TU WIEN Informatics

Dynamic Power Management for Edge AI

A Sustainable Self-adaptive Approach

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieurin

in

Data Science

by

Julia Oberauner, BSc.
Registration Number 11816869

to the Faculty of Informatics

at the TU Wien

Advisor: Univ.Prof.in Mag.a rer.soc.oec. Dr.in rer.soc.oec. Ivona Brandic
Assistance: Alessandro Tundo, PhD.

Vienna, November 30, 2025

Julia Oberauner	Ivona Brandic

Declaration of Authorship

Julia Oberauner, BSc.

I hereby declare that I have written this Diploma Thesis independently, that I have completely specified the utilized sources and resources and that I have definitely marked all parts of the work - including tables, maps and figures - which belong to other works or to the internet, literally or extracted, by referencing the source as borrowed.
I further declare that I have used generative AI tools only as an aid, and that my own intellectual and creative efforts predominate in this work. In the appendix "Overview of Generative AI Tools Used" I have listed all generative AI tools that were used in the creation of this work, and indicated where in the work they were used. If whole passages of text were used without substantial changes, I have indicated the input (prompts) I formulated and the IT application used with its product name and version number/date.

Vienna, November 30, 2025

Julia Oberauner

Acknowledgements

I wish to first express my sincere thanks to netidee for their generous financial support of this thesis through a scholarship.

I owe deep gratitude to my supervisor, Alessandro, for his patience and guidance over the last year and a half.

To my friends: your support was essential. Thank you for collectively preventing me from dropping out of university and for your unmatched emotional control during this process.

Finally, to my parents: thank you for your unwavering belief and support. This accomplishment would not have been possible without you.

Abstract

The rapidly growing deployment of Edge AI devices performing high-demand tasks, such as real-time object detection, creates a critical challenge: balancing high performance (maintaining a target confidence) against the severe constraints of intermittent power supply from solar energy harvesting. This thesis addresses the necessity for a dynamic policy that can effectively manage this dual-objective trade-off over long operational horizons. The research establishes an empirical foundation via a parameter study conducted on Raspberry Pi hardware, quantifying the stochastic relationship between configuration parameters (model variant, resolution, frame rate) and actual power consumption/detection confidence, which revealed median shifts of up to 1.37W in power consumption and up to 26 percentage points in detection confidence between different operational configurations. This data informed the construction of a custom Reinforcement Learning (RL) environment that utilizes Kernel Density Estimation (KDE) to model hardware stochastically and physics-based models for solar dynamics. To solve the dual-objective problem of maximizing performance while satisfying the long-term survival goals, a Proximal Policy Optimization (PPO) agent was trained within a Constrained Optimization framework. The agent's policy was rigorously evaluated over 24-hour and 48-hour cycles across six dynamic scenarios against static and random baselines. The results confirm that the PPO agent successfully learned an adaptive strategy: it consistently manages the trade-off better than non-learning baselines, strategically scaling its resource use based on real-time energy context. Quantitative analysis showed that the PPO agent survived up to 1.5 hours longer than the more power-hungry baselines while achieving at least 40 percentage points more SLA satisfaction than the least power-hungry static policy. This work provides a validated, data-driven approach for sustainable resource management in energy-constrained Edge AI systems.

Contents

CHAPTER 1

Introduction

With the rapidly growing number of mobile and Internet-of-Things (IoT) devices, the concept of Edge Computing has gained significant importance. The basic idea is to move computation away from centralized locations like cloud data centers and closer to the user at the network edge [1, 2]. This results in a wide array of advantages, including lower latency, improved security, reduced network congestion, and lower overall costs [1]. Recent developments in machine learning and hardware have made it possible to deploy models on the network edge, a practice known as Edge AI, paving the way for real-time inference and automatic decision-making across industries from smart factories to the automotive sector [2, 3].

However, while there are many advantages, there are also major concerns, especially regarding sustainability. There are currently billions of edge devices deployed globally. Even though they are individually small and lightweight, their collective energy consumption is on par with large cloud data centers and cannot be ignored [4]. The number of these systems is only set to increase [5]. Since Edge AI systems are often deployed remotely and battery-operated, energy harvesting (deriving energy from external sources like solar, wind, or kinetic energy) from sustainable sources is crucial to power the devices and expand their battery life [6]. This introduces a core conflict: renewable energy sources are often unreliable, and the power generated by them varies significantly depending on outside conditions like weather or daytime [7]. This means the system must maximize long-term survival while simultaneously performing highly demanding tasks at a certain Quality of Service (QoS).

The general problem becomes clearer when applied to specific applications. Real-time object detection is one of the most common and computationally demanding tasks for Edge AI [8]. When this task is paired with solar power, one of the most common IoT energy solutions, it creates a direct contradiction between the system's requirement for high, continuous detection confidence and its intermittent power supply [9]. The system must therefore manage this energy conflict while continuously processing visual data,

ensuring it performs reliably (i.e., meets a target confidence) even when solar input is unavailable. This can be achieved by dynamically adjusting parameters like resolution or model type used at runtime depending on energy availability [10].

The heterogeneity of AI models, diverse hardware, and dynamic workloads mean simple rule and heuristic-based policies are often inaccurate and inefficient [11]. Alternatively, Reinforcement Learning (RL) presents a promising approach. It allows a system to learn optimal strategies through continuous interaction with its environment and adjust its behaviour based on real-time feedback [12]. *This thesis addresses the specific dual-objective problem within the constrained environment of a solar-powered edge device performing real-time object detection.* It aims to make an advancement for a very *specific and widely used hardware, the Raspberry Pi*, by building a prototype of an RL-based power management system and while doing so, answering the following research questions:

RQ1: *What is the influence of frame rate, image resolution, and (YOLOv11) model type on the power consumption and prediction confidence of a Raspberry Pi-based Edge AI system performing object detection?*

RQ2: *How does the implemented PPO-based dynamic configuration-switching prototype perform compared to static and random baseline strategies in maintaining inference confidence and energy efficiency under varying environmental conditions?*

The contributions of this thesis are summarized as follows: A *parameter study* was conducted on the target hardware to find the relation between YOLOv11 [13] model size, frame rate and image resolution and the power consumption and detection confidence of the system. This study quantified the operational range, revealing *median shifts of up to* 1.37 *W in power and up to* 26 *percentage points in confidence* between configurations. This work led to the development of a *custom RL environment*, built on the Gymnasium [14] framework, that incorporates physics-based solar modelling and utilizes data-driven stochastic models for power and confidence. Based on this, an *RL-based energy-aware power adaptation prototype* was designed using the PPO algorithm within a constrained optimization framework. Finally, a *comprehensive evaluation* was performed across multiple dynamic scenarios against four baselines to validate the policy's long-term sustainability. The evaluation confirmed the prototype's superiority, showing *the PPO agent survives up to* 1.5 *hours longer than power-hungry baselines while achieving at least* 40 *percentage points more SLA satisfaction* than the most conservative baseline.

The remainder of this thesis is structured as follows: *Chapter 2* provides the necessary theoretical *background* and reviews *related work*. *Chapter 3* details the *methodology*, including the parameter study, simulation environment design, and agent configuration. *Chapter 4* presents the *results of the evaluation*, followed by an analysis of the agent's adaptive performance. *Chapter 5* and *Chapter 6* conclude the thesis, summarizing the findings and outlining *limitations*.

CHAPTER 2

The Power-Performance Conflict and Prior Adaptive Solutions

This chapter provides the necessary *theoretical foundation and contextualizes this thesis within the existing literature*. The first part establishes the operational domain by defining Edge Computing and Real-Time Edge AI, alongside the constraints inherent to energy harvesting systems and the principles of Self-Adaptive Systems. The second part addresses the solution framework, detailing the Reinforcement Learning (RL) methodology and the Proximal Policy Optimization (PPO) method. The chapter concludes with a review of *related work*, identifying the specific research gaps addressed by this thesis's approach to energy-aware resource management.

2.1 Background

2.1.1 Edge Computing, Edge AI and Real-time Object Detection

Edge Computing describes a distributed computing paradigm where data processing and storage occur close to the source of data generation. The primary goal is to bring the services and utilities of Cloud Computing closer to the end user [1]. This results in clear advantages: lower latency, better security, reduced network congestion, and lower costs. However, edge devices operate under severe resource constraints, including limited computational power and restricted thermal envelopes [4]. This resource scarcity creates a fundamental conflict when executing complex tasks.

The utilization of AI on edge devices is defined as *Edge AI*, involving computations near the users on the network edge [3]. Due to the reduced latency inherent in edge deployment, real-time applications become feasible. One such application is *real-time object detection*, which requires the system to continuously localize, detect, and classify objects within a video stream using bounding boxes, labels, and confidence scores. This requires running

sophisticated deep learning models, such as *the YOLO family* (*You Only Look Once*) [13]. The core concept of YOLO models is that a single neural network, often using a Convolutional Neural Network (CNN) backbone, predicts both bounding boxes and class probabilities directly from the image in one evaluation pass, allowing the model to be optimized end-to-end for detection performance. This design makes the YOLO family specialized for extremely fast, high-throughput, real-time inference [15]. Running this high-demand inference task significantly intensifies the power constraint, as the device must process visual data quickly for timely decision-making.

2.1.2 Energy-aware Edge Computing

Energy-aware computing is highly important at the network edge because individual devices are severely resource-constrained, and collectively due to their large number they can contribute to energy consumption just as much as their cloud data center counterparts. At the edge, energy efficiency involves operations across the entire data life cycle, including data generation, processing, and transmission. This necessity means energy awareness must be addressed at every system layer. Solutions span the entire stack: on the hardware and architecture side, this includes implementing Dynamic Voltage Frequency Scaling (DVFS) and power capping, optimized cache management, and ensuring reconfigurable device designs for different scenarios. The operating system focuses on energy-aware resource management and scheduling. In the software and services layer, solutions range from application-specific data analytics to optimizing computation offloading policies, ensuring energy consumption is considered alongside metrics like latency and Quality of Service (QoS) [4].

To ensure continuous operation in autonomous edge deployments, devices often rely on *energy harvesting (EH) systems*. EH is the process of powering devices by capturing and converting small amounts of ambient environmental energy, such as solar, wind, or thermal, into usable electrical power. EH technologies suitable for edge and IoT applications include thermal, vibrational, radio frequency (RF), and Solar Photovoltaic (PV) systems [9]. Solar PV is particularly relevant for outdoor deployments due to its high power density and widespread availability [16]. However, many EH sources share the common challenge of providing intermittent and highly variable power supply, which necessitates sophisticated management [7].

This necessity makes Self-Adaptive Systems crucial. A Self-Adaptive System is designed to autonomously modify its own behaviour or structure at runtime in response to changes in its operating environment or internal goals. This capability is often conceptualized through the MAPE-K loop (Monitor, Analyse, Plan, Execute, and Knowledge), which describes the feedback control mechanism necessary for autonomous adaptation [17]. In Edge Computing, self-adaptation is vital for maintaining performance and resource efficiency under constantly varying conditions [10].

This line of reasoning directly applies to the hardware model of this work, the Raspberry Pi 4-class devices [18]. These are highly representative and widely used, low-cost IoT

hardware [19]. While highly capable, these devices introduce a critical power-processing conflict. Since they often run on batteries or intermittent energy sources, energy consumption must be meticulously managed to ensure continuous operation. This necessity for managing highly variable power while optimizing performance fundamentally motivates the policy developed in this thesis.

2.1.3 Reinforcement Learning and Proximal Policy Optimization

Reinforcement Learning (RL) is a machine learning paradigm concerned with how an *agent* should take actions in an *environment* to maximize a cumulative *reward* [12]. The interaction is modelled as a sequence of decisions governed by a *policy* (π) within the Markov Decision Process (MDP) framework. The agent learns the optimal policy by observing the resulting state transitions and rewards from its taken actions over time as well as considering possible future rewards. While doing this, the agent has to balance exploration of new strategies with the exploitation of actions that are known to yield a high reward [12].

The specific RL approach utilized is *Proximal Policy Optimization (PPO)*, which belongs to the class of *policy gradient methods*. With this kind of approach, instead of learning the values of actions and then selecting actions based on estimated values, the agent instead learns the parameter θ of a parametrized policy $\pi(a|s,\theta)$ (where a is the chosen action and s the state of the environment). Policy gradient methods seek to maximize performance based on the gradient of a scalar performance measure $J(\theta)$ with respect to the policy parameter. Their updates approximate gradient ascent in J:

$$\theta_{t+1} = \theta_t + \alpha\widehat{\nabla J}(\theta_t),$$

where $\widehat{\nabla J}(\theta) \in \mathbb{R}^d$ is a stochastic estimate that approximates the true gradient of the performance measure. Since they seek to maximize performance, their updates approximate gradient ascent in the performance measure [12].

The PPO algorithm is a state-of-the-art policy gradient method known for its stability and strong performance in complex control problems [20]. PPO offers the benefits of trust-region policy optimization but is significantly easier to implement. It achieves stability by employing a clipped objective function, which restricts the policy change at each step, preventing large, destabilizing updates. Furthermore, PPO is highly sample efficient. Unlike standard policy gradient methods that perform one gradient update per data sample, PPO's objective function allows for multiple epochs of minibatch updates per sample, extracting more signal from the gathered experience [20]. This makes PPO robust and effective in complex, non-linear environments.

2.2 Related Work

Research on energy-aware operation of Edge AI systems spans several connected themes. This section reviews (i) energy-harvesting and solar-powered edge systems, (ii) adaptive

energy-aware inference at the edge, (iii) reinforcement-learning-based energy and resource management, and (iv) optimization-based frameworks for energy efficiency.

2.2.1 Energy Harvesting and Solar-powered Edge Systems

Early work on energy-harvesting embedded systems established the foundations for *energy-neutral operation*. Hsu et al. [21] developed an adaptive duty-cycling algorithm that predicts future solar input to match consumption with harvested power, ensuring long-term sustainability of sensor nodes. Hörmann et al. [22] empirically evaluated indoor photovoltaic harvesting for IoT devices, showing strong dependence of harvestable power on light conditions and placement. Abas et al. [23] presented *SlugCam*, a solar-powered wireless smart-camera network, and characterized the power budget of video capture and transmission in outdoor environments. Collectively, these studies motivate accurate modelling of solar generation and battery storage in energy-aware system design, which provides the contextual basis for this thesis's simulation environment.

2.2.2 Energy-aware Edge AI and Adaptive Inference Configurations

In parallel, research on *adaptive inference* explores how configuration parameters influence energy and performance on constrained devices. Tundo et al. [10] proposed an energy-aware design-time methodology that helps developers identify pareto-optimal operating modes, which are combinations of frame rate, resolution, and model variant using a meta-heuristic search with weighted Grey Relational Analysis. These configurations can later support runtime adaptation and demonstrated energy reductions exceeding 80% with limited accuracy loss.

The *EcoMLS* framework [24] applied a runtime MAPE-K feedback loop to switch between lightweight and full neural models (e.g., YOLO variants) according to monitored accuracy and energy usage. Zhang et al. [6] optimized configuration selection and bandwidth allocation for multi-camera edge analytics via Lyapunov optimization, balancing latency and accuracy.

PowerPi [25] measured Raspberry Pi power draw under CPU and I/O workloads, showing a linear relationship between CPU utilization and power. *Unlike such utilization-based modelling, this thesis relies on direct power measurements from an object-detection workload including the Raspberry Pi and camera module, modelled via Gaussian Kernel Density Estimators (KDEs) to capture empirical power distributions across FPS, resolution, and model type.* These data-driven models form the foundation for simulation and adaptive control.

2.2.3 Reinforcement Learning for Energy Management and Edge Adaptation

Reinforcement Learning has recently become a key paradigm for optimizing energy use and scheduling under uncertainty. Prauzek et al. [26] applied Q-learning to manage the

duty cycle of solar-powered wireless sensor nodes, improving measurement throughput while preventing battery depletion. Hanif et al. [16] presented *The Solar AI Nexus*, a comprehensive review of RL applications in solar energy systems. They surveyed RL approaches for photovoltaic forecasting, energy-storage dispatch, and grid control, identifying challenges such as poor sample efficiency, computational overhead, and limited cross-environment generalization. The authors emphasized the need for lightweight, energy-aware RL algorithms deployable at the edge or in microgrids and highlighted opportunities for transfer and Meta-RL to enhance adaptability. These insights motivate the exploration of RL-based adaptive control within solar-powered Edge AI systems.

Beyond energy harvesting, RL has been widely applied for resource management in edge and cloud environments. Tuli et al. [11] combined an A3C agent with a residual RNN for stochastic edge-cloud scheduling, reducing energy use and SLA violations. Yang et al. [27] used a Deep Q-Network (DQN) to schedule workloads across energy-limited edge nodes, learning to balance job completion and energy consumption under dynamic conditions. The agent jointly optimized energy efficiency and throughput, outperforming heuristic schedulers even during power outages. Jayanetti et al. [28] extended this line of work to precedence-constrained workflows using a hierarchical PPO architecture that optimized both node selection and execution tier, achieving up to 56% energy savings and 46% faster execution than classical heuristics. Munir et al. [29] further incorporated risk-awareness through a multi-agent A3C approach, embedding Conditional Value-at-Risk (CVaR) into the reward to enhance reliability under renewable uncertainty. Together, these studies demonstrate how RL can autonomously learn energy-performance trade-offs across heterogeneous, dynamic systems, a principle this thesis adapts to configuration switching for solar-powered Edge AI.

Collectively, these studies validate RL's potential for multi-objective optimization in energy-constrained systems. However, most focus on task scheduling, voltage control, or grid optimization rather than on-device inference configuration. *This thesis addresses that gap by employing a PPO-based agent that dynamically switches model type, frame rate, and resolution on a solar-powered Edge AI camera, learning to balance inference confidence and energy autonomy across changing environmental conditions.*

2.2.4 Optimization-based Energy and Resource Management

Classical optimization methods also contribute to energy-efficient operation of IoT and edge systems. Wang et al. [30] proposed adaptive energy-saving algorithms that coordinate end-node and edge-server operation. While such formulations achieve provable energy-latency trade-offs, they rely on simplified analytical models and cannot easily capture the non-linear, context-dependent interactions among workload configuration, confidence, and harvested energy addressed in this thesis.

2.2.5 Summary

Prior research has addressed solar energy harvesting, adaptive inference, and RL- or optimization-based energy management, typically as separate problems. Few studies integrate measured energy characterization, renewable energy supply modelling, and learning-based configuration control in a unified framework. *This thesis bridges those areas by experimentally modelling power-performance relationships, incorporating solar and battery dynamics, and deploying a PPO-based controller that learns to sustain inference quality while maximizing energy autonomy in a solar-powered Edge AI system.*

CHAPTER 3

Data-driven Simulation and PPO Policy Formulation

3.1 Overview and General Approach

This chapter details the *methodology* used to answer the research questions defined in Section 1. Since the context of this work is a solar-powered edge device performing continuous object detection inference, the operational goal is to achieve a critical balance: the device must maximize its long-term survival while simultaneously utilizing available power to maintain high prediction performance and a defined target confidence level.

As a first step, an *empirical parameter study* is conducted, for which real power consumption and confidence data were collected in a lab setting. This study focuses on *three operational parameters* specific to the use case: frame rate (FPS), image resolution, and YOLO model variant. Statistical analysis of this data directly addresses the first research question and delivers valuable insights that are used in the subsequent design of the prototype of the *energy-aware RL agent*. This design process involves the creation of a custom data-driven *simulation environment* based on the Gym [14] framework, featuring specific modules for solar energy, battery dynamics, and probabilistic power and confidence modelling.

The chapter then outlines the selection and justification of the Proximal Policy Optimization (PPO) agent architecture and its specific training configuration and concludes by defining the *non-learning baselines*, the key *evaluation metrics*, and the comprehensive *evaluation procedure* used to test the agents' long-term sustainability across various dynamic scenarios.

3.2 Parameter Study

3.2.1 Data Collection and Experimental Setup

The investigation focused on three operational parameters that directly influence the edge device's performance-energy trade-off:

- **Model Variants:** Two different versions of YOLOv11 [13] (nano and small) were selected. YOLOv11 was chosen as a state-of-the-art model for real-time object detection that conveniently offers multiple sizes with inherent performance and power consumption trade-offs.
- **Resolution:** Four resolutions were tested: 128, 256, 320, and 640 pixels, ranging from very low to relatively high resolution.
- **Frame Rate (FPS):** Frame rates of 1, 3, 5, 15, and 30 FPS were investigated, covering a range from minimal to high speed operation. Higher FPS settings were tested initially but yielded negligible differences in power consumption as the system was already operating at full capacity.

The experiment was carried out in the HPC laboratory at TU Wien [31]. Since the intention was a full factorial experiment, power and confidence readings for all possible parameter combinations were required. The physical setup consisted of a Raspberry Pi 4 [18] device, chosen as a representative and widely used low-cost Edge IoT hardware model, connected to a Picamera2 [32] module and a digital power meter that could be queried remotely. The Raspberry Pi was pre-configured with the two YOLOv11 models and folders of images sourced from the COCO 2017 [33] dataset pre-scaled to the four available resolutions.

The experiment was managed by a workload generation and automation script. The procedure for each configuration was defined as follows: A remote command initiated the script on the Raspberry Pi, which loaded the specified YOLO model and instructed the Picamera to capture frames at the set FPS and resolution. As the captured frames from the lab environment were irrelevant, they were immediately discarded and substituted with a random image from the pre-scaled COCO 2017 folder. Inference was then run on the substituted image. A detection was generously defined as anything with a confidence larger than 0.25. Per image, the average confidence of all valid detections was calculated and written out. The workload ran for 90 seconds, during which power readings from the digital power meter and the calculated average confidences were continuously recorded. For robustness, the entire process was repeated 5 times.

3.2.2 Statistical Analysis

The statistical analysis served to validate the visual observations and quantify the influence of the three operational parameters on system performance and power consumption.

The raw data first required preparation. While the confidence data was ready for analysis, the raw power traces were cropped. Specifically, the first 15 and final 15 seconds of the 90-second traces were removed to ensure the data accurately represented power consumption during stable operation, discarding periods associated with system warm-up and cool-down.

Following data preparation, visual inspection was performed using boxplots (Figures 3.1 and 3.2) to assess the preliminary effects of model, resolution, and frame rate (FPS). This inspection focused on shifts in the central tendency (median) and spread (interquartile range, IQR) across the defined conditions.

The boxplots for *power consumption* (Figure 3.1) were faceted by all three parameters (model, FPS, and resolution), clearly illustrating shifts in the central tendency and variance across operating conditions:

- **FPS Effect:** Power consumption exhibits a strong positive correlation with FPS. The median power consumption for most conditions shifts upward from spanning approximately between 4.0W and 5.0W depending one the other parameters at fps = 1 to consistently above 5.0W at fps = 30. This increase is accompanied by a dramatic reduction in variability (tighter IQR) at higher FPS settings, suggesting a saturation point where power usage is dominated by the system's maximum operational frequency, regardless of configuration details.

- **Resolution Effect:** At low FPS (fps = 1 and fps = 3), resolution is a dominant factor. For the YOLOv11n model at fps = 1, the median power consumption spans a broad range, from approximately 3.8W (128 pixels) to $\sim$ 5.1W (640 pixels). This effect drastically diminishes at high FPS (e.g., fps = 30), where the distributions for all four resolutions become nearly identical, clustered around a median of $\sim$ 5.1 W.

- **Model Comparison:** The YOLOv11s model consistently shows a higher median power consumption than YOLOv11n at low and intermediate FPS. For example, at fps = 1 (256 pixels), the YOLOv11s median is $\sim$ 4.9W compared to YOLOv11n at $\sim$ 3.82W. Like the resolution effect, this difference vanishes at high FPS ($\geq$ 10), where the distributions for both models converge.

Confidence distributions (Figure 3.2) were only faceted by model and resolution, as visual inspection confirmed that the distributions did not change across varying FPS settings. This outcome is theoretically sound, as the requested input rate of the camera should not influence the detection confidence score of an individual image. The YOLOv11s model consistently outperformed YOLOv11n, achieving a higher central tendency across all resolutions. For YOLOv11n, the lowest 128-resolution setting exhibited extremely high variability and a low median ($\sim$ 0.40). Performance drastically improved at 256 pixels, and further increases to 320 and 640 resolutions showed only minor marginal gains, suggesting a performance plateau.

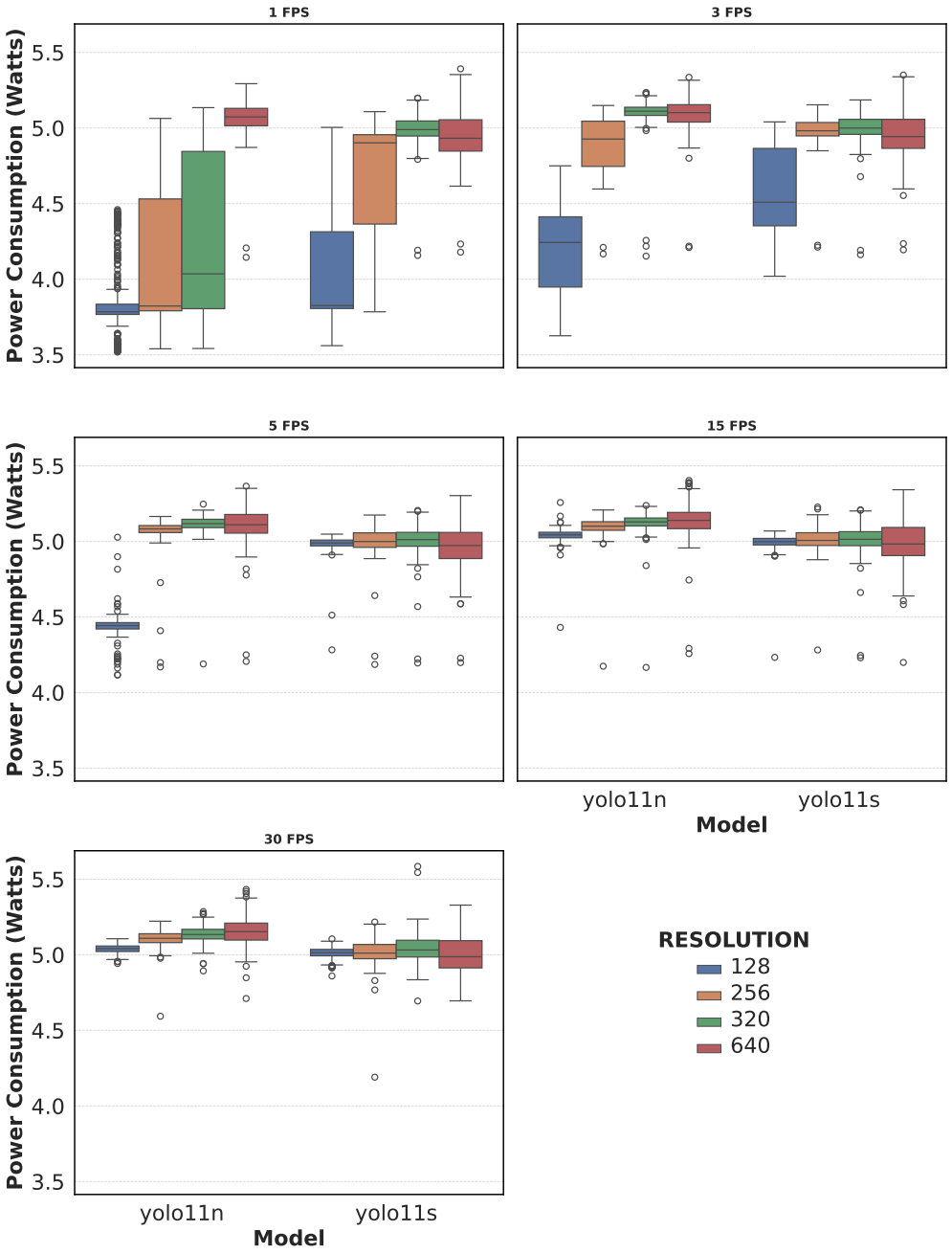


Figure 3.1: Power consumption distributions across FPS, resolution and model

To confirm whether the visually observed effects were statistically significant, proper statistical tests were carried out. Preliminary visual inspection of the boxplots showed substantial deviations from normality, including skewness, outliers and heteroscedasticity across groups. This was confirmed by Shapiro-Wilk tests, which returned p-values near

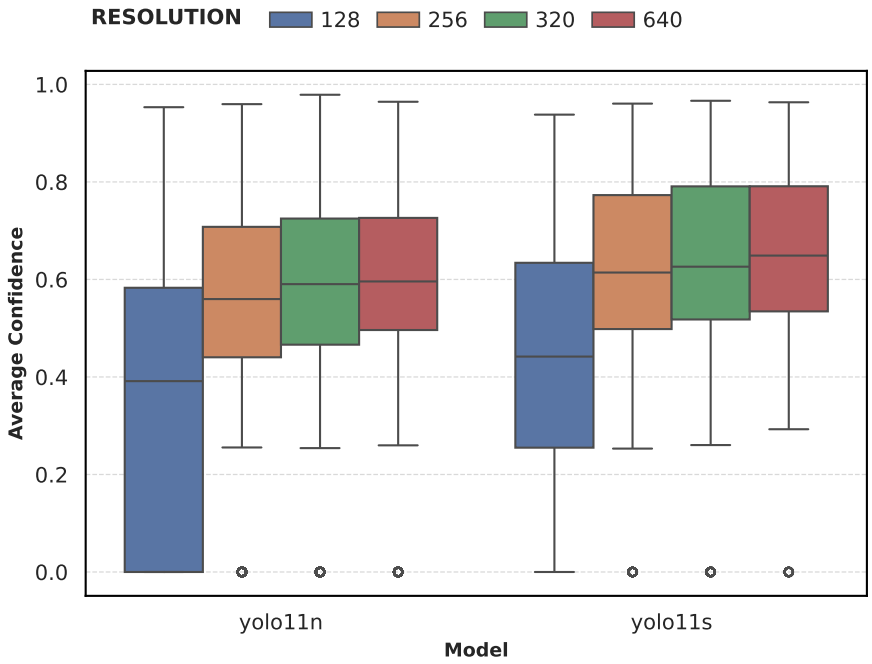


Figure 3.2: Distribution of average confidence per-image by resolution and model

zero for the majority of factor combinations, indicating clear violation of the normality assumption required for one-way ANOVA [34]. Given these violations, the Kruskal-Wallis H-test was selected as the appropriate non-parametric alternative to ANOVA [35, 36]. Following a significant omnibus result, pairwise post-hoc comparisons were conducted using Dunn's test, which is the standard rank-based method for multiple comparisons after a Kruskal-Wallis test [37, 35].

The Kruskal-Wallis H-test confirmed that for both power consumption and confidence, each relevant parameter had at least one level that was significant. This was followed up by Dunn's post-hoc tests with Bonferroni correction to localize the specific level differences. The results confirmed the quantitative trade-offs:

As can be seen in Table 3.1 **model architecture** exerts a highly significant impact on power consumption, with a corresponding statistically significant shift in detection confidence.

Table 3.1: Pairwise Dunn's Test results for model comparison

(a) Test results on power consumption

Model Pair	p-value	Significant
YOLO11n vs YOLO11s	5.81×10^{-183}	Yes

(b) Test results on detection confidence

Model Pair	p-value	Significant
YOLO11n vs YOLO11s	7.41×10^{-39}	Yes

Table 3.2 shows that *resolution* significantly affects both power and confidence, particularly between low and intermediate settings. The analysis confirmed a plateau effect in both metrics: power consumption and confidence increased significantly up to 320 pixels, but no significant difference was observed between the 320-pixel and 640-pixel resolutions.

Table 3.2: Pairwise Dunn's Test results for image resolution

(a) Test results on power consumption

Comparison	**p-value**	**Significant**
128 vs 256	9.69×10^{-184}	Yes
128 vs 320	≈ 0	Yes
128 vs 640	≈ 0	Yes
256 vs 320	8.18×10^{-51}	Yes
256 vs 640	7.85×10^{-66}	Yes
320 vs 640	0.251	No

(b) Test results on detection confidence

Comparison	**p-value**	**Significant**
128 vs 256	0.000000	Yes
128 vs 320	0.000000	Yes
128 vs 640	4.65×10^{-272}	Yes
256 vs 320	5.53×10^{-10}	Yes
256 vs 640	3.01×10^{-10}	Yes
320 vs 640	0.192	No

The analysis of the *frame rate* focused exclusively on power consumption, as previously justified. The The Dunn's test results shown in Table 3.3 indicated that frame rate significantly affects power, especially at low FPS settings. However, power usage was confirmed to plateau between 15 FPS and 30 FPS, validating the visual observation that the hardware system nears its full processing capacity regardless of further increases in the requested frame rate.

Table 3.3: Pairwise Dunn's Test results for frame rate on power consumption

Comparison	**p-value**	**Significant**
1 vs 3	1.40×10^{-106}	Yes
1 vs 5	2.25×10^{-243}	Yes
1 vs 15	≈ 0	Yes
1 vs 30	≈ 0	Yes
3 vs 5	1.75×10^{-29}	Yes
3 vs 15	2.69×10^{-128}	Yes
3 vs 30	3.13×10^{-162}	Yes
5 vs 15	2.70×10^{-35}	Yes
5 vs 30	7.91×10^{-54}	Yes
15 vs 30	2.47×10^{-2}	Yes (marginal)

3.2.3 Implications for the Prototype Design

The statistical analysis of the parameter study directly informed the design of the action space for the RL prototype. The objective was to select the minimum set of

configurations that offer the most effective trade-off between power consumption and confidence performance.

First, both YOLOv11 model variants (nano and small) were retained. The analysis proved they have a highly significant influence on both power consumption and confidence, establishing the fundamental operational range.

Second, the 128-pixel resolution was discarded. The confidence results for this setting were too low and exhibited high variability, meaning there is virtually no use case where that would be a good choice for the agent. The resolutions 256 and 320 pixels were retained because they deliver acceptable confidence scores while maintaining a statistically significant difference in power draw, offering a clear decision point for the agent. The 640-pixel resolution was also kept. Although the analysis confirmed a plateau effect in both power and confidence above 320 pixels, the boxplots suggest it may still be possible to squeeze out a marginal performance gain with only a slightly higher power cost, providing the agent with the option to attempt maximum performance.

Finally, the frame rate was fixed to 1 FPS, the lowest setting, and excluded as a variable parameter from the action space. This decision was based on two observations: FPS has no direct influence on detection confidence, and its primary effect is a negative influence on power consumption. Explicitly encouraging higher FPS would require implementing a direct incentive structure in the reward function, which is not beneficial for the defined goals of maximizing survival and achieving target confidence. Excluding FPS simplifies the learning problem and keeps the focus on the basic power-performance trade-off defined by the model and resolution.

Key Results RQ1

1. **Core Significance:** Confirmed *model variant and resolution are highly significant* factors influencing both power consumption and detection confidence.
2. **Power Magnitude & Plateau:** Demonstrated a *significant* $1.37\,W$ *difference in median power consumption* between operational extremes. A power plateau was confirmed between 320px and 640px.
3. **Confidence Range:** *Median detection confidence showed a substantial performance shift*, spanning from approximately 0.40 (the lowest 128px setting) up to a peak of 0.65 across the model and resolution configurations.
4. **Confidence Plateau:** A clear confidence plateau was observed, with no statistically significant gain found between the 320px and 640px resolutions.
5. **FPS Role:** FPS significantly im*pacts power consumption* but was confirmed to have *no significant influence on detection confidence.*

3.3 Simulation Environment

3.3.1 Environment Overview

With the empirical constraints and the final *action space* defined by the parameter study and its resulting implications, the next step involved translating these validated metrics into the operational environment. The following section details the design of the custom *simulation environment*, which utilizes the data and limits established here to accurately model the system's dynamics and facilitate agent training.

A *simulated environment* was developed to provide a feasible and reproducible platform for training and evaluating the agent. Real-world implementation, involving live solar panel data and continuous monitoring of an edge device in uncontrolled outdoor conditions, was deemed unrealistic and impractical for the scope of this work. The simulation addresses this limitation by creating a virtual learning environment where the agent can learn over extended, continuous periods of time.

The environment is built upon the Gymnasium [14] framework and features custom definitions for its *state and action spaces*, step logic, and *reward function*. It allows for the simulation of multiple continuous time episodes (for example, several days), during which it models four core components of the system: *power consumption, incoming solar energy, average prediction confidence, and the battery level* of the Raspberry Pi in an object detection scenario. The environment's setup allows for adjustable operational parameters such as the confidence threshold for SLA satisfaction tracking, time step length, and episode length.

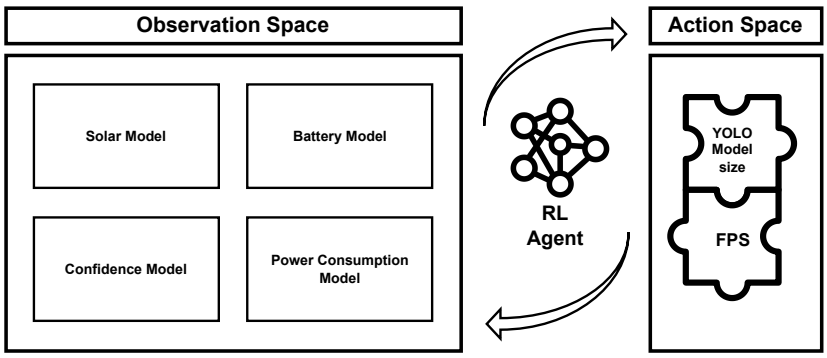


Figure 3.3: Architecture of prototype

The environment operates in two primary modes: a training mode, which uses randomized or continuous environmental conditions, and an evaluation mode, which makes it possible to run specific, predefined scenarios (e.g., specific seasons or weather conditions). The simulation architecture relies on specific, dedicated classes or modules responsible for modelling each of the four core components. The overall architecture model for the

simulation environment can be found in Figure 3.3. The subsequent subsections outline the design and logic of each component in detail.

3.3.2 State and Action Space

The environment is designed with distinct *state and action spaces* to facilitate the agent's learning process. The foundation for the definition of the *action space* has been established through the results and implications of the parameter study. Based on this, it is defined as a three-dimensional MultiDiscrete Gym [14] space defined by: model variant, resolution, and FPS. FPS has been added as a dimension for potential future experiments, but as discussed for the remainder of this work it is fixed to 1. The models available to the agent are theYOLOv11 "nano" (YOLOv11n) and "small" (YOLOv11s), and the resolutions are 256, 320, and 640 pixels.

The *state space* is initialized as a Gymnasium Box [14] space with four continuous dimensions, representing the system's observable metrics: battery level in percent (between 0 and 1), solar output in milliwatts (mW), average confidence (between 0 and 1), and power consumption in mW (between 0 and a ceiling of 6000 mW, reflecting the 6W capacity of the Raspberry Pi 4). Observations are obtained by querying the corresponding custom observation space functions.

3.3.3 Power Consumption and Confidence Modelling

The core of the simulation relies on the detailed modelling of power consumption and detection confidence, directly utilizing the empirical data obtained from the parameter study. For *power consumption*, the cropped power data from the experiments was grouped by FPS, resolution, and model type. A Gaussian KDE was then fitted to the power readings of each group, with the Gaussian distribution chosen based on visual validation via histograms. These KDEs are exported as pickle files. The dedicated Power() class is responsible for loading these KDEs and allowing the simulation to sample a realistic power consumption value corresponding to the agent's chosen action at each time step.

The *confidence modelling* is implemented using an identical methodology: the collected average confidence scores are grouped by resolution and model type (omitting FPS), and a Gaussian KDE is fitted to each group. The Confidence() class functions the same way as the power module, allowing the simulation to sample a realistic confidence score for the selected configuration.

3.3.4 Energy System Modelling

The energy system is composed of the *solar* and *battery* modules, which manage the inflow and outflow of energy. The *battery module* is an instance of the Battery() class, which simulates a specific battery configuration defined by its capacity (mAh) and voltage (V) at initialization. It features functions that charge and drain the battery linearly over a given time step duration using the simulated incoming solar power and outgoing power

consumption. It also provides functions to read the current battery level and check for battery depletion.

The *solar module* is responsible for simulating the energy inflow and is the most complex component. It simulates a solar panel with a specified maximum output (W) for a given geographical latitude, time of day, and day of the year, also accounting for cloud cover. The primary solar output is modelled using a cosine curve that peaks at solar noon. This output is scaled by two factors: a seasonal modifier and a cloud factor. The seasonal modifier is based on the solar elevation factor, which is approximated by $\cos\left(\frac{\pi}{2} - |\phi - \delta|\right)$, where δ is the solar declination on the specified day, and ϕ is the latitude in radians [38]. The solar declination itself is approximated by $23.45° \cdot \sin\left(2\pi - \frac{n-81}{2}\right)$, where $23.45°$ is the Earth's tilt and $n = 81$ corresponds to the equinox day [38]. The cloud factor modifies the output based on selected weather conditions. The SolarPanel() class allows for specific cloud modes: clear (cloud cover fixed at 0), cloudy (fixed at 0.9), and ramp (a dynamic cloudy day with intermittent sunny hours). If no cloud option is specified, the cloud cover is randomized between 0 and 1, based on a user-defined mean.

3.3.5 Episode Initialisation

The simulation environment uses the self.reset() function to initialize the state at the beginning of every episode. This function handles the environment's starting conditions based on whether specific options were set for the run. If specific scenario options (e.g., start day, hour, and cloud cover) are provided, the environment uses those values. Otherwise, the start day, start hour and cloud cover are randomized. After setting the start time, the function sets the global simulation time, resets the step counter, and resets the battery level to full.

3.4 Reinforcement Learning Setup

3.4.1 Reward Function

The reward function was engineered to capture the fundamental dual objectives of energy-harvesting inference: minimizing battery downtime and maximizing detection confidence whenever sufficient energy is available. The design combines elements of the Constrained Markov Decision Process (CMDP) with energy-aware design principles [39].

In line with the CMDP approach, detection confidence is treated as the primary objective (utility term), while energy consumption and battery downtime are treated as constraint costs. The main objective is quantified via the confidence margin to threshold (m_t). A form of potential-based shaping is applied to this margin [40, 41]. This reward or punishment is defined by the quadratic margin, meaning rewards grow quickly the further the confidence is above the threshold, but also, punishment becomes heavier more quickly the further the confidence falls below the threshold.

Energy inefficiency is penalized via a draw norm when the current power drawn by the device (P_t) is greater than the incoming solar power (S_t). This mechanism ensures that the agent is penalized for unsustainable operation.

When the battery is not empty and the system is operational, the final reward is a weighted sum of the confidence reward (R_{conf}) and the energy efficiency term (R_{eff}). The size of the weights is made dependent on the energy context derived from the current solar output and battery level [7]. When the energy context is high, the confidence reward (R_{conf}) is leveraged. Conversely, when the energy context is low, the weighting shifts, forcing the agent to prioritize minimizing the energy inefficiency term (R_{eff}). When the battery is empty, instead of the weighted sum, a simple per-step penalty is applied. Mathematically the reward function can be formulated as follows:

Let:

$$\begin{aligned} B_t &= \text{battery level at time } t, & S_t &= \text{solar power output (mw)}, \\ P_t &= \text{power demand (mw)}, & S_{\max} &= \text{maximum solar output (mw)}, \\ C_t &= \text{confidence estimate}, & C_{\text{th}} &= \text{confidence threshold}. \end{aligned}$$

The normalized solar contribution and energy context are:

$$s_t = \frac{S_t}{S_{\max} + 10^{-6}}, \qquad E_t = \tfrac{1}{2}(B_t + s_t).$$

The normalized net grid draw (inefficiency term) is:

$$D_t = \max\left(0, \frac{P_t - S_t}{S_{\max} + 10^{-6}}\right).$$

If the battery is depleted, a small constant penalty is applied:

$$R_t = -\kappa, \qquad \kappa = 0.3.$$

Otherwise, the confidence margin and its shaped reward are:

$$m_t = C_t - C_{\text{th}},$$
$$R_{\text{conf}} = \begin{cases} m_t^2, & m_t > 0, \\ -m_t^2, & m_t \le 0, \end{cases} \qquad R_{\text{eff}} = -D_t.$$

Energy-dependent weights adjust the reward emphasis:

$$\alpha_t = 1 + 2E_t, \qquad \beta_t = 1 - E_t,$$

leading to the final reward:

$$R_t = \alpha_t R_{\text{conf}} + \beta_t R_{\text{eff}}.$$

The per-step penalty $\kappa = 0.3$ was calibrated relative to the magnitude of "alive" rewards. Each environment step corresponds to 10 minutes, so $\kappa = 0.3$ implies a cumulative cost of approximately -1.8 per hour of depletion. During normal operation (when $C_t > C_{\text{th}}$ and sufficient energy is available), typical rewards lie around 0.08-0.10 per step ($\approx$ 0.5-0.6 per hour). Thus, one hour of battery downtime negates roughly three hours of successful operation, reflecting the design goal that maintaining uptime is more important than incremental efficiency gains.

3.4.2 Agent Architecture

The agent architecture utilizes the PPO algorithm. This selection was based on pragmatic reasons: StableBaselines3 [42] was chosen for the implementation of the learning agent, because it seamlessly integrates into the Gym environment, and out of its limited algorithms that were compatible with the state and action space structure of the custom environment, PPO proved to be the most suitable choice. The alternative, Advantage Actor-Critic (A2C), is recommended to be used with GPU resources, which are unavailable for this work [42].

Moreover, PPO is better suited for this control problem for several technical reasons. The reward function is complex, featuring multiple interacting and dynamically weighted terms. This interaction creates a *non-stationary reward landscape* where the scale shifts significantly depending on the real-time energy context. PPO's clipped objective function inherently handles these non-stationary rewards more robustly than standard policy gradient methods [43].

PPO is also known for its *sample efficiency*. It utilizes multiple epochs of minibatch updates per rollout, ensuring a higher signal extraction from the gathered experience. In environments characterized by slow temporal dynamics and potentially sparse reward signals, such as this simulation, this efficiency is critical, as A2C performs only a single gradient step per batch [20].

Finally, PPO offers greater *reliability and stability in complex*, dynamic environments. It constrains policy changes within a *trust region*, which minimizes large, destabilizing updates. This mechanism makes the algorithm less sensitive to the reward scale and hyperparameter settings, ensuring a reliable policy determination in an energy-aware system where multiple, conflicting objectives must be balanced simultaneously [43].

3.4.3 Training Procedure

The simulation environment was configured with specific parameters for agent training. The energy system utilized a 10,000 mAh battery operating at 3.7 V. The solar panel maximum output was set to 20 W, with the geographical latitude configured to simulate Vienna. The temporