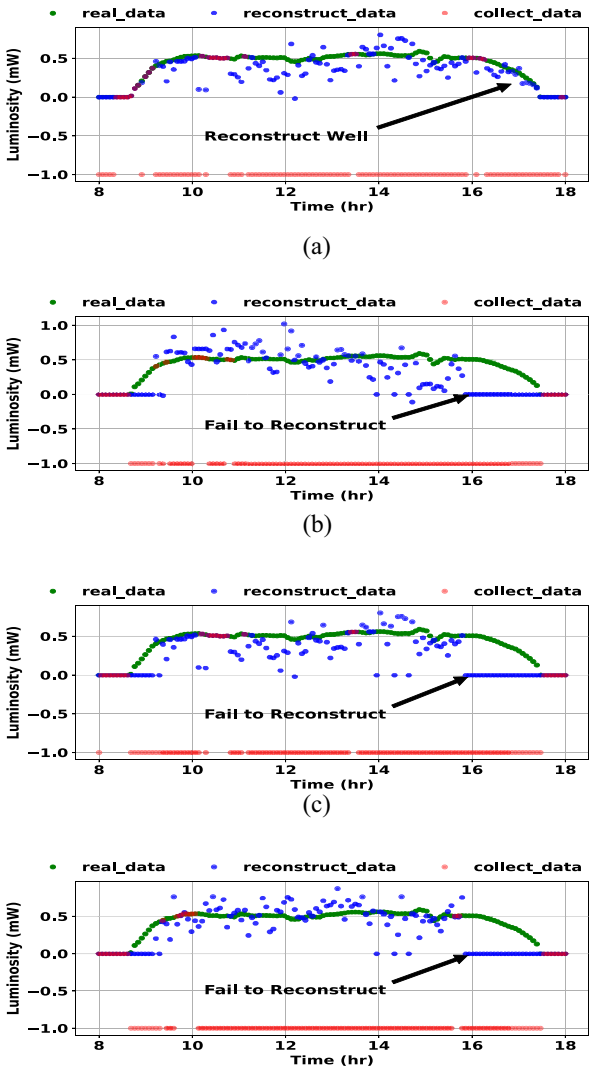

Fig. 9. Reconstructed traces at [25.75°N, 80.85°W] by (a) S-Agent, (b) DQN, (c) QR-ST, and (d) CS.

data at -1 indicates that the sensing at that device or time point is skipped. For readability, we only indicate a portion of reconstructed data and real data. The S-Agent can precisely capture the spatial features both at 10:00 and 16:00. DQN completes a better reconstruction at 10:00 compared with its reconstruction at 16:00. It is worth mentioning that at 16:00 QR-ST almost achieves the same reconstruction as that of S-Agent with the peak reconstruction performance among baselines. However, it does not capture any features at 10:00. No device collects data at 10:00 AM for QR-ST, which is the situation analyzed in Fig. 2(a). Therefore, due to the unstable environment of the EH IoT system, a fixed sensing strategy, such as QR-ST, cannot be adaptive with environmental variation. Due to the randomness of compressed sensing, the reconstruction in Figs. 7 and 8 is severely fluctuated.

Figs. 9 and 10 visualize the reconstruction from temporal dimension. It indicates the real luminosity data, reconstructed luminosity data, and collected luminosity data of two devices, located at [25.75°N, 80.85°W] and [36.25°N, 102.95°W], respectively. In Fig. 9, from 8:00 to 12:00, the number of collected data of four algorithms is almost the same. The S-Agents is the only one to successfully reconstruct the data at around 9:00. Moreover, from 16:00 to 18:00, although S-Agents captures three real data that is more than the number of collected data by other baselines, the reconstruction of S-Agents is merged better with the real data. In Fig. 10, also only S-Agents completed the reconstruction from 16:00 to 18:00. Excepting the CS, the other three methods achieve an outstanding reconstruction from 8:00 to 14:00, especially for the S-Agents with a smaller number of data collection.

D. Energy Discussion

We further explore the device's energy consumption of Fig. 9. Fig. 11 indicates the runtime energy usage of Fig. 9 given the corresponding sensing strategy. Without compressed sensing, the EH IoT devices are frequently interrupted because of the weak power supply. Due to the sparse sensing, the interruption is rarely caused by sensing. Furthermore, the energy of the S-Agent is sharply down from 8:00 to 9:00 to sense data, which reflects the S-Agent dynamically fine-tune the sensing strategy so as to collect the most critical data

