

# Adaptive Metric Selection for Monitoring Cloud–Edge–IoT Continuum Systems Using Reinforcement Learning

Ahmad Atwi, Abdelkader Lahmadi

Université de Lorraine, CNRS, Inria, Loria, F-54000 Nancy, France, {firstname.lastname}@inria.fr

**Abstract**—Modern distributed systems, such as the Cloud–Edge–IoT continuum, are becoming increasingly complex, heterogeneous, and large-scale, making continuous monitoring essential for ensuring reliability and detecting failures. Yet, monitoring such systems can generate large volumes of data across many nodes, services, and devices, which makes exhaustive metric collection costly, resource-intensive, and difficult to manage especially at the edge/IoT layer, in terms of collection, transmission, processing, and storage overhead. To address this issue, we focus on adaptive metric selection under monitoring constraints. We propose a reinforcement learning based framework that learns to adaptively select at each monitoring window a compact and informative subset of metrics by prioritizing anomalous and causally influential signals while balancing long-term monitoring utility and resource usage. The approach is implemented using a Deep Sets inspired actor-critic policy trained with PPO and evaluated on both synthetic and real-world datasets. Experimental results show that the proposed method reduces monitoring volume by up to 66% compared to exhaustive monitoring and 33% compared to capacity-aware baselines, while maintaining strong anomaly coverage and importance preservation.

**Index Terms**—Reinforcement learning, resource-aware monitoring, metric subset selection, anomaly detection, causal analysis.

## I. INTRODUCTION

Monitoring Cloud–Edge–IoT continuum systems is a foundational yet challenging task. These systems span a wide range of heterogeneous resources, from high-capacity cloud servers to resource-constrained edge and IoT devices distributed across geographic and administrative boundaries [1]. To ensure reliable performance, fault detection, and resource optimization, system components must be continuously monitored through the collection of telemetry data, including logs, traces, and metrics. In this work, we focus specifically on metrics, as they provide continuous quantitative indicators of system behavior and are widely used for performance analysis, anomaly detection, and operational diagnosis.

The main challenge is that observability cost is strongly tied to the amount of monitoring data that is continuously collected, transmitted, processed, and stored. As systems scale with more services, interactions, devices, and user requests, the number of collected metrics and time series also increases, especially when metrics include high-cardinality labels or wide-ranging dimensions. In cloud environments, this directly translates into monetary cost, since providers charge customers based on collected monitoring data and metrics [2]. However,

many collected metrics may never be used in dashboards, alerts, or diagnosis, leading to storage and processing costs with limited operational value [3]–[5]. This issue is even more critical for edge and IoT devices, which operate under limited CPU, memory, bandwidth, energy, and storage resources. It is further amplified when the set of collected metrics and their collection frequency are configured in advance and remain mostly static during operation, regardless of the current system state or the actual diagnostic value of each metric. Therefore, exhaustive metric collection may be acceptable in some cloud settings, but it becomes costly or impractical in resource-constrained continuum systems. This motivates adaptive monitoring approaches that decide not only how often telemetry should be collected, but also which metrics should be selected under resource constraints.

Reinforcement Learning (RL) provides a principled framework for sequential decision-making under uncertainty and has been widely explored for resource management in networking, distributed systems, and IoT networks [6], [7]. In the monitoring context, unlike static monitoring strategies, RL can learn policies that account for long-term trade-offs between metric informativeness, monitoring cost, and capacity constraints. In addition, anomaly detection and causal analysis provide valuable insights into system behavior: anomaly scores can highlight metrics that reflect abnormal conditions, while causal relationships can identify structurally influential metrics. However, their use as informative priors for adaptive metric selection remains underexplored.

In this paper, we propose a RL-based framework for adaptive metric selection under resource constraints. We formulate metric selection as a sequential decision making problem where, at each monitoring window, the agent selects a subset of metrics to monitor in the next window. The policy is guided by anomaly and causal indicators and learns to balance monitoring utility, resource usage, and soft node-level capacity constraints. To model the structured nature of the decision, we implement the policy using a Deep Sets inspired actor-critic architecture [8] optimized with PPO [9]. The policy represents node-metric pairs with shared parameters and combines local metric features with a global monitoring context to produce multi-binary selection decisions. We evaluate the proposed approach in controlled monitoring environments with heterogeneous node capacities and dynamically evolving metric behaviors derived from synthetic and real-world datasets. The

main contributions of this work are summarized as follows:

- We formulate adaptive metric selection for Cloud-Edge-IoT monitoring as a fine-grained sequential decision making problem under soft node-level resource constraints.
- We adapt a Deep Sets inspired actor-critic architecture that represents node-metric pairs with shared parameters and combines local metric information with a global monitoring context to produce multi-binary selection decisions.
- We introduce an importance-driven monitoring strategy that integrates anomaly and causal indicators, enabling the agent to prioritize informative metrics while reducing unnecessary monitoring.
- We evaluate the proposed approach against multiple baselines on synthetic and real-world datasets, showing that it can substantially reduce monitoring volume and cost while preserving monitoring utility under heterogeneous constraints.

