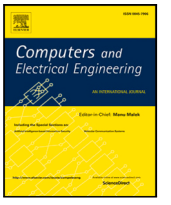
Computers and Electrical Engineering 123 (2025) 110424

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng



Cell selection in mobile crowdsensing using multi-objective deep reinforcement learning

Shabnam Seradji [a], Ahmad Khonsari [a,b,*], Mahdi Dolati [b], Vahid Shah-Mansouri [a]

[a] School of Electrical and Computer Engineering, University of Tehran, Tehran, 1439957131, Iran
[b] School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Tehran, 19538-33511, Iran

ARTICLE INFO

Keywords:
Mobile crowdsensing
Cell selection
Multi-objective optimization
Deep reinforcement learning
Data correlation
Budget constraints

ABSTRACT

Mobile Crowdsensing (MCS) has emerged as a leading paradigm that harnesses user participation to collect data through mobile devices. To minimize operational costs, MCS platforms typically divide the environment into smaller subareas (cells) and employ inference algorithms to estimate data across the entire region based on measurements obtained from a selected subset of cells. While previous studies have investigated cell selection strategies in MCS, limited research has focused on multi-objective approaches that simultaneously prioritize specific cells for direct sensing while minimizing overall inference errors. This dual objective presents a significant challenge, as the most prioritized cells may not always correspond to the most informative ones. This paper introduces MODRQN-CS, a novel multi-objective cell selection strategy that integrates data correlations and budget constraints to enhance efficiency. The proposed approach leverages Multi-Objective Deep Reinforcement Learning (MODRL), utilizing the Deep Recurrent Q-Network (DRQN) to optimize cell selection in terms of both prioritization and informativeness. Experimental evaluations on a real-world dataset demonstrate that MODRQN-CS outperforms baseline methods, achieving enhanced performance in balancing data accuracy and the coverage of prioritized cells.

1. Introduction

Leveraging advancements in wireless communication and the proliferation of smart devices, along with the rise of Internet of Things (IoT) applications [1], Mobile Crowdsensing (MCS) has emerged as an effective paradigm for large-scale sensing across domains such as environmental monitoring [2], transportation [3], traffic monitoring [4], and text data collection [5]. By utilizing widely available smart devices, MCS ensures extensive coverage and cost-efficient data collection. Data acquisition in MCS occurs through *participatory* sensing, requiring active user involvement, or *opportunistic* sensing, which passively collects data in the background.

Effective resource management in MCS relies on efficient task assignment strategies. Crowdsensing Platforms (CSPs) recruit users from specific areas, often subdivided into smaller regions or cells, to perform sensing tasks. Some approaches classify users as trusted, suspicious, or malicious [6], while encryption-based task allocation mechanisms have been proposed to enhance privacy [7]. However, this study assumes that users are trusted and capable of accurately completing assigned tasks. Under this assumption, the

* Corresponding author at: School of Electrical and Computer Engineering, University of Tehran, Tehran, 1439957131, Iran.
E-mail addresses: seradji.shabnam@ut.ac.ir (S. Seradji), a_khonsari@ut.ac.ir (A. Khonsari), m.dolati@ipm.ir (M. Dolati), vmansouri@ut.ac.ir (V. Shah-Mansouri).
URLs: https://hpnl.ir/people/ahmad-khonsari.html (A. Khonsari), https://ece.ut.ac.ir/en/ (V. Shah-Mansouri).

https://doi.org/10.1016/j.compeleceng.2025.110424
Received 2 January 2025; Received in revised form 8 April 2025; Accepted 5 May 2025
Available online 25 May 2025

user recruitment problem simplifies to the cell selection problem (CS), which is a key challenge in MCS that involves selecting cells for sensing to ensure high-quality data collection while adhering to budget constraints [8–10].

Traditional MCS methods often rely on recruiting large user groups to ensure data quality, however, budget constraints make this approach impractical. Sparse MCS addresses this challenge by using data inference algorithms, such as Compressive Sensing [11] and Kriging [12], to estimate unsensed cell values based on sensed data and spatio-temporal correlations in applications like environmental monitoring [10]. This reduces costs while maintaining acceptable inference accuracy, measured by the difference between inferred and ground truth data [9]. Thus, optimizing CS to minimize inference error remains a critical research challenge.

MCS performance also depends on the spatial and temporal coverage of specific cells, which vary based on application needs. Spatial coverage maximizes the number of covered subareas, while temporal coverage ensures sufficient data collection over time [13]. Certain cells, referred to as popular cells, are prioritized due to their significance to the CSP. Prioritizing specific cells within a target area is essential for optimizing data collection, primarily based on application requirements. In environmental monitoring, cell prioritization is influenced by factors such as population density and human exposure (e.g., residential or industrial areas) [14], critical environmental zones [15], economic incentives [16], and the demand for real-time data [3]. Popularity can be quantified using metrics like location entropy [17].

Previous studies have explored inference accuracy and coverage objectives separately [8–10,13,18–20], but optimizing both simultaneously remains an open challenge, highlighting the need for multi-objective optimization in MCS. To address this, we adopt a Reinforcement Learning (RL) framework, which offers greater adaptability than traditional optimization methods and requires minimal prior knowledge of the environment [8,9,18]. However, due to the large state space, conventional RL techniques (e.g., Q-tables) become inefficient, necessitating Deep Reinforcement Learning (DRL) to mitigate the curse of dimensionality. DRL also enables the integration of diverse models, such as feedforward neural networks and long short-term memory (LSTM) networks, to capture spatio-temporal correlations. Accordingly, we employ a Deep Recurrent Q-Network (DRQN) with an LSTM layer to model these correlations, enhancing performance over standard DRL methods like Deep Q-Networks (DQN).

Multi-Objective Reinforcement Learning (MORL) effectively balances competing objectives but has seen limited application in the CS problem. To address the aforementioned challenges, we propose MODRQN-CS, a multi-objective CS strategy that optimizes both inference accuracy and prioritized cell coverage under budget constraints. By formulating the problem as a finite Multi-Objective Markov Decision Process (MOMDP), we develop a multi-objective deep reinforcement learning solution. MODRQN-CS leverages the DRQN architecture to model state dependencies, achieving improved performance. We implement scalarized multi-objective deep reinforcement learning by integrating concepts from existing frameworks [21,22] to combine multiple DRQNs for solving the multi-objective cell selection problem. To the best of our knowledge, this study is the first to apply Multi-Objective Deep Reinforcement Learning (MODRL) to a cell selection strategy that simultaneously maximizes prioritized cell coverage and inference accuracy.

To focus on the cell selection problem and align with prior studies [8,9,18], we assume a sufficiently large and consistently distributed participant pool, with users remaining in their designated cells until transmitting sensing data. Additionally, to preserve the Markovian assumptions of the MOMDP, we assume a steady-state environment that evolves at a manageable rate, allowing for timely neural network retraining. These assumptions are particularly relevant to environmental monitoring, which is the primary application of this study.

The main contributions of this paper are summarized as follows:

- We formally formulate the budget-constrained multi-objective CS problem and establish its NP-completeness.
- We propose a CS approach that simultaneously maximizes inference accuracy and the coverage of prioritized cells, extending beyond prior studies.
- We develop MODRQN-CS, an algorithm that integrates DRQNs for each objective and employs scalarization techniques to effectively combine multiple Q-functions.
- We validate the proposed approach using real-world Sensor-Scope dataset experiments, demonstrating its effectiveness in optimizing both inference accuracy and coverage rewards. The Sensor-Scope dataset is selected due to (1) its suitability in representing environments relevant to this study and (2) its extensive use in prior related studies for evaluation purposes.