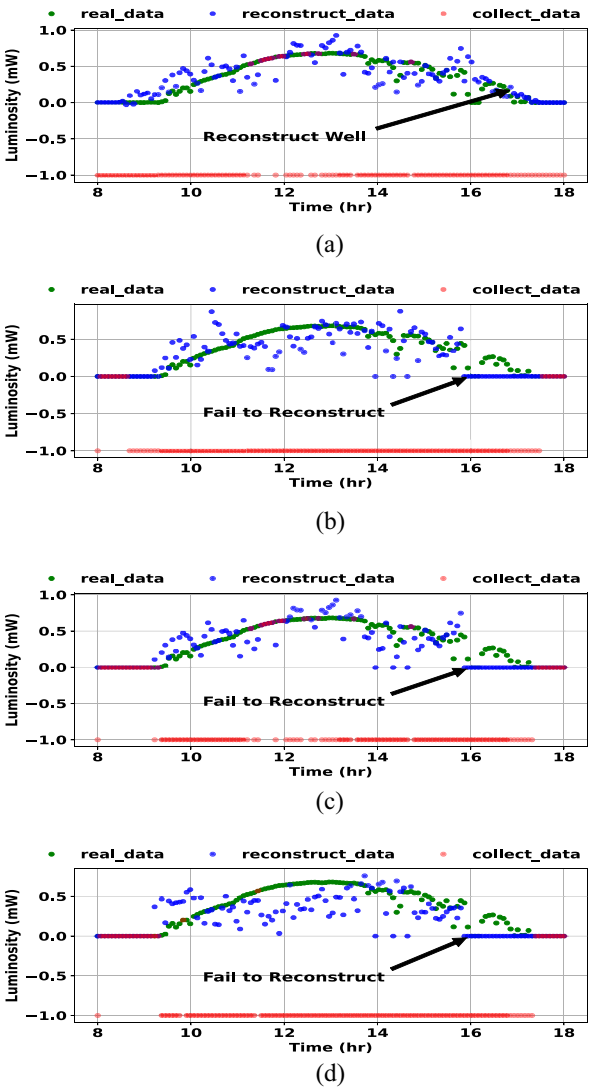

Fig. 10. Reconstructed traces at [36.25°N, 102.95°W] by (a) S-Agent, (b) DQN, (c) QR-ST, and (d) CS.

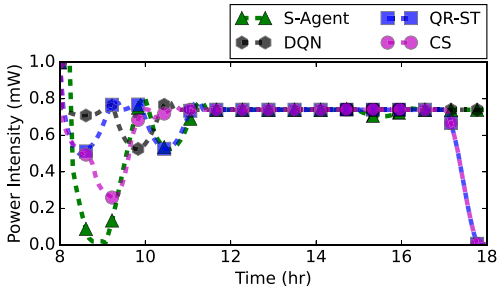

Fig. 11. Energy cost on sensing for the device of Fig. 9.

with limited energy. It actually implies the energy allocation capability of the S-Agent.

The raw data in Table II indicates the data size of luminosity data in 2006 collected by 343 devices without applying the sparsity-aware sensing. Table II shows the total data size collected in our simulation for reconstructing while we applied

TABLE II
TOTAL COLLECTED DATA SIZE (KB)

Methods	Raw Data	*S-Agent*	DQN	*QR-ST*	CS	Q-table
Data Size	$1.5e3$	$1.34e2$	$1.40e2$	$1.31e2$	$2.61e2$	$3.24e2$

TABLE III
OVERHEAD ON MCU OF DIFFERENT ALGORITHMS

Methods	**FLOPs**	**Latency**	**Energy overhead**
S-Agents	33.855k	$6.8\mu s$	6.8e-6J
DQN	33.727k	$6.2\mu s$	6.2e-6J
QR-ST	-	-	-
CS	33.855k	$6.8\mu s$	6.8e-6J
Q-table	1.561k	$0.27\mu s$	2.7e-5J

the algorithms of S-Agent, DQN, QR-ST, CS, and Q-table for the sparsity-aware sensing. Fig. 6 and Table II indicate that the S-Agent only needs to collect 8.93% data that can reconstruct the full-state data with 7.86 errors. Thus, we can save the sensing and communication resources on the 91.07% redundant data while the full-state data is reconstructed with 7.86 errors (6.78 errors with multiresolution distance measurement).

E. Overhead Evaluation

We discuss the overhead of the different algorithms in Table III, which illustrates the number of floating-point operations (FLOPs), runtime, and energy overhead of the proposed *S-Agents* and four benchmarks, where the FLOPs are the number of FLOPs on running a 3-64-256-2 fully connected neural network, runtime is tested on TI MSP430FR6989 LaunchPad [29], and the energy overhead is the processing energy costs while the processing power is 0.001 J/s. The QR-ST does not have overhead because the QR-ST algorithm uses the fixed scheduling that is calculated by the servers. Therefore, at the runtime stage, the QR-ST algorithm only needs to retrieve the decision from the memory. The runtimes of the *S-Agents* and the *CS* are slightly higher than that of the *DQN*, which is caused by that DQN does not take any suggestion from the QR-ST so that its FLOPs are less than that of the *S-Agents* and the *CS*. Although the overhead of the *S-Agents* are the highest among the four benchmarks in terms of the FLOPs, runtime, and energy overhead on processing, its reconstruction effectiveness dramatically outperforms that of the four benchmarks.