The remainder of this paper is organized as follows. Section II reviews related work on monitoring systems, anomaly detection, causal analysis, and RL. Section III presents the problem formulation and the proposed RL-based metric selection approach. Section IV describes the experimental setup and discusses the results. Finally, Section V concludes the paper and outlines future research directions.

## II. RELATED WORK

### A. Monitoring Systems and Adaptive Monitoring

Widely adopted monitoring systems like Prometheus [10] operate under a pull-based model, periodically scraping all exposed metrics from configured targets. In practice, exporters expose dozens or even hundreds of metrics per node, typically collected uniformly at fixed intervals. While this design simplifies observability and deployment, it does not distinguish between informative and non-informative metrics, nor does it adapt to evolving system conditions. These limitations motivate the need for adaptive metric selection mechanisms that prioritize relevant signals while respecting system-level capacity constraints, a capability not addressed by traditional monitoring systems.

Several works have investigated adaptive monitoring strategies for resource-constrained environments. For instance, [11] proposes low-cost adaptive monitoring techniques that dynamically adjust sampling frequency to reduce communication and energy overhead in IoT systems. Similarly, [12] leverage probabilistic time-series forecasting to adapt monitoring intensity in edge computing environments. These approaches improve efficiency by adjusting monitoring frequency [13] or scheduling policies. In contrast, our work focuses on what to monitor under explicit per-node monitoring budgets.

### B. Anomaly Detection and Causal Discovery

Anomaly detection (AD) plays a central role in system monitoring, where time-series metrics are analyzed to identify performance degradation, faults, or abnormal operational

conditions. A wide range of AD approaches have been proposed, including statistical, isolation-based, and deep learning methods [14]–[18]. In this work, we do not aim to propose a new AD method. Instead, we use anomaly scores as decision signals that help the monitoring agent prioritize metrics that are more likely to reflect abnormal system behavior.

Causal inference, on the other hand, aims to identify directional relationships among variables and determine how changes in one metric may influence others. Several families of causal discovery methods have been proposed for both i.i.d. and time-series data, including constraint-based and functional causal model-based approaches, as well as Granger-causality-based methods for temporal settings [19], [20]. In monitoring environments, understanding causal dependencies between time-series metrics is useful for distinguishing root causes from secondary effects and for prioritizing structurally influential signals.

### C. Reinforcement Learning

RL has been increasingly used for adaptive resource management in distributed systems, networking, and wireless IoT environments [21]. For example, Mao et al. [6] showed that deep RL can learn resource management policies as an alternative to hand-crafted heuristics. Xu et al. [22] proposed an experience-driven DRL framework for traffic engineering in communication networks, where the agent learns resource allocation decisions from network experience. More recently, DRL has also been investigated for online resource allocation in IoT networks, highlighting its potential for dynamic and resource-constrained environments [7].

These works show the relevance of RL for decision-making in networked systems. However, most existing RL-based resource-management approaches focus on allocating computational, storage, or communication resources. They do not directly address the problem of deciding which monitoring metrics should be collected at fine granularity under node-level capacity constraints. In our setting, the monitoring action is not a single allocation decision, but a set of simultaneous binary decisions over many node-metric pairs. The agent must therefore balance metric usefulness, monitoring cost, and capacity violations over time.

This makes adaptive metric selection different from classical resource allocation problems. Since the decision is defined over a collection of node-metric pairs, set-based neural architectures such as Deep Sets [8] provide a useful modeling perspective through shared representations and aggregation. Building on this idea, our work formulates adaptive metric selection as an RL problem for resource-aware monitoring, where anomaly and causal indicators guide the selection policy under soft capacity constraints.

## III. REINFORCEMENT LEARNING-BASED DYNAMIC METRIC SELECTION

We propose a RL-based approach for dynamic metric selection in Cloud-Edge-IoT environments under resource constraints. The proposed framework combines monitoring

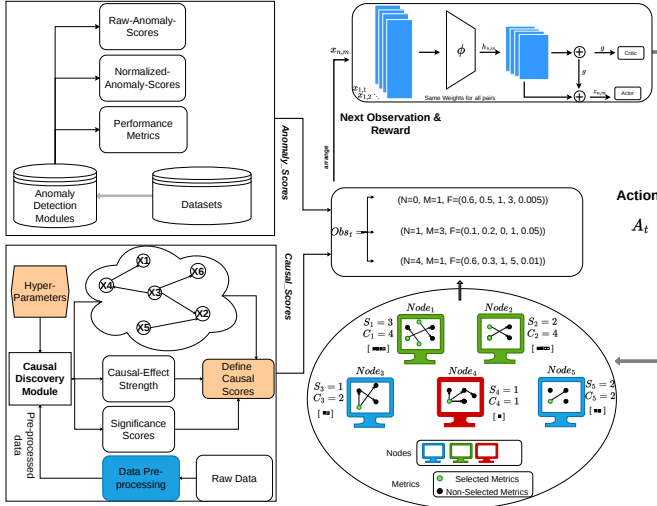


Fig. 1. Overview of our RL-based learning framework for selecting the subset of monitoring metrics at each monitoring window.

indicators, resource context, and a learning-based selection policy to decide which metrics should be monitored over time. Figure 1 illustrates the overall pipeline, including the anomaly detection module, the causal discovery module, and the RL-based monitoring agent.

#### A. System Model

We consider a system composed of  $N$  nodes, where each node  $n \in \{1, \dots, N\}$  exposes  $\mathcal{M}$  monitorable metrics. The nodes belong to different layers of the computing continuum (Cloud, Edge, and IoT), and are characterized by two layer dependent monitoring constraints: a soft monitoring budget  $MC_n$ , which defines the preferred number of metrics that can be monitored simultaneously on node  $n$ , and a per-metric monitoring cost that reflects the relative overhead of collecting, transmitting, processing, and storing each metric. These quantities abstract local resource limitations, including CPU, memory, bandwidth, energy, and storage capacity. Therefore, resource constrained IoT and edge nodes are assigned smaller monitoring budgets and higher monitoring costs, while cloud nodes can support larger budgets and lower relative costs due to their higher computational, communication, and storage capabilities.

#### B. Problem Formulation

We formulate dynamic metric selection as a RL problem defined by the tuple

$$(S, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma),$$

where  $S$  denotes the state space,  $\mathcal{A}$  the action space,  $\mathcal{T}$  the transition dynamics,  $\mathcal{R}$  the reward function, and  $\gamma \in (0, 1]$  the discount factor.

At the beginning of each monitoring window  $w$ , the agent observes the current state  $s_w \in S$ , constructed from the latest available monitoring indicators and resource context. Based

on this state, it selects an action  $a_w \in \mathcal{A}$  corresponding to the subset of metrics to monitor during the next window. After that window is evaluated, the environment provides a reward  $r_w$  reflecting the usefulness of the selected metrics and their resource usage, and transitions to the next state  $s_{w+1}$ . The objective is to learn a policy that maximizes the expected discounted return.

Node-level capacity constraints are not enforced as hard action constraints, instead violations are incorporated into the reward as penalty terms, allowing the agent to learn constraint-aware metric selection through experience.

#### C. Anomaly and Causal Scores

The proposed policy uses anomaly and causal indicators as complementary signals of metric importance. Anomaly scores capture the temporal abnormality of each metric, while causal scores reflect its structural influence on other system metrics.

To estimate anomaly scores, we use TimeEval [23], a benchmarking framework for time-series AD. In particular, we rely on LSTM-AD to compute anomaly scores for metrics over time. The score is based on the variance-normalized squared deviation of the mean absolute prediction error. Since raw anomaly scores may differ across metrics, we use normalized scores to make them comparable. At each monitoring window, the anomaly indicator of a metric is obtained by aggregating its anomaly scores within that window.

For causal importance estimation, we use PCMCi+ [19], a method designed to identify causal relationships in multivariate time-series data. For each metric  $i$ , we compute a static causal influence score by aggregating its statistically significant outgoing causal effects on the other metrics in the system across the considered time lags. Formally, the causal score is defined as

$$\text{CausalScore}(i) = \sum_{j \in \mathcal{M}, i \neq j} \sum_{\tau \in \mathcal{T}} \mathbf{1}(i_{\tau} \rightarrow j) \cdot |\text{val}_{i \rightarrow j}(\tau)|,$$

where  $\mathcal{M}$  denotes the set of metrics in the system,  $\mathcal{T}$  the set of considered time lags, and  $\text{val}_{i \rightarrow j}(\tau)$  represents the estimated causal strength at lag  $\tau$ .

Both anomaly and causal indicators can in principle be updated as new monitoring data becomes available, allowing the selection policy to be integrated into an online monitoring pipeline. In this work, however, we compute these indicators from historical training data to focus the evaluation on the metric selection policy itself.

#### D. State Space

The state space is represented as a three-dimensional tensor

$$s_w \in \mathbb{R}^{N \times M \times 5},$$

where each metric on each node is represented by the following 5 features:

- **Anomaly score:** Previous window anomaly score reflecting abnormal behavior of the metric.
- **Causal score:** Metric's causal influence on the system.

- **Placement history:** Boolean, true if the metric was selected for monitoring at the previous monitoring window.
- **Capacity ratio:** Normalized node level monitoring capacity.
- **Cost feature:** A normalized per-metric feature that represent the resource effort to collect, transmit, process, and store the metric.

This representation provides the agent with the latest available system signals and historical monitoring context, while explicitly exposing capacity constraints.

#### E. Action Space

At each monitoring window  $w$ , the agent selects a subset of metrics to monitor across all nodes. The action is represented as a multi-binary decision matrix

$$a_w \in \{0, 1\}^{N \times M}$$

where each entry  $a_{n,m}(w)$  corresponds to the monitoring decision for metric  $m$  on node  $n$ . An action value  $a_{n,m}(w) = 1$  indicates that the metric is selected for monitoring, while  $a_{n,m}(w) = 0$  indicates that it is not selected. This formulation induces a high-dimensional action space, allowing the policy to output one logit for each node-metric pair.

#### F. Reward Function

The reward function is designed to guide the agent toward selecting a compact and informative subset of metrics while reducing monitoring cost and limiting capacity violations. At each monitoring window  $w$ , the agent receives a positive reward for selecting metrics with high monitoring importance, and penalties for exceeding capacities, increasing monitoring cost, changing selections unnecessarily, or selecting no metrics. The reward is defined as:

$$r_w = \alpha r_{\text{imp}(w)} - \beta r_{\text{cap}(w)} - \lambda r_{\text{cost}(w)} - \delta r_{\text{switch}(w)} - \eta r_{\text{idle}(w)},$$

where the coefficients  $\alpha$ ,  $\beta$ ,  $\lambda$ ,  $\delta$ , and  $\eta$  control the relative weight of each term. These weights are selected empirically and kept fixed across experiments unless stated otherwise (last row in II).

*a) Importance reward:* Encourages the selection of metrics that are either anomalous or causally influential (above their baselines). Let  $AS_{n,m}(w)$  denote the anomaly score of metric  $m$  on node  $n$  at window  $w$ , and let  $CS_{n,m}$  denote its causal score. The importance reward is defined as:

$$r_{\text{imp}(w)} = \sum_{n=1}^N \sum_{m=1}^M a_{n,m}(w) \left( \max(0, AS_{n,m}(w) - b_a) + \max(0, CS_{n,m} - b_c) \right)$$

where  $AS_{n,m}(w) = \max_{t \in w} AS_{n,m}(t)$ , and  $b_a$ ,  $b_c$  define minimum anomaly and causal informativeness levels, with higher rewards assigned to metrics whose scores exceed these thresholds by larger margins.

*b) Capacity Penalty:* The goal of this penalty is to penalize the agent when it exceeds the preferred capacity ( $MC_n$ ). The capacity overflow is defined as

$$r_{\text{cap}(w)} = \sum_{n=1}^N \max(0, |S_{n,w}| - MC_n).$$

where  $S_{n,w}$  denote the set of selected metrics on node  $n$  at window  $w$ .

*c) Cost Penalty:* This term complements the capacity penalty by accounting for the additional heterogeneous costs in the system.

$$r_{\text{cost}(w)} = \sum_{n=1}^N \sum_{m=1}^M a_{n,m}(w) k_{n,m}.$$

*d) Additional Penalties:* The switching penalty  $r_{\text{switch}}$  penalizes changes in monitoring decisions between consecutive windows, but only for metrics that are not informative, thereby discouraging unnecessary reconfigurations. The idle penalty  $r_{\text{idle}}$  prevents degenerate policies that select no metrics.

#### G. Policy Architecture

The monitoring decision is defined over a collection of node-metric pairs rather than over a single scalar resource. This motivates the use of a Deep Sets-inspired actor-critic architecture [8], where each node-metric pair is processed with shared parameters before being aggregated into a global representation. This structure is well suited to our setting because the same notion of metric relevance can be learned across all node-metric pairs, while preserving the local structure of each monitoring decision and avoiding the treatment of the full monitoring state as a single unstructured vector.

*a) Local encoding:* We represent the state at monitoring window  $w$  as a collection of node-metric feature vectors. Each vector corresponds to one pair  $(n, m)$  and contains the features defined in Section III-D:

$$x_{n,m}^{(w)} \in \mathbb{R}^F.$$

Each vector is independently encoded by a shared two-layer MLP  $\phi_\theta$  with LayerNorm and ReLU activations:

$$h_{n,m}^{(w)} = \phi_\theta(x_{n,m}^{(w)}) \in \mathbb{R}^{d_\ell}.$$

Although the encoder is shared, layer-aware behaviour is preserved: features encoding node cost  $c_n$  and capacity  $MC_n$  differ across layers, so metrics on different layers produce distinct local embeddings even when their anomaly scores are identical.

*b) Global context:* The local embeddings are aggregated into a single global context vector that summarises the overall monitoring state:

$$\mathbf{g}^{(w)} = \rho_\theta \left( \text{pool} \left( \left\{ h_{n,m}^{(w)} \right\}_{n,m} \right) \right) \in \mathbb{R}^{d_g},$$

where pool is mean pooling, and  $\rho_\theta$  is a learned linear projection followed by LayerNorm and ReLU.

c) *Actor head*: Following the segmentation branch of PointNet [24], the global context is concatenated with each local embedding before producing per-pair outputs. This allows the policy to make individual selection decisions while conditioning each decision on the overall monitoring state:

$$z_{n,m}^{(w)} = \left[ h_{n,m}^{(w)} \parallel \mathbf{g}^{(w)} \right] \in \mathbb{R}^{d_\ell + d_g}.$$

The actor then outputs one logit for each node-metric pair, defining independent Bernoulli decisions and producing the multi-binary action matrix  $a_w \in \{0, 1\}^{N \times M}$ . The critic uses the global representation to estimate the value of the current state. The actor and critic are trained jointly using PPO [9].

#### IV. EXPERIMENTAL EVALUATION

We evaluate the proposed approach from three perspectives. First, we assess whether the learned policy can reduce monitoring volume and cost while preserving anomaly coverage. Second, we compare it with exhaustive, heuristic, and learning-based baselines under different anomaly densities and on real monitoring data. Third, we examine the contribution of the policy architecture and analyze how different reward configurations affect the learned behavior.

##### A. Experimental Setup

We consider a monitoring environment composed of nodes distributed across the Cloud-Edge-IoT continuum, where IoT nodes represent 30% of the system, edge nodes 40%, and cloud nodes the remaining 30%. The preferred monitoring capacity of each node is defined as a fraction of the number of available metrics: 20–40% for IoT nodes, 40–70% for edge nodes, and 70–90% for cloud nodes. Monitoring costs are also layer-dependent, with IoT metrics assigned the highest cost, followed by edge and cloud metrics. These settings follow the qualitative resource heterogeneity reported for Cloud-Edge-IoT systems, where endpoint and edge devices are resource-constrained relative to cloud data centers [1], and where telemetry collection and transmission dominate the energy budget of constrained devices [11], [25]. The literature therefore motivates the adopted ordering of resource constraints and monitoring costs across the IoT, edge, and cloud layers, while the specific layer proportions, capacity ranges, and cost values are chosen experimental parameters rather than measurements from a particular deployment.

We evaluate the proposed policy on two synthetic settings and one real-world monitoring dataset. The first synthetic setting contains  $N = 5$  nodes and  $M = 7$  metrics per node, and is used to study the behavior of the policy under different anomaly densities. The second synthetic setting contains  $N = 10$  nodes and  $M = 15$  metrics per node, generated with GutenTAG, an extensible tool integrated with TimeEval [23] and it is used for the reward ablation study under a 0.25 anomaly density. For the real-world evaluation, we use the Server Machine Dataset (SMD) [16]. The dataset contains monitoring data from  $N = 28$  machines, each described by  $M = 38$  metrics, including CPU load, memory usage, and network activity. This dataset is used primarily to evaluate

the scalability of the proposed policy and its behavior on real monitoring data. Layer assignments, monitoring capacities, and costs are therefore introduced synthetically using the same proportions as in the previous setting.

We compare our method against four baseline strategies: **Random-Selection** where the agent randomly selects metrics while respecting the preferred per-node budget, **Monitor-Everything** selects all available metrics at each monitoring window, **TopK-Anomaly-Causal** which is a non-learning baseline that, at each monitoring window, ranks the metrics of each node using a combined score based on the latest available anomaly and causal scores, and selects the top metrics up to the node’s preferred budget, and **RL-Flat-MLP** which flattens the complete monitoring state into a single vector and processes it using a standard MLP-Policy.

Experiments were executed on a server equipped with dual Intel Xeon Gold 6258R CPUs, 503 GB RAM, and an NVIDIA RTX A6000 GPU (48 GB VRAM) running Ubuntu 22.04.

To evaluate the performance of each approach we define the following metrics:

- **Anomaly Coverage Rate (ACR)**: Rate of anomalous metrics selected for monitoring.
- **False Monitoring Rate (FMR)**: Rate of selected metrics that are not anomalous.
- **Total-Coverage**: Defined as the harmonic mean between ACR and  $(1 - FMR)$ .
- **Budget Consumption Ratio (BCR)**: It measures how much of the available monitoring capacity is consumed.
- **Capacity Violation Rate (CVR)**: Rate of nodes where the capacity constraint is violated.
- **Cost**: Average monitoring cost per window, including collection, transmission, processing, and storage overhead.

The PPO agent is trained with a discount factor  $\gamma = 0.99$ , a learning rate of  $3 \times 10^{-4}$ , and a clipping range of 0.2. Episodes contain 512 or 1024 raw timesteps depending on the dataset size, and decisions are made over monitoring windows of size 10 timesteps. The best-performing model is selected through periodic evaluation during training. The size of feature vector  $F$  is set to 5 in all the experiments.

##### B. Performance Comparison

We first evaluate the proposed monitoring policy on the first synthetic dataset, resulting in 35 monitorable metrics. Table I reports the performance under two anomaly-density settings over 5 paired seeds, allowing us to assess the behavior of each method when anomalies are either sparse or more frequent.

As expected, *Monitor-Everything* achieves the highest ACR, since it collects all available metrics at every monitoring window. However, this exhaustive strategy comes at the cost of the highest budget consumption ratio, highest cost, and a large number of unnecessary metric selections, as reflected by its high FMR. In contrast, *Random-Selection* respects the monitoring budget but provides weak coverage, since it selects metrics without considering their importance. The *TopK-Anomaly-Causal* baseline improves over *Random-Selection* by prioritizing metrics with high anomaly or causal importance,

TABLE I  
PERFORMANCE COMPARISON UNDER DIFFERENT ANOMALY DENSITIES (MEAN  $\pm$  BETWEEN-SEED STD. OVER FIVE SEEDS).