The rest of this paper is structured as follows: Section 2 reviews related work on CS in MCS and multi-objective reinforcement learning. Section 3 defines the system model and formulates the multi-objective CS problem. Section 4 details the MODRQN-CS algorithm. Section 5 presents simulation results, followed by concluding remarks in Section 6.

2. Related works

This section reviews prior research on MODRL and CS strategies in MCS.

2.1. Multi-objective reinforcement learning

While many approaches focus on single-objective settings, real-world problems often involve multiple conflicting objectives. In domains like robotics, agents must balance competing goals, such as maximizing server performance while minimizing power consumption. This complexity has driven increasing interest in the MORL paradigm, which optimizes multiple criteria simultaneously [23].

Multi-objective problems can be addressed using single-policy or multi-policy approaches. Single-policy methods derive a solution based on predefined objective preferences, often employing scalarization functions to convert multiple objectives into a single one

Table 1
Comparison with recent works.

Related works	Properties			
	Sparse MCS	RL-based	Multi-objective	MODRL solution
[4,25,28]	✓	✘	✘	✘
[2,8–10,18,19,27]	✓	✓	✘	✘
[26,29]	✓	✘	✓	✘
[20]	✘	✓	✓	✘
Proposed model	✓	✓	✓	✓

using a weight matrix. For example, some methods apply linear prioritization to objectives to obtain an optimal policy [22]. In contrast, multi-policy MORL methods generate a set of policies that approximate optimal solutions for various user preferences [23]. Unlike single-policy approaches, they do not require predefined objective weights and can determine a Pareto-optimal policy at runtime.

Single-policy methods are computationally more efficient than multi-policy approaches but require prior knowledge of objective preferences. In contrast, multi-policy methods approximate the Pareto front, offering valuable trade-off insights between competing objectives. The primary drawback of multi-policy methods is their substantial computational cost, which can hinder the efficiency of MORL algorithms in online learning scenarios [23,24]. Given the lower computational demands of single-policy methods and the need for problem simplification, this study adopts a single-policy approach to better suit the online learning context. Future research may explore alternative single-policy or multi-policy methods to enhance performance and adaptability.

The authors in [24] highlight the broad application of MORL methods in real-world problems but note the limited research on deep learning for MORL. They propose a generic MODRL framework for finding Pareto-optimal solutions, supporting both single-policy and multi-policy strategies with linear and non-linear action selection. Similarly, the study in [22] introduces a modular approach integrating multiple DQNs in multi-objective environments. This architecture employs separate DQNs for different objectives, combines them based on user-defined priorities, and enhances scalarization using decision values. Additionally, it enables dynamic adaptation of the agent's behavior post-learning, maintaining consistent performance amid shifting objective priorities.

2.2. Cell selection

Considering the correlations among sensing data in various MCS applications, cell selection for inferring broader environmental areas remains a critical research challenge. The studies in [8,9,18] employ deep reinforcement learning for crowdsensing, highlighting its effectiveness in selecting informative cells. In [8,9], the authors seek to minimize the number of selected cells while maintaining data quality through leave-one-out Bayesian inference and DRQN. The study in [18] extends this by incorporating user recruitment through a three-step strategy: user selection, CS, and user-cell cross-selection.

The limitations of uniform cell division are addressed in [25], where the authors propose a learning-based task assignment framework for sparse MCS. Their approach divides target areas into non-uniform cells to enhance representativeness and integrates cell division learning, task assignment, sensing map reconstruction, and data quality evaluation. Similarly, the study in [26] introduces a dynamic region partitioning strategy that considers correlations between sensing regions. The proposed method addresses robust data inference in the presence of sparse noise through robust matrix completion techniques.

The time-dependent nature of urban environments is considered in [2], where the CS model is continuously updated to adapt to environmental changes. The framework uses updated training data to maintain the relevance of the CS model. In [19], an online sparse crowdsensing framework integrates online matrix completion, area importance estimation, and user recruitment, leveraging reinforcement learning for real-time evaluation of area importance. A hierarchical architecture for collaborative sensing is proposed in [27], which includes a budget allocation agent and multiple data collection agents, accounting for inter- and intra-data correlations.

Trustworthiness and cost-efficiency are addressed in [4], where the authors consider heterogeneous sensing costs and user reliability, using regularized mutual coherence for data inference based on compressed sensing theory. A dual online worker selection strategy that incorporates worker trajectories and budgets is presented in [10]. In [28], the proposed method focuses on recruiting users to sense significant cells, emphasizing data similarity over distance-based metrics for cell importance evaluation.

While most studies primarily focus on single-objective optimization, [29] addresses cost variations within cells as a bi-objective optimization problem, aiming to maximize informativeness and minimize costs. They propose Pareto Optimization Selection and Generalized Cost-Benefit Greedy Selection strategies. Similarly, [20] employs multi-agent DRL to optimize spatial–temporal coverage and costs simultaneously. In contrast, our work concurrently addresses the dual objectives of maximizing inference accuracy and ensuring the coverage of prioritized cells—an approach not yet explored in CS strategies. We utilize MODRL, which offers advantages over traditional methods in sparse MCS scenarios. This paper proposes an algorithm to achieve these objectives, as summarized in Table 1.

3. System model and problem formulation

In this section, we describe the system model for MCS cell selection and present the formulation of the optimization problem. Table 2 summarizes the main notations used throughout this paper.

Table 2
List of main notations.

Notation	Description
M	The number of cells in the target area.
K	The number of cycles.
B	The platform budget for selecting cells.
$\mathbf{F}$	The ground truth data matrix.
$\hat{\mathbf{F}}$	The inferred data matrix.
e_j	The inference error in cycle j.
cov_j	The coverage of popular cells in cycle j.
$\mathcal{P}$	The set of popular cells.

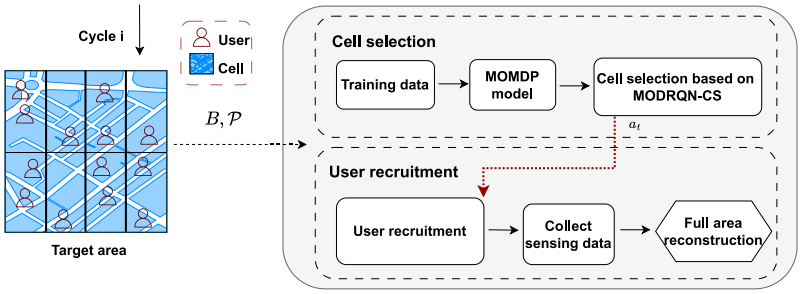


Fig. 1. Schematic representation of the system model.

3.1. System model description

Consider a geographic region divided into M cells, represented by the set $C = \{c_1, c_2, \ldots, c_M\}$, as shown in Fig. 1. The target area is partitioned into a fixed number of cells of uniform size, consistent with prior studies [8,9,18,29]. The CSP conducts sensing activities over K cycles of equal duration. At the start of each cycle, the CSP selects a subset of cells occupied by users to perform specific sensing tasks. This selection process is characterized by a *cell selection matrix*, denoted by $\mathbf{X}_{M\times K}$, where $\mathbf{X}(i,j) = 1$ indicates that cell i is chosen for data collection during cycle j, and $\mathbf{X}(i,j) = 0$ otherwise [8,9,18].

By leveraging the inherent correlations among data from different cells, the CSP aims to maximize inference accuracy within the given budget B. We assume that, for each selected cell in each cycle, at least one user is available for recruitment by the CSP. This user can perform the sensing task accurately and remain in the cell until the task is completed, consistent with prior works on CS in MCS [8,9,18].

The ground truth data matrix for a given sensing task is denoted as $\mathbf{F}_{M\times K}$, where $\mathbf{F}(i,j)$ represents the true value in cell i during cycle j. Selected users perform sensing tasks and transmit the results to the CSP. Based on the received data, the CSP uses a data inference technique to estimate the data for the unsensed cells, represented by the matrix $\hat{\mathbf{F}}_{M\times K}$. We employ the Ordinary Kriging algorithm for data inference, driven by its demonstrated effectiveness over compressed sensing algorithms in [2]. Ordinary Kriging is a linear unbiased estimator that leverages nearby data points to interpolate unknown values at specific locations, relying on a predefined or estimated covariance structure. It can employ either semivariograms or covariance functions. Inference accuracy is assessed using an inference error, which measures the deviation between the ground truth and the inferred data [8]. The Mean Absolute Error (MAE) metric is used to measure inference error. The average inference error across M cells during cycle j is defined as:

$$e_j = \frac{1}{M}\sum_{i=1}^{M}\left|\hat{\mathbf{F}}(i,j) - \mathbf{F}(i,j)\right|. \tag{1}$$

Another critical objective for the CSP is to optimize the coverage of popular cells. While the identification of popular cells depends on the MCS application and is beyond the scope of this paper, location diversity serves as a potential metric for defining popular cells. Specifically, cells that are less accessible and infrequently visited by users are considered more popular and should be prioritized for selection [17]. We assume that the CSP can determine the set of popular cells within the target area, denoted as $\mathcal{P}$, prior to initiating the crowdsensing process. To evaluate the coverage of popular cells, we adopt temporal coverage as a metric [13]. The coverage of popular cells during cycle j is defined as:

$$cov_j = \min_{i\in\mathcal{P}}\left\{\sum_{t=j-4}^{j}\mathbf{X}(i,t)\right\}. \tag{2}$$

Budget constraints limit the ability to guarantee inference accuracy. Thus, our strategy adopts direct visits to popular cells over inferring their data from neighboring cells to enhance confidence in sensing values. Direct visits are crucial when precise data is needed for high-stakes decisions, with users receiving higher rewards for providing critical data. This work aims to (i) maximize inference accuracy and (ii) maximize the temporal coverage of popular cells, framing the problem as a multi-objective optimization that balances these competing goals.

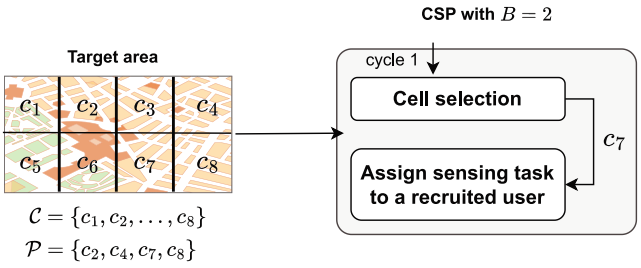


Fig. 2. Illustration of the cell selection approach, where the target area is divided into 8 cells, and the optimal cells are selected based on multi-objective criteria.

3.2. Problem formulation

The sensing process operates over K sensing cycles, where the CSP tackles a multi-objective optimization problem. It selects B cells per cycle to maximize inference accuracy and the temporal coverage of popular cells across K cycles and M target area cells. The problem is formulated as follows:

$$\max \quad \sum_{j=1}^{K} w_1 \left(1 - e_j\right) + w_2 \cdot cov_j \tag{3a}$$

$$\text{s.t.} \quad \sum_{i=1}^{M} \mathbf{X}(i, j) \leq B, \qquad \forall j \in \{1, \ldots, K\}. \tag{3b}$$

Constraint (3b) ensures that the number of selected cells per cycle does not exceed the budget B. For simplicity, $1 - e_j$ represents inference accuracy without loss of generality. The objective function in (3a) maximizes both inference accuracy and the coverage of popular cells across all sensing cycles, where w_1 and w_2 denote the CSP's relative preferences for these objectives.

Theorem 1. *Problem (3) is NP-hard.*

Proof. To establish the NP-hardness of Problem (3), we consider a simplified case where only the objective of maximizing inference accuracy $(1 - e_j)$ is addressed, disregarding the coverage objective. In this scenario, the CS problem reduces to a subset selection problem, wherein a subset of B elements is chosen to optimize a criterion function. It is well-established that the subset selection problem is NP-hard. As this simplified case is a special instance of Problem (3), it follows that Problem (3) is at least as hard as the subset selection problem. Hence, we conclude that Problem (3) is NP-hard. □

Fig. 2 illustrates a target area divided into eight cells, $\{c_1, \ldots, c_8\}$. At the beginning of the first sensing cycle, the CSP selects $B = 2$ cells for sensing. Assuming equal weights for inference accuracy and coverage, $w_{\text{inf}} = w_{\text{pop}} = 1$, the corresponding q-vectors are $\mathbf{q}_1 = [0.6, 0.36, 0.63, 0.41, 1, 0.68, 0.9, 0.18]$ for inference accuracy and $\mathbf{q}_2 = [0.18, 0.68, 0.3, 1, 0.47, 0.48, 0.79, 0.8]$ for coverage. While cell c_5 maximizes inference accuracy and cell c_4 maximizes coverage, the optimal choice considering both objectives differs. By summing the q-vectors element-wise, the cell with the highest resulting q-value is selected. Here, cell c_7 has the maximum combined q-value, making it the optimal choice.

4. Methodology

This section presents our multi-objective CS method, drawing on concepts from the MODRL framework and scalarization techniques from [21,22]. Unlike [22], which employs decision values for advanced scalarization of multiple Q-functions, our approach omits decision values and adapts the framework to meet our specific problem requirements.

Let $\mathcal{J}$ denote the set of objectives pursued by the agent, where $|\mathcal{J}| = N$ represents the total number of objectives. Each objective $j \in \mathcal{J}$ is assigned a priority $f(j)$, such that j_i is assigned a lower priority than j_k if $f(j_i) < f(j_k)$. For simplicity, we assume $f(j) \geq 0 \, \forall j \in \mathcal{J}$, enabling priorities to be interpreted as weights. In MORL, the agent receives multiple rewards corresponding to different objectives. At time step t, the agent receives a reward vector $\mathbf{r}_t = [r_{1,t}, \ldots, r_{N,t}]$, where $r_{i,t}$ corresponds to the reward for objective j_i.

Each objective j_i is characterized by a Q-function $Q_i(s, a)$, which quantifies the expected discounted return for taking action a in state s. The set of Q-functions is represented as $\mathbf{Q}(s, a) = [Q_1(s, a), \ldots, Q_N(s, a)]$, where $Q_i(s, a)$ denotes the Q-function for objective j_i. The optimal action for objective j_i at time step t, given the state s_t, is defined as $a_{i,t} = \arg\max_a Q_i(s_t, a)$.

By modeling the problem as a MOMDP, we define states, actions, and reward functions:

- **State:** The state is defined by the CS decisions made over the previous k cycles. It is represented as an $M \times k$ matrix, denoted by $\mathbf{S}$, and given as:

$$\mathbf{S} = \left[\mathbf{s}_{-k}, \ldots, \mathbf{s}_0\right], \tag{4}$$

Table 3
DRQN architecture parameters.

Parameter	Value	Parameter	Value
Number of LSTM Layers	1	Number of Dense Layers	1
Input Layer Size	150	Output Layer Size	30

where $\mathbf{s}_{-i}$, $i \in \{1, \ldots, k\}$, represents a column vector that indicates the selected cells in the i-th previous cycle. In this work, we set $k = 5$.

- **Action:** Cells are selected sequentially during each cycle. The selected cell is denoted by a, and the action set, represented as $\mathcal{A} = \{1, 2, \ldots, M\}$, includes all available cells in the current cycle.
- **Reward:** The reward for the inference accuracy objective, denoted by r^j_{inf}, is a positively scaled value exponentially proportional to the accuracy of data inference:

$$r^j_{\text{inf}} = e^{-e_j}, \tag{5}$$

The reward for the popular cell coverage objective, denoted by r^j_{pop}, is defined as:

$$r^j_{\text{pop}} = \begin{cases} 0.5 & \text{if } a \in \mathcal{P} \text{ and } \mathbf{X}(a, j-4 : j) \leq \tau, \\ 0.1 & \text{if } a \in \mathcal{P} \text{ and } \mathbf{X}(a, j-4 : j) > \tau, \\ 0 & \text{if } a \notin \mathcal{P}, \end{cases} \tag{6}$$

where τ is a threshold parameter determined through empirical observations. Consequently, the reward vector is defined as:

$$\mathbf{r}_j = [r^j_{\text{inf}}, r^j_{\text{pop}}]. \tag{7}$$

The agent employs multiple DRQNs, each estimating Q-values for a specific objective. For each objective j_i, we define a vector $\mathbf{q_i}$ containing the Q-values computed by $Q_i(s, a)$ for all possible actions: $\mathbf{q_i} = [Q_i(s, a_0), \ldots, Q_i(s, a_k)]$. To aggregate Q-values across objectives, scalarization combines all $\mathbf{q_i}$ vectors into a single vector, and the action corresponding to the highest value in the resulting scaled Q-value vector is selected for execution. Since Q-values can take any real number, each $\mathbf{q_i}$ vector is rescaled to the interval $[0, 1] \subseteq \mathbb{R}$ to represent normalized preferences. We adopt the scaling function from [22], which maps $\min(\mathbf{q_i})$ to 0 and $\max(\mathbf{q_i})$ to 1, as follows:

$$\text{scl}(\mathbf{q_i}) = \frac{\mathbf{q_i} - \min(\mathbf{q_i})}{\max(\mathbf{q_i}) - \min(\mathbf{q_i})}. \tag{8}$$

This normalization proportionally scales Q-values across different objectives, facilitating effective scalarization in multi-objective decision-making problems. Let $\mathbf{w}$ denote the weight vector corresponding to the priorities assigned to specific objectives. The scalarized Q-vector is defined as follows [22]:

$$\mathbf{q}_d = \mu + \sum_{i=1}^{N} w_i \, \text{scl}(\mathbf{q}_i), \tag{9}$$

where μ is a vector containing very small random values to introduce stochasticity. In our cell selection problem, the weight assigned to maximize inference accuracy is denoted as w_{inf}, while the weight for maximizing the coverage of popular cells is denoted as w_{pop}.

In DRQN (i), corresponding to objective i, the network architecture comprises a single LSTM layer followed by a dense layer, as adopted in the single-objective study in [9]. The LSTM layer processes sequential states (five recent cycles of the cell selection matrix) over time, with the dense layer generating the Q-values. The temporal correlation can be more effectively managed through the use of LSTM. The key parameters of the DRQN architecture are presented in Table 3.

The Q-function is defined as $Q_i(s, h_{k-1}, a, \theta_{i,k})$, where h_{k-1} represents the hidden state carried over from the LSTM layer at the previous time step $k-1$, and $\theta_{i,k}$ denotes the network parameters at time step k. Each DRQN (i) is optimized iteratively using the following loss function for iteration k:

$$L_{i,k}(\theta_{i,k}) = \mathbb{E}_{\langle s,a,r_i,s' \rangle} \left[(r_i + \gamma \max_{a'} Q_i(s', h'_{k-1}, a', \theta'_{i,k}) \right.$$
$$\left. - Q_i(s, h_{k-1}, a, \theta^2_{i,k})) \right]. \tag{10}$$

The training process for each DRQN, which addresses objective i, is described in Algorithm 1 and referred to as DRQN(i). To mitigate oscillations during training, a target network is incorporated. This target network, with parameters θ', is periodically updated by copying the parameters from the primary network. In each cycle, B_c cells are selected. In step 3, the network DRQN(i) employs actions chosen according to the multi-objective-based Algorithm 2. During each cycle, the a_t-th entry of the current cell selection vector is set to 1 for each selected action (cell), as detailed in steps 4–7. Steps 8–11 reset the state for a new cycle by

Algorithm 1 DRQN(i): DRQN Model for Objective i.

Require: B_c ▷ Number of cells to be selected per cycle

1: **Initialize:**
- $t \leftarrow 0$, $B_{\text{cur}} \leftarrow B_c$, $a_0 \leftarrow 0$
- $\mathbf{s}_0 \leftarrow 0_{M\times 1}$
- $\mathbf{S}_t \leftarrow [\mathbf{s}_{-k}, \dots, \mathbf{s}_0]$
- Initialize primary and target neural networks

2: **while** True **do**
3: Select the action a_t based on the multi-objective approach using Algorithm 2
4: **if** $B_{\text{cur}} > 0$ **then**
5: $B_{\text{cur}} \leftarrow B_{\text{cur}} - 1$
6: Update $\mathbf{s}'_0 \leftarrow \mathbf{s}_0 + [0, \dots, 0, 1, 0, \dots, 0]^T$ ▷ Set the a_t-th entry
7: Update $\mathbf{S}_{t+1} \leftarrow [\mathbf{s}_{-k}, \dots, \mathbf{s}'_0]$
8: **else**
9: $B_{\text{cur}} \leftarrow B_c$ ▷ Start a new cycle
10: Reset $\mathbf{s}_1 \leftarrow 0_{M\times 1}$
11: Update $\mathbf{S}_{t+1} \leftarrow [\mathbf{s}_{-k+1}, \dots, \mathbf{s}_0, \mathbf{s}_1]$
12: **end if**
13: Update $\theta_{i,t}$ using loss function (10) based on experience traces from Algorithm 2
14: $t \leftarrow t + 1$
15: **if** $t \,\%\, R_{\text{iter}} == 0$ **then**
16: Update target network: $\theta'_i \leftarrow \theta_{i,t}$
17: **end if**
18: **end while**

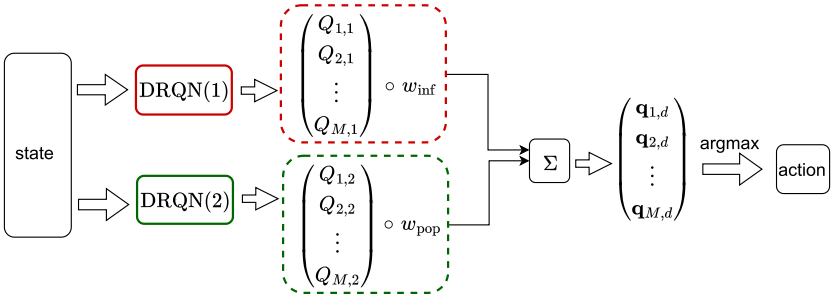


Fig. 3. MODRQN-CS architecture.

appending a zero cell selection vector to the previous four cell selection vectors. During the learning phase, traces provided by Algorithm 2 are utilized to update the network and compute the loss as defined in (10) (step 13). Additionally, the target network is updated every R_{iter} iterations, as described in steps 15–17.

Theorem 2. *Algorithm 1 has a computational complexity of* $O(B_c \cdot K \cdot (c_1 + c_2))$.

Proof. The computational complexity of the LSTM layer at each time step is approximately $c_1 = 4 \cdot n_c^2 + 4 \cdot n_i \cdot n_c + n_c \cdot n_{i+1} + 3 \cdot n_c$, where n_c represents the number of memory blocks and n_i represents the number of neurons in the i-th layer. For the fully connected layer, the complexity is determined by the number of multiplications performed. Specifically, for M_f fully connected layers, the complexity is given by $c_2 = \sum_{j=0}^{M_f-1} m_j \cdot m_{j+1}$, where m_j represents the number of neurons in the j-th layer. Considering that each cycle lasts for B_c time steps and the algorithm runs for K sensing cycles, the overall computational complexity is $O(B_c \cdot K \cdot (c_1 + c_2))$. This completes the proof. □

In the MODRQN-CS architecture, as illustrated in Fig. 3, two DRQNs run in parallel, processing the same input state, with each DRQN corresponding to a specific objective. The input state for both DRQNs consists of the cell selection matrix of the five most recent cycles. The input state is fed into each DRQN, which then outputs a Q-value vector that represents all possible actions (i.e., the number of cells). User-defined preference weights are applied to scale the Q-values generated by the respective DRQNs, determining the selection of the optimal action.

We present the multi-objective-based CS algorithm, MODRQN-CS, in Algorithm 2 and provide a detailed explanation of its steps. In steps 3–5, the scalarized Q-values, $\text{scl}(\mathbf{q}_i)$, are computed for each objective i using Eq. (8), considering the outputs of DRQN(i).

Algorithm 2 MODRQN-CS: Proposed Cell Selection Method.

Require: B_c, w_{inf}, w_{pop}

1: **Initialize:**

- $t \leftarrow 0$, $\boldsymbol{E} \leftarrow \emptyset$, $B_{\text{cur}} \leftarrow B_c$, $a_0 \leftarrow 0$
- $\mathbf{s}_0 \leftarrow 0_{M\times 1}$
- $\mathbf{S}_t \leftarrow \left[\mathbf{s}_{-k}, \ldots, \mathbf{s}_0\right]$

2: **while** True **do**
3: **for** $i \in \mathcal{J}$ **do** ▷ Iterate over all objectives
4: Calculate $\text{scl}(\mathbf{q}_i)$ using (8).
5: **end for**
6: Compute $\mathbf{q}_d$ using (9) considering w_{inf}, w_{pop}.
7: Select action a_t corresponding to the maximum value in $\mathbf{q}_d$.
8: **if** $B_{\text{cur}} > 0$ **then**
9: $B_{\text{cur}} \leftarrow B_{\text{cur}} - 1$
10: Update $\mathbf{s}'_0 \leftarrow \mathbf{s}_0 + \left[0, \ldots, 0, 1, 0, \ldots, 0\right]^T$ ▷ Set a_t-th entry
11: Update $\mathbf{S}_{t+1} \leftarrow \left[\mathbf{s}_{-k}, \ldots, \mathbf{s}'_0\right]$
12: **else**
13: $B_{\text{cur}} \leftarrow B_c$ ▷ Start a new cycle
14: Reset $\mathbf{s}_1 \leftarrow 0_{M\times 1}$
15: Update $\mathbf{S}_{t+1} \leftarrow \left[\mathbf{s}_{-k+1}, \ldots, \mathbf{s}_0, \mathbf{s}_1\right]$
16: **end if**
17: Calculate $\mathbf{r}_t$ using (7).
18: Store $\langle \mathbf{S}_t, a_t, \mathbf{r}_t, \mathbf{S}_{t+1} \rangle$ in $\boldsymbol{E}$.
19: Randomly sample experience traces from $\boldsymbol{E}$ and provide them to each DRQN(i), where $i \in \mathcal{J}$, for learning.
20: Increment $t \leftarrow t + 1$.
21: **end while**

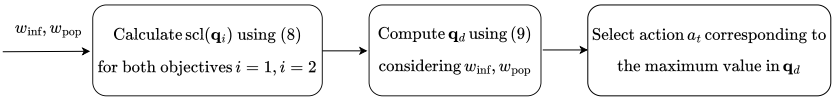


Fig. 4. The effect of the parameters w_{inf} and w_{pop} on action selection.

Next, in step 6, the scalarized Q-vector $\mathbf{q}_d$ is calculated, incorporating the given preference weights, w_{inf} and w_{pop}, which correspond to our two defined objectives: maximizing inference accuracy and maximizing the coverage of popular cells, respectively. During training, we set $w_{\text{inf}} = w_{\text{pop}} = 1$; however, various combinations of randomly assigned weights can be adjusted during the testing phase. The multi-objective action is then selected in step 7 by maximizing the values in $\mathbf{q}_d$. The process of action selection is depicted in Fig. 4.

Steps 8–16 focus on updating the state $\mathbf{S}_{t+1}$ after executing the selected multi-objective action a_t. The update process accounts for whether a sensing cycle has been completed. The reward vector $\mathbf{r}_t$ is subsequently computed in step 17 using Eq. (7). Finally, the transitions $\langle \mathbf{S}_t, a_t, \mathbf{r}_t, \mathbf{S}_{t+1} \rangle$ are stored in a memory pool $\boldsymbol{E}$ (step 18). During the learning phase, each network DRQN(i) updates its parameters by sampling experience traces from $\boldsymbol{E}$ (step 19).

4.1. Discussion

We assume that the target area contains a sufficient number of participants who carry out the sensing tasks reliably and completely. Furthermore, we assume that these participants remain within their respective cells while collecting and transmitting sensing data, allowing the CSP to recruit users in any cell at all times. These assumptions are consistent with those made in previous studies on cell selection [8,9,18].

The main challenge in deploying MODRQN-CS in real-world systems is the potential insufficiency of participants in the target area, leading to unoccupied cells. In particular, user availability and mobility present significant challenges for implementation. Furthermore, maintaining the reliability of users' sensing data and ensuring accurate truth estimation remain critical concerns. A further challenge involves ensuring the robustness of the chosen inference algorithm in noisy conditions.

However, existing strategies in the literature provide potential solutions to these challenges, which we aim to investigate in future work. For example, the study in [4] examines the impact of malicious participants. Moreover, truth discovery techniques can be utilized to assess the reliability of collected data [17]. Similarly, studies such as [10,18] have explored methods to handle

Table 4
Key simulation parameters.

Parameter	Value
Target network update rate (R_{iter})	300
Batch size	64
Cells per cycle (B_c)	8
Discount factor (γ)	0.9
Learning rate (α)	0.01

uncertainty in user mobility. We adopt the Ordinary Kriging algorithm for data inference. However, matrix completion techniques that are robust to noise could be incorporated into our method. For instance, the study in [26] addresses the challenge of robust data inference in the presence of sparse noise. By integrating these approaches, our method could be adapted to relax the aforementioned assumptions.

5. Performance evaluation

In this section, we evaluate the performance of the proposed multi-objective CS method through experiments conducted on a real-world dataset.

5.1. Simulation setup and metrics

We evaluate the proposed CS algorithm using the Sensor-Scope dataset [30], which contains a variety of environmental measurements collected by static stations deployed across the EPFL campus. The dataset includes sensor readings for surface temperature, solar radiation, relative humidity, soil moisture, and wind speed, which are recorded at regular intervals. For this evaluation, we focus on the "surface temperature" readings, with each epoch lasting approximately 30 s.

The dataset comprises approximately 5×10^4 entries. The EPFL campus spans an area of approximately $500 \times 300\text{m}^2$, divided into 100 cells, each measuring $50 \times 30\text{m}^2$. Sensor data is available from 57 of these cells. For simulation purposes, we model a subset of the campus area consisting of 30 cells, covering a total area of approximately $300 \times 150\text{m}^2$, with each cell measuring $50 \times 30\text{m}^2$. Samples from the dataset are used to represent environmental data as sensed by users through their mobile devices.

The Sensor-Scope dataset has been extensively employed in numerous crowdsensing studies for evaluation purposes [8,9,18,19,26,29]. As a relevant environmental monitoring dataset, it aligns well with the context of our environmental application and is appropriate for evaluating the proposed method. Given our focus on the correlation among sensing data to assess inference accuracy, and considering that this dataset provides comprehensive and reliable evaluation capabilities, we have selected it for our study.

The experimental setup comprises an Intel® Core™ i7-8550U CPU @ 1.80 GHz × 8 and 16 GB of RAM. For the parameters, we set the discount factor to $\gamma = 0.9$ and the learning rate to $\alpha = 0.01$. We update the target network every 300 iterations and set the batch size to 64. The exploration rate ϵ is dynamically adjusted from 1 to 0.1 throughout the entire training process. These parameter values are tuned to achieve optimal results in simulation, with some being adopted from [9,18]. Table 4 summarizes the key simulation parameters.

We compare the MODRQN-CS algorithm with the following baseline methods to evaluate its performance:

1. **DV-MODRQN-CS**: This variant incorporates decision values when calculating Eq. (9), following the framework presented in [22] and leveraging DRQN. The effect of decision values is explicitly considered during scalarization.
2. **S-MODRQN-CS**: In this baseline, an agent is trained to maximize inference accuracy and coverage by leveraging DRQN and considering a simplified reward function $r^j = v_j e^{-e_j}$, where v_j is defined as:

$$v_j = \begin{cases} 1 & \text{if } a \in \mathcal{P} \text{ and } \mathbf{X}(a, j-4 : j) \leq \tau, \\ 0.2 & \text{if } a \in \mathcal{P} \text{ and } \mathbf{X}(a, j-4 : j) > \tau, \\ 0 & \text{if } a \notin \mathcal{P}. \end{cases} \tag{11}$$

3. **S-MODQN-CS**: In this baseline, an agent is trained using DQN to maximize inference accuracy and coverage while employing the reward function defined in the baseline S-MODRQN-CS.

We evaluate the algorithms using the following metrics for CS:

- **Average Inference Accuracy Reward (AIAR):** The average reward obtained by maximizing inference accuracy, calculated for each selected cell after the specified sensing cycles.
- **Average Temporal Coverage Reward (ATCR):** The average reward obtained by maximizing the temporal coverage of specified popular cells, calculated for each selected cell after the specified sensing cycles.
- **Average Total Reward (ATR):** The average total reward achieved after the specified sensing cycles.

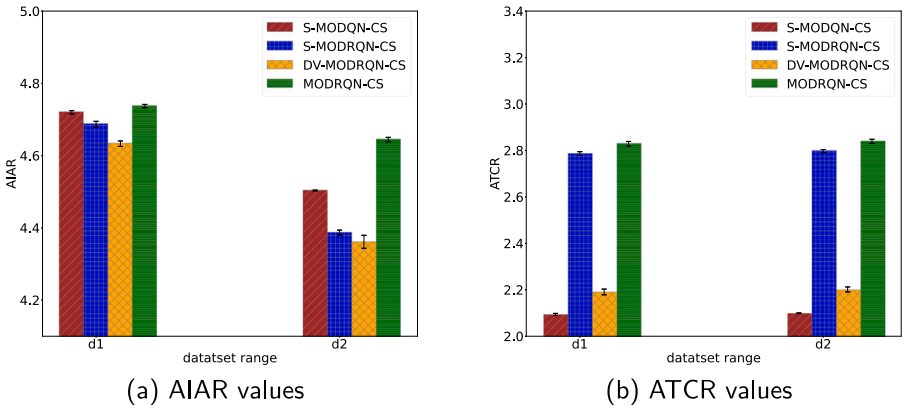


Fig. 5. Average rewards (AIAR and ATCR) for two dataset ranges using the popular cell set $\mathcal{P}_1$.

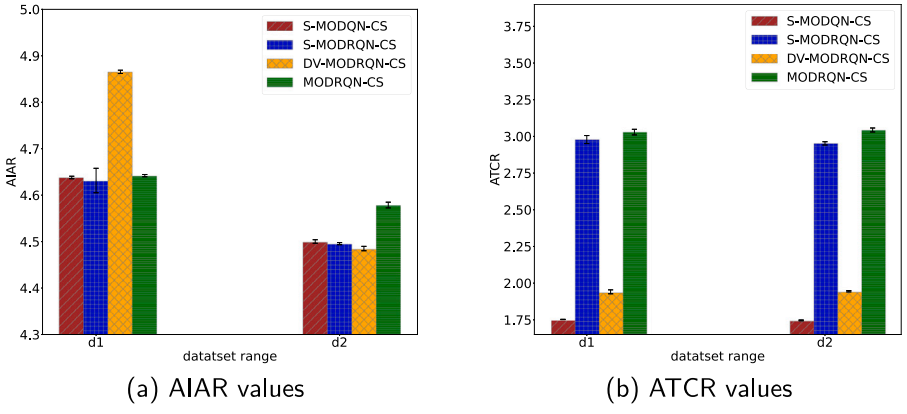


Fig. 6. Average rewards (AIAR and ATCR) for two dataset ranges using the popular cell set $\mathcal{P}_2$.

5.2. Evaluation results

We conducted two independent simulations, each consisting of one thousand cycles on the test dataset, denoted as d_1 and d_2, to evaluate the reliability and effectiveness of the proposed MODRQN-CS. A fixed budget of $B = 8$ was used for all runs.

Popular Cells: To evaluate the algorithm's performance across different sets of popular cells, we considered three distinct sets:

$$\mathcal{P}_1 = \{c_1, c_2, c_9, c_{10}, c_{12}, c_{13}, c_{19}, c_{20}, c_{21}, c_{28}, c_{29}, c_{30}\},$$
$$\mathcal{P}_2 = \{c_4, c_5, c_6, c_7, c_9, c_{10}, c_{17}, c_{18}, c_{19}, c_{20}, c_{21}, c_{26}\},$$
$$\mathcal{P}_3 = \{c_2, c_3, c_6, c_7, c_8, c_{12}, c_{16}, c_{18}, c_{19}, c_{21}, c_{22}, c_{25}\}.$$

The results for both test datasets d_1 and d_2 with training weights $w_{\text{inf}} = w_{\text{pop}} = 1$ are presented in Figs. 5, 6, and 7 for $\mathcal{P}_1$, $\mathcal{P}_2$, and $\mathcal{P}_3$, respectively. As depicted in these figures, the trained MODRQN-CS achieves higher AIAR and ATCR values compared to DV-MODRQN-CS, S-MODRQN-CS, and S-MODQN-CS in most runs. Specifically, MODRQN-CS improves the AIAR up to 14% and the ATCR up to 10%. This is due to MODRQN-CS's enhanced ability to manage action selection by utilizing different policies derived from distinct DRQNs at each step. Consequently, the agent tends to select actions that align with the best objective for a given state.

Various weights: We compare two scalarization-based methods, MODRQN-CS and DV-MODRQN-CS, by considering different weight combinations for the two objectives across $\mathcal{P}_1$, $\mathcal{P}_2$, and $\mathcal{P}_3$ in a post-learning evaluation. The weight combination, denoted as $W^s = (w_{\text{inf}}, w_{\text{pop}})$, is considered according to Table 5. The trained DRQNs are obtained using baseline weights of $w_{\text{inf}} = w_{\text{pop}} = 1$. The results for both test datasets, d_1 and d_2, under varying values of W^s, are presented in Figs. 8(a) and 8(b) for $\mathcal{P}_1$, Figs. 9(a) and 9(b) for $\mathcal{P}_2$, and Figs. 10(a) and 10(b) for $\mathcal{P}_3$.

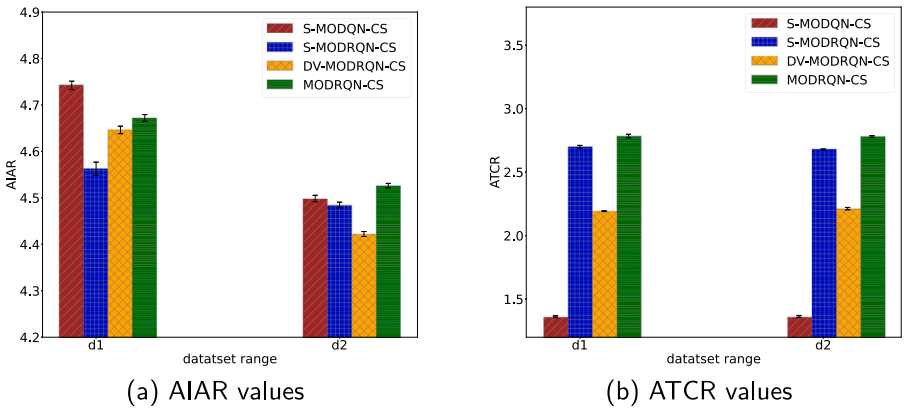


Fig. 7. Average rewards (AIAR and ATCR) for two dataset ranges using the popular cell set $\mathcal{P}_3$.

Table 5
Scalarization weights.

Parameter	Value
W_1^s	(1,1)
W_2^s	(1,0)
W_3^s	(0,1)
W_4^s	(0.1, 0.9)
W_5^s	(0.9, 0.1)
W_6^s	(0.3, 0.7)
W_7^s	(0.7, 0.3)
W_8^s	(0.6, 0.4)
W_9^s	(0.4, 0.6)

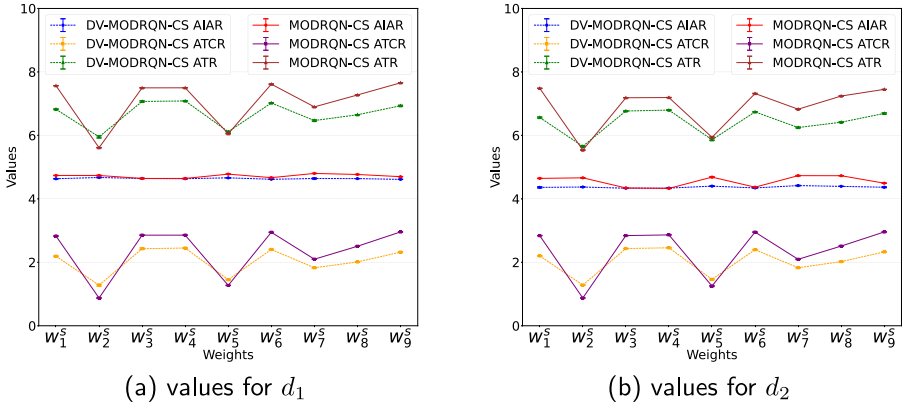


Fig. 8. scalarization comparison for two dataset ranges using $\mathcal{P}_1$.

As observed in Figures, both scalarization-based methods demonstrate strong performance when weights are adjusted post-learning. In this study, the emphasis lies on the ATR value itself, as it reflects the agent's overall performance. Across all figures, a direct comparison of the ATR values reveals that MODRQN-CS consistently outperforms DV-MODRQN-CS, achieving higher values across all weight combinations in nearly all runs. The improvements achieved by AIAR are lower than those of ATCR when comparing MODRQN-CS with DV-MODRQN-CS. Specifically, MODRQN-CS enhances AIAR by up to 43% and ATCR by up to 112% in comparison with DV-MODRQN-CS. This is because both MODRQN-CS and DV-MODRQN-CS achieve high inference accuracy by leveraging the existing correlation among sensing data.

Table 6 presents the runtime for both the training and testing phases. The testing duration is measured after completing 1000 episodes. In our experiments, given the specifications of the equipped machine, the training phase of MODRQN-CS requires significantly more time compared to S-MODQN-CS and S-MODRQN-CS, which is reasonable for real-world deployments, as the training phase is performed offline [9]. Although using DQN would reduce training time, it lacks efficiency in achieving optimal results, as previously shown. We consider the extra time spent on offline training a worthwhile investment to enhance efficiency and reliability.

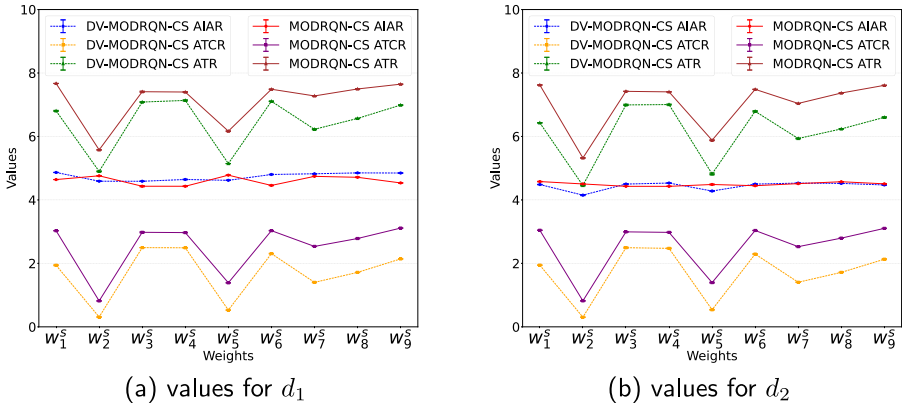


Fig. 9. scalarization comparison for two dataset ranges using $\mathcal{P}_2$.

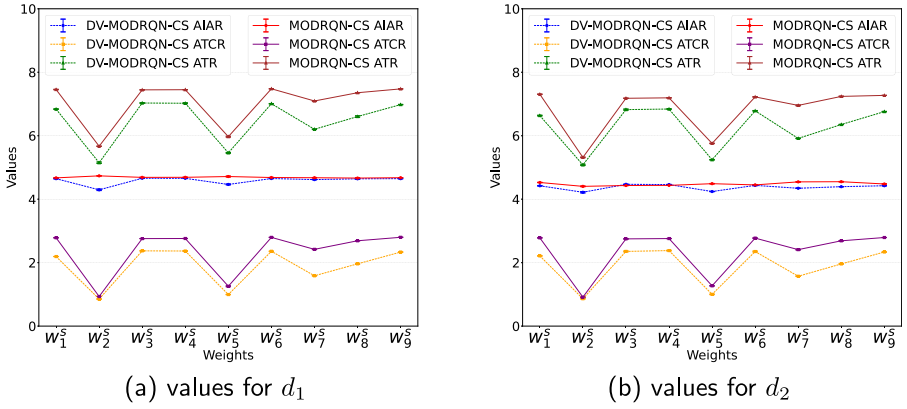


Fig. 10. scalarization comparison for two dataset ranges using $\mathcal{P}_3$.

Table 6
Average runtime of algorithms.

	Training (s)	Testing (s)
S-MODQN-CS	641	36
S-MODRQN-CS	12,488	47
DV-MODRQN-CS	22,616	43
MODRQN-CS	19,284	45

6. Conclusion

This paper addressed the dual objectives of maximizing inference accuracy and ensuring temporal coverage of popular cells in the context of the cell selection problem in crowd sensing for environmental monitoring applications. We proposed a novel cell selection strategy that effectively incorporated data correlations and adhered to budget constraints, enabling efficient cell selection. By employing MODRL, our approach simultaneously targeted both informative and prioritized cells, utilizing a flexible framework that allowed for the adjustment of priorities post-training. We utilized DRQN for each objective to effectively capture and address temporal dependencies within the environmental data. Notably, the proposed MODRQN-CS enabled the deactivation of specific DRQNs, providing adaptability based on application needs. The results showed that MODRQN-CS outperforms baseline methods in terms of AIAR and ATCR values. Despite these advancements, further refinement is needed to optimize the influence of individual objectives on the agent's behavior. Future work could explore alternative learning paradigms and expand the problem to include dimensions like dynamic user behaviors and environmental changes. Additionally, addressing sparse user participation, unreliable sensing data, and robustness to noise presents promising directions. The multi-objective system could also be extended to incorporate additional objectives, such as energy efficiency and fairness. Incorporating additional datasets would further strengthen the robustness of our analysis, an aspect we plan to address in future work. Additionally, extending the proposed approach to other domains, such as UAV-based sensing, or areas like health monitoring and emergency response, along with relevant datasets, offers a promising direction for future research.

CRediT authorship contribution statement

Shabnam Seradji: Conceptualization, Formal analysis, Methodology, Writing – original draft, Visualization, Investigation. **Ahmad Khonsari:** Conceptualization, Supervision, Writing – review & editing, Validation. **Mahdi Dolati:** Supervision, Writing – review & editing, Validation. **Vahid Shah-Mansouri:** Supervision, Writing – review & editing, Validation.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work the author(s) used ChatGPT-4o in order to only improve the quality of the English text. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the publication.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

We used publicly available datasets.

References

[1] Bharti M, Kumar R, Saxena S, Jindal H. Optimal resource selection framework for internet-of-things. Comput Electr Eng 2020;86:106693.
[2] Han L, Yu Z, Wang L, Yu Z, Guo B. Keeping cell selection model up-to-date to adapt to time-dependent environment in sparse mobile crowdsensing. IEEE Internet Things J 2021;8(18):13914–25.
[3] Nguyen TT, T. Nguyen T, Nguyen T-H, Nguyen PL. Fuzzy Q-learning-based opportunistic communication for MEC-enhanced vehicular crowdsensing. IEEE Trans Netw Serv Manag 2022;19(4):5021–33.
[4] Sun P, Wang Z, Wu L, Shao H, Qi H, Wang Z. Trustworthy and cost-effective cell selection for sparse mobile crowdsensing systems. IEEE Trans Veh Technol 2021;70(6):6108–21.
[5] Hossain MY, Rahman T. A crowdsource based framework for Bengali scene text data collection and detection. Comput Electr Eng 2023;112:109025.
[6] Zhao G, Chen R, Wang J. A cooperative detection scheme for malicious nodes based on DS trust evidence reasoning in mobile crowdsensing networks. Comput Electr Eng 2024;118:109456.
[7] Ding X, Lv R, Pang X, Hu J, Wang Z, Yang X, et al. Privacy-preserving task allocation for edge computing-based mobile crowdsensing. Comput Electr Eng 2022;97:107528.
[8] Wang L, Liu W, Zhang D, Wang Y, Wang E, Yang Y. Cell selection with deep reinforcement learning in sparse mobile crowdsensing. In: Proc. IEEE int. conf. on distr. comput. syst.. 2018, p. 1543–6.
[9] Liu W, Wang L, Wang E, Yang Y, Zeghlache D, Zhang D. Reinforcement learning-based cell selection in sparse mobile crowdsensing. Comm Com. Inf Sc. 2019;161:102–14.
[10] Guo X, Tu C, Hao Y, Yu Z, Huang F, Wang L. Online user recruitment with adaptive budget segmentation in sparse mobile crowdsensing. IEEE Internet Things J 2024;11(5):8526–38.
[11] Kong L, Xia M, Liu X-Y, Wu M-Y, Liu X. Data loss and reconstruction in sensor networks. In: Proc. IEEE int. conf. on computer commun. (INFOCOM). 2013, p. 1654–62.
[12] Van Beers WC, Kleijnen JP. Kriging interpolation in simulation: a survey. In: Proceedings of the 2004 winter simulation conference, 2004.. vol. 1, IEEE; 2004, p. 121.
[13] He Z, Cao J, Liu X. High quality participant recruitment in vehicle-based crowdsourcing using predictable mobility. In: 2015 IEEE conference on computer communications. INFOCOM, IEEE; 2015, p. 2542–50.
[14] Liu Y, Yu Z, Cui H, Helal S, Guo B. Safecity: A heterogeneous mobile crowd sensing system for urban public safety. IEEE Internet Things J 2023;10(20):18330–45.
[15] Cho Y, Shin M, Man KL, Kim M. SafeWitness: Crowdsensing-based geofencing approach for dynamic disaster risk detection. Fractal Fract 2025;9(3).
[16] Wu E, Peng Z. Research progress on incentive mechanisms in mobile crowdsensing. IEEE Internet Things J 2024;11:24621–33.
[17] Wang X, Jia R, Tian X, Gan X, Fu L, Wang X. Location-aware crowdsensing: Dynamic task assignment and truth inference. IEEE Trans Mob Comput 2020;19(2):362–75.
[18] Liu W, Yang Y, Wang E, Wu J. User recruitment for enhancing data inference accuracy in sparse mobile crowdsensing. IEEE Internet Things J 2020;7(3):1802–14.
[19] Liu W, Wang E, Yang Y, Wu J. Worker selection towards data completion for online sparse crowdsensing. In: Proc. IEEE int. conf. on computer commun.. INFOCOM, 2022, p. 1509–18.
[20] Tran ND, Dao MC, Nguyen TH, Dinh THL, Nguyen K, Nguyen PL. A deep reinforcement learning-based multi-objective optimization for crowdsensing-based air quality monitoring systems. In: Asian conference on intelligent information and database systems. Springer; 2023, p. 436–48.
[21] Van Moffaert K, Drugan MM, Nowé A. Scalarized multi-objective reinforcement learning: Novel design techniques. In: 2013 IEEE symposium on adaptive dynamic programming and reinforcement learning. ADPRL, IEEE; 2013, p. 191–9.
[22] Tajmajer T. Modular multi-objective deep reinforcement learning with decision values. In: 2018 federated conference on computer science and information systems (fedCSIS). IEEE; 2018, p. 85–93.
[23] Yang R, Sun X, Narasimhan K. A generalized algorithm for multi-objective reinforcement learning and policy adaptation. In: Proceedings of the 33rd international conference on neural information processing systems. Curran Associates Inc.; 2019, p. 14636–47.
[24] Nguyen TT, Nguyen ND, Vamplew P, Nahavandi S, Dazeley R, Lim CP. A multi-objective deep reinforcement learning framework. Eng Appl Artif Intell 2020;96:103915.
[25] Wei X, Li Z, Liu Y, Gao S, Yue H. SDLSC-TA: Subarea division learning based task allocation in sparse mobile crowdsensing. IEEE Trans Emerg Top Comput 2020;9(3):1344–58.

[26] Li C, Li Z, Long S, Qiao P, Yuan Y, Wang G. Robust data inference and cost-effective cell selection for sparse mobile crowdsensing. IEEE/ACM Trans Netw 2024;32:3760–75.
[27] Tu C, Yu Z, Han L, Guo X, Huang F, Guo W, et al. Adaptive budgeting for collaborative multi-task data collection in online sparse crowdsensing. IEEE Trans Mob Comput 2023;1–17.
[28] Wang E, Tian Z, Yang Y, Liu W, Li B, Jiang N, et al. Data-driven similarity-based worker recruitment towards multi-task data inference for sparse mobile crowdsensing. In: 2023 IEEE/ACM 31st international symposium on quality of service (iWQoS). IEEE; 2023, p. 1–10.
[29] Zhu Z, Chen B, Liu W, Zhao Y, Liu Z, Zhao Z. A cost-quality beneficial cell selection approach for sparse mobile crowdsensing with diverse sensing costs. IEEE Internet Things J 2021;8(5):3831–50.
[30] Ingelrest F, Barrenetxea G, Schaefer G, Vetterli M, Couach O, Parlange M. SensorScope: Application-specific sensor network for environmental monitoring. ACM Trans Sens Networks 2010;6(2):1–32.