IX. CONCLUSION

This article aims to construct a sparsity-aware spatiotemporal data sensing framework for the EH-powered IoT system by employing the multiagent DRL to optimize the sensor selection from spatial perspective and sensing scheduling from temporal perspective. We first construct a comprehensive spatiotemporal data sensing for the EH-powered IoT system. With the consideration of limited computation capability of EH IoT devices, we then develop QR-ST to compute the determined spatiotemporal sensing scheduling. Through leveraging the determined sensing scheduling, the DRL-based *S-Agents* is further developed to dynamically fine-tune sensing operation in the runtime.

REFERENCES

[1] C. Pan, S. Gu, M. Xie, Y. Liu, C. J. Xue, and J. Hu, "Wear-leveling aware page management for non-volatile main memory on embedded systems," *IEEE Trans. Multi-Scale Comput. Syst.*, vol. 2, no. 2, pp. 129–142, Apr.–Jun. 2016.

[2] C. Pan, M. Xie, and J. Hu, "ENZYME: An energy-efficient transient computing paradigm for ultralow self-powered IoT edge devices," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 37, no. 11, pp. 2440–2450, Nov. 2018.

[3] W. Zhang, J. Zhang, M. Xie, T. Liu, W. Wang, and C. Pan, "M2M-routing: Environmental adaptive multi-agent reinforcement learning based multi-hop routing policy for self-powered IoT systems," in *Proc. Design, Autom. Test Europe Conf. Exhibit. (DATE)*, 2022, pp. 316–321.

[4] W. Zhang, T. Liu, M. Xie, L. Li, D. Kar, and C. Pan, "Energy harvesting aware multi-hop routing policy in distributed IoT system based on multi-agent reinforcement learning," in *Proc. 27th Asia South Pacific Design Autom. Conf. (ASP-DAC)*, 2022, pp. 562–567.

[5] J. Xie *et al.*, "Distance measure to cluster spatiotemporal scenarios for strategic air traffic management," *J. Aerosp. Inf. Syst.*, vol. 12, no. 8, pp. 545–563, 2015.

[6] E. J. Candès and M. B. Wakin, "An introduction to compressive sampling," *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 21–30, Mar. 2008.

[7] K. Manohar, B. W. Brunton, J. N. Kutz, and S. L. Brunton, "Data-driven sparse sensor placement for reconstruction: Demonstrating the benefits of exploiting known patterns," *IEEE Control Syst. Mag.*, vol. 38, no. 3, pp. 63–86, Jun. 2018.

[8] S. Liu, M. Fardad, E. Masazade, and P. K. Varshney, "Optimal periodic sensor scheduling in networks of dynamical systems," *IEEE Trans. Signal Process.*, vol. 62, no. 12, pp. 3055–3068, Jun. 2014.

[9] H. Jamali-Rad, A. Simonetto, X. Ma, and G. Leus, "Distributed sparsity-aware sensor selection," *IEEE Trans. Signal Process.*, vol. 63, no. 22, pp. 5951–5964, Nov. 2015.

[10] S. Liu, S. Kar, M. Fardad, and P. K. Varshney, "Sparsity-aware sensor collaboration for linear coherent estimation," *IEEE Trans. Signal Process.*, vol. 63, no. 10, pp. 2582–2596, May 2015.

[11] M. Calvo-Fullana, J. Matamoros, and C. Antón-Haro, "Sensor selection and power allocation strategies for energy harvesting wireless sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3685–3695, Dec. 2016.

[12] Y. Wang, Z. Tian, and C. Feng, "Sparsity order estimation and its application in compressive spectrum sensing for cognitive radios," *IEEE Trans. Wireless Commun.*, vol. 11, no. 6, pp. 2116–2125, Jun. 2012.

[13] X. Wang, Z. Yang, J. Huang, and R. C. de Lamare, "Robust two-stage reduced-dimension sparsity-aware STAP for airborne radar with coprime arrays," *IEEE Trans. Signal Process.*, vol. 68, no. 10, pp. 81–96, Dec. 2019.

[14] E. Masazade, M. Fardad, and P. K. Varshney, "Sparsity-promoting extended Kalman filtering for target tracking in wireless sensor networks," *IEEE Signal Process. Lett.*, vol. 19, no. 12, pp. 845–848, Dec. 2012.

[15] E. Clark, J. N. Kutz, and S. L. Brunton, "Sensor selection with cost constraints for dynamically relevant bases," *IEEE Sensors J.*, vol. 20, no. 19, pp. 11674–11687, Oct. 2020.

[16] E. Clark, T. Askham, S. L. Brunton, and J. N. Kutz, "Greedy sensor placement with cost constraints," *IEEE Sensors J.*, vol. 19, no. 7, pp. 2642–2656, Apr. 2019.

[17] M. Calvo-Fullana, J. Matamoros, and C. Antón-Haro, "Sensor selection in energy harvesting wireless sensor networks," in *Proc. IEEE Global Conf. Signal Inf. Process. (GlobalSIP)*, 2015, pp. 43–47.

[18] D. L. Donoho, "For most large underdetermined systems of linear equations the minimal $\ell 1$-norm solution is also the sparsest solution," *Commun. Pure Appl. Math.*, vol. 59, no. 6, pp. 797–829, 2006.

[19] A. Wyner, "Recent results in the Shannon theory," *IEEE Trans. Inf. Theory*, vol. 20, no. 1, pp. 2–10, Jan. 1974.

[20] W. Zhang, L. Li, N. Zhang, T. Han, and S. Wang, "Air-ground integrated mobile edge networks: A survey," *IEEE Access*, vol. 8, pp. 125998–126018, 2020.

[21] W. Zhang, J. Xie, and Y. Wan, "Spatiotemporal scenario data-driven decision-making framework for strategic air traffic flow management," in *Proc. IEEE 15th Int. Conf. Control Autom. (ICCA)*, 2019, pp. 1108–1113.

[22] J. Xie and Y. Wan, "Scalable multidimensional uncertainty evaluation approach to strategic air traffic flow management," in *Proc. AIAA Model. Simulat. Technol. Conf.*, 2015, p. 2492.

[23] P. Businger and G. H. Golub, "Linear least squares solutions by householder transformations," *Numerische Mathematik*, vol. 7, no. 3, pp. 269–276, 1965.

[24] G. H. Golub and C. Reinsch, "Singular value decomposition and least squares solutions," in *Linear Algebra*. Heidelberg, Germany: Springer, 1971, pp. 134–151.

[25] M. Morales, *Grokking Deep Reinforcement Learning*. Shelter Island, NY, USA: Manning Publ., 2020.

[26] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[27] "Measurement and Instrumentation Data Center (MIDC)." Accessed: Mar. 6, 2022. [Online]. Available: https://midcdmz.nrel.gov/apps/sitehome.pl?site=ORNL

[28] "Light Intensity Data." Accessed: Mar. 6, 2022. [Online]. Available: https://www.nrel.gov/grid/solar-power-data.html

[29] "The User Guide of MSP430FR6989 LaunchPad." Accessed: Mar. 6, 2022. [Online]. Available: https://www.ti.com/lit/ug/slau627a/slau627a.pdf?ts=1654849977248

Wen Zhang received the B.E. degree from Chang'an University, Xi'an, China, in 2017. She is currently pursuing the Ph.D. degree with Texas A&M University at Corpus Christi, Corpus Christi, TX, USA.

Her research interests are in self-sustaining IoT system, deep reinforcement learning for energy-efficient IoT system, and intelligent sparsity-aware sensing for low-power IoT devices.

Mimi Xie (Member, IEEE) received the B.E. and M.S. degrees from the College of Computer Science, Chongqing University, Chongqing, China, in 2010 and 2013, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Pittsburgh, Pittsburgh, PA, USA, in 2019.

She is currently an Assistant Professor with the Department of Computer Science, University of Texas at San Antonio, San Antonio, TX, USA. Her current research interests include energy harvesting embedded systems and AI on edge.

Caleb Scott received the B.S. degree in computer science from the University of Texas at San Antonio, San Antonio, TX, USA, in 2021, where he is currently pursuing the Ph.D. degree.

His current research interests are signal processing, compressive sensing, and energy harvesting embedded systems.

Chen Pan (Member, IEEE) received the M.S. degree in electrical engineering from Oklahoma State University, Stillwater, OK, USA, in 2017, and the Ph.D. degree in electrical and computer engineering from the University of Pittsburgh, Pittsburgh, PA, USA, in 2019.

He is currently an Assistant Professor with the Department of Computing Sciences, Texas A&M University at Corpus Christi, Corpus Christi, TX, USA. His current research interests include sustainable and intelligent IoT systems, edge-driven AI, low-power embedded systems, and emerging nonvolatile memories.