| Anomaly-Density | Method              | ACR $\uparrow$          | FMR $\downarrow$        | Total-Cov $\uparrow$    | BCR              | Selected $\downarrow$ | CVR $\downarrow$        | Cost $\downarrow$       |
|-----------------|---------------------|-------------------------|-------------------------|-------------------------|------------------|-----------------------|-------------------------|-------------------------|
| 0.05            | Random-Selection    | 0.516 $\pm$ .008        | 0.884 $\pm$ .002        | 0.189 $\pm$ .003        | 1.000 $\pm$ .000 | 18.1 $\pm$ 0.1        | <b>0.000</b> $\pm$ .000 | 0.255 $\pm$ .002        |
|                 | Monitor-Everything  | <b>1.000</b> $\pm$ .000 | 0.883 $\pm$ .000        | 0.209 $\pm$ .000        | 1.935 $\pm$ .000 | 35.0 $\pm$ 0.0        | 1.000 $\pm$ .000        | 0.700 $\pm$ .000        |
|                 | TopK-Anomaly-Causal | 0.915 $\pm$ .001        | 0.794 $\pm$ .001        | 0.336 $\pm$ .001        | 1.000 $\pm$ .000 | 18.1 $\pm$ 0.1        | <b>0.000</b> $\pm$ .000 | 0.255 $\pm$ .002        |
|                 | RL-Flat-MLP         | 0.907 $\pm$ .006        | 0.741 $\pm$ .009        | 0.403 $\pm$ .011        | 0.784 $\pm$ .024 | 14.2 $\pm$ 0.4        | 0.176 $\pm$ .015        | 0.230 $\pm$ .006        |
|                 | <b>Ours</b>         | 0.936 $\pm$ .010        | <b>0.611</b> $\pm$ .075 | <b>0.543</b> $\pm$ .066 | 0.574 $\pm$ .093 | <b>10.4</b> $\pm$ 1.7 | 0.067 $\pm$ .018        | <b>0.146</b> $\pm$ .015 |
| 0.25            | Random-Selection    | 0.518 $\pm$ .003        | 0.533 $\pm$ .003        | 0.491 $\pm$ .003        | 1.000 $\pm$ .000 | 18.1 $\pm$ 0.1        | <b>0.000</b> $\pm$ .000 | 0.255 $\pm$ .002        |
|                 | Monitor-Everything  | <b>1.000</b> $\pm$ .000 | 0.533 $\pm$ .001        | 0.636 $\pm$ .001        | 1.935 $\pm$ .006 | 35.0 $\pm$ 0.0        | 1.000 $\pm$ .000        | 0.700 $\pm$ .000        |
|                 | TopK-Anomaly-Causal | 0.788 $\pm$ .001        | 0.296 $\pm$ .001        | 0.743 $\pm$ .001        | 1.000 $\pm$ .000 | 18.1 $\pm$ 0.1        | <b>0.000</b> $\pm$ .000 | 0.255 $\pm$ .002        |
|                 | RL-Flat-MLP         | 0.891 $\pm$ .004        | 0.309 $\pm$ .004        | 0.778 $\pm$ .003        | 1.162 $\pm$ .010 | 21.0 $\pm$ 0.1        | 0.504 $\pm$ .013        | 0.387 $\pm$ .007        |
|                 | <b>Ours</b>         | 0.861 $\pm$ .012        | <b>0.106</b> $\pm$ .034 | <b>0.877</b> $\pm$ .017 | 0.870 $\pm$ .038 | <b>15.0</b> $\pm$ 0.7 | 0.261 $\pm$ .019        | <b>0.254</b> $\pm$ .007 |

resulting in higher ACR and Total-Coverage. However, it always uses the full monitoring capacity and cannot reduce the monitoring volume when fewer metrics would be sufficient. This limitation is more clear in the sparse anomaly setting, where the method maintains a relatively high FMR.

Compared with these baselines, our proposed policy achieves the best overall trade-off between anomaly coverage, false monitoring, and monitoring cost. The trained policy learns not only which metrics are informative, but also when selecting additional metrics is not worth the monitoring cost. This is clearly shown by its lower FMR and higher Total-Cov in both anomaly-density settings. For instance, at anomaly density 0.05, our policy selects only 10 metrics on average while achieving the best Total-Cov score. Similarly, at anomaly density 0.25, it outperforms TopK-Anomaly-Causal in ACR, FMR, and Total-Cov, while selecting fewer metrics at a comparable monitoring cost.

The comparison with *RL-Flat-MLP* further highlights the benefit of the structured policy architecture. In both anomaly densities, our policy generally outperforms the *RL-Flat-MLP*, having lower FMR, higher Total-Coverage, fewer selected metrics, lower cost and lower capacity violation rate. When anomalies are more frequent, *RL-Flat-MLP* reaches a slightly higher ACR at the cost of all other performance metrics. This shows that preserving the node-metric structure leads to a better overall trade-off than processing the monitoring state as a single flat vector.

### C. Evaluation and Comparison with Real Dataset

We further evaluate the proposed approach on the SMD dataset to assess its behavior on real monitoring data and in a larger monitoring setting. Anomaly and causal scores are computed as described in III-C.

As shown in Fig. 2, the proposed policy achieves the best trade-off between coverage and monitoring cost. It reaches an ACR of 0.92 while selecting only 436 metrics on average, and thus reduces the cost by 19% compared with TopK-Anomaly-Causal and up to 68% compared to the Monitor-Everything baseline. More importantly, the structured policy clearly outperforms *RL-Flat-MLP* baseline where the last clearly shows degradation in performance when we scale,

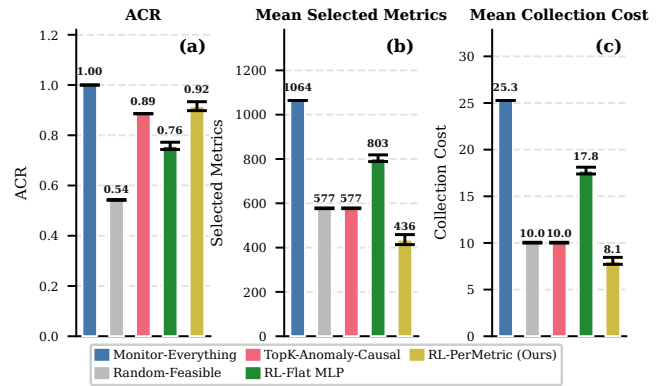


Fig. 2. Performance evaluation on the SMD dataset, comparing our approach with the baselines and an *RL-Flat-MLP* policy. Bars report the mean over 5 paired seeds and error bars denote  $\pm$  one between-seed standard deviation.

our approach obtains higher anomaly coverage (0.92 vs 0.76) while selecting 45.7% fewer metrics than *RL-Flat-MLP*. This suggests that explicitly encoding each node-metric pair and combining it with a global monitoring context is more effective than learning from a fully flat vector especially when we scale.

Overall, the SMD results show that the proposed policy remains effective on real monitoring data.

### D. Ablation Study of Reward Parameters

To investigate the impact of each reward term, we perform an ablation study by varying the reward weights using the second synthetic setting generated with GutenTAG. Table II summarizes the results.

When only the importance term is active ( $\alpha = 10.0$ ), the agent prioritizes anomalous and causally influential metrics, achieving near optimal anomaly coverage (0.98). However, without cost or capacity penalties, the policy selects a large number of metrics, resulting in the highest CVR and monitoring cost. This configuration illustrates that optimizing instantaneous importance alone leads to near exhaustive monitoring and highlights the necessity of cost and capacity regularization.

Adding the cost penalty ( $\lambda = 3.0$ ) substantially changes this behavior. The average number of selected metrics decreases

TABLE II

EFFECT OF REWARD WEIGHTS ON THE PERFORMANCE OF THE PROPOSED APPROACH SINGLE RUN PER EVALUATION.

| $\alpha$ | $\beta$ | $\lambda$ | $\delta$ | $\eta$ | ACR $\uparrow$ | Tot-Cov. $\uparrow$ | Selected $\downarrow$ | CVR $\downarrow$ | Cost $\downarrow$ |
|----------|---------|-----------|----------|--------|----------------|---------------------|-----------------------|------------------|-------------------|
| 10.0     | 0.0     | 0.0       | 0.0      | 0.0    | 0.983          | 0.493               | 140                   | 0.967            | 3.471             |
| 10.0     | 0.0     | 3.0       | 0.0      | 0.0    | 0.734          | 0.788               | 40                    | 0.136            | 0.884             |
| 10.0     | 3.0     | 0.0       | 0.0      | 0.0    | 0.578          | 0.688               | 31                    | 0.013            | 0.573             |
| 0.0      | 3.0     | 3.0       | 1.0      | 0.0    | 0.000          | 0.000               | 0                     | 0.000            | 0.000             |
| 10.0     | 1.0     | 0.1       | 0.1      | 2.0    | 0.827          | 0.799               | 50                    | 0.128            | 0.996             |

from 140 to 40, while the monitoring cost drops from 3.4705 to 0.8844. We can notice here that the CVR remains above zero, indicating that the cost penalty alone does not always keep the policy within the preferred node capacities.

When the capacity penalty is added instead ( $\beta = 3.0$ ), the policy becomes more conservative. The average number of selected metrics decreases to 31, and the CVR falls to 0.0132, but this comes at the expense of the ACR and Tot-Cov.

Finally, when the importance weight is set to zero, the agent prefers not to monitor any metrics, leading to negligible coverage. This behavior motivates the inclusion of the no-monitor penalty in the reward design to prevent degenerate solutions.

## V. CONCLUSIONS AND FUTURE WORK

This paper presented a RL framework for adaptive metric selection in Cloud-Edge-IoT continuum systems under heterogeneous monitoring constraints. The proposed policy processes node-metric pairs with shared parameters and combines their local features with a global monitoring context, enabling fine-grained selection decisions based on anomaly relevance, causal influence, monitoring cost, and node capacity.

The learned policy outperforms the baselines in balancing monitoring utility and resource use, reducing monitoring volume by 57% - 70% over exhaustive monitoring and 17% - 42% over capacity-aware methods while preserving strong anomaly coverage. The SMD results also show that maintaining the node-metric structure improves scalability, enabling the policy to select only metrics whose value justifies their cost.

Future work will investigate online updates of anomaly and causal indicators, generalization across systems with heterogeneous metric sets, and decision-making under partial observability when unselected metrics are unavailable.

## ACKNOWLEDGMENT

This work is partially funded by an ANR - French government grant under the France 2030 program, project SPIREC of PEPR Cloud (ANR-23-PECL-0006).

## REFERENCES

- [1] A. Al-Dulaimy, M. Jansen, B. Johansson, A. Trivedi, A. Iosup, M. Ashjaei, A. Galletta, D. Kimovski, R. Prodan, K. Tserpes *et al.*, "The computing continuum: From iot to the cloud," *Internet of Things*, vol. 27, p. 101272, 2024.
- [2] E. Fadda, P. Plebani, and M. Vitali, "Monitoring-aware optimal deployment for applications based on microservices," *IEEE Transactions on Services Computing*, vol. 14, no. 6, pp. 1849–1863, 2019.
- [3] D. Horovits, "Expensive Metrics: Why Your Monitoring Data and Bill Get Out Of Hand," Medium, Jan. 2023, accessed: 2026-05-11.
- [4] M. Andreolini, M. Colajanni, and R. Lancellotti, "Assessing the overhead and scalability of system monitors for large data centers," in *Proceedings of the First International Workshop on Cloud Computing Platforms*, 2011, pp. 1–7.
- [5] N. Yaseen, B. Arzani, K. Chintalapudi, V. Ranganathan, F. Frujeri, K. Hsieh, D. S. Berger, V. Liu, and S. Kandula, "Towards a cost vs. quality sweet spot for monitoring networks," in *Proceedings of the 20th ACM Workshop on Hot Topics in Networks*, 2021, pp. 38–44.
- [6] H. Mao, M. Alizadeh, I. Menache, and S. Kandula, "Resource management with deep reinforcement learning," in *Proceedings of the 15th ACM workshop on hot topics in networks*, 2016, pp. 50–56.
- [7] P. Cheng, Y. Chen, M. Ding, Z. Chen, S. Liu, and Y.-P. P. Chen, "Deep reinforcement learning for online resource allocation in iot networks: Technology, development, and future challenges," *IEEE Communications Magazine*, vol. 61, no. 6, pp. 111–117, 2023.
- [8] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola, "Deep sets," *Advances in neural information processing systems*, vol. 30, 2017.
- [9] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.
- [10] B. Rabenstein and J. Volz, "Prometheus: A {Next-Generation} monitoring system (talk)," 2015.
- [11] D. Trihinas, G. Pallis, and M. D. Dikaiakos, "Low-cost adaptive monitoring techniques for the internet of things," *IEEE Transactions on Services Computing*, vol. 14, no. 2, pp. 487–501, 2021.
- [12] D. Scheinert, B. S. Zadeh Aghdam, S. Becker, O. Kao, and L. Thamsen, "Probabilistic time series forecasting for adaptive monitoring in edge computing environments," in *2022 IEEE International Conference on Big Data (Big Data)*, 2022, pp. 4583–4588.
- [13] M. S. Frikha, A. Lahmadi, S. M. Gammar, and L. Andrey, "Leveraging reinforcement learning for adaptive monitoring of low-power iot networks," in *2020 16th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. IEEE, 2020, pp. 1–6.
- [14] R. Paffenroth, K. Kay, and L. Servi, "Robust pca for anomaly detection in cyber networks," *arXiv preprint arXiv:1801.01571*, 2018.
- [15] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 eighth IEEE international conference on data mining*. IEEE, 2008, pp. 413–422.
- [16] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, and D. Pei, "Robust anomaly detection for multivariate time series through stochastic recurrent neural network," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp. 2828–2837.
- [17] B. Schölkopf, R. C. Williamson, A. Smola, J. Shawe-Taylor, and J. Platt, "Support vector method for novelty detection," *Advances in neural information processing systems*, vol. 12, 1999.
- [18] J. Audibert, P. Michiardi, F. Guyard, S. Marti, and M. A. Zuluaga, "Usad: Unsupervised anomaly detection on multivariate time series," in *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, 2020, pp. 3395–3404.
- [19] J. Runge, "Discovering contemporaneous and lagged causal relations in autocorrelated nonlinear time series datasets," in *Conference on uncertainty in artificial intelligence*. Pmlr, 2020, pp. 1388–1397.
- [20] A. Tank, I. Covert, N. Foti, A. Shojai, and E. B. Fox, "Neural granger causality," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 8, pp. 4267–4279, 2021.
- [21] M. S. Frikha, S. M. Gammar, A. Lahmadi, and L. Andrey, "Reinforcement and deep reinforcement learning for wireless internet of things: A survey," *Computer Communications*, vol. 178, pp. 98–113, 2021.
- [22] Z. Xu, J. Tang, J. Meng, W. Zhang, Y. Wang, C. H. Liu, and D. Yang, "Experience-driven networking: A deep reinforcement learning based approach," in *IEEE INFOCOM 2018-IEEE conference on computer communications*. IEEE, 2018, pp. 1871–1879.
- [23] P. Wenig, S. Schmidl, and T. Papenbrock, "Timeeval: A benchmarking toolkit for time series anomaly detection algorithms," 2022.
- [24] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," pp. 652–660, 2017.
- [25] J. Krawiec, M. Wybraniak-Kujawa, I. Jacyna-Golda, P. Kotylak, A. Panek, R. Wojtachnik, and T. Siedlecka-Wójcikowska, "Energy footprint and reliability of iot communication protocols for remote sensor networks," *Sensors*, vol. 25, no. 19, p. 6042, 2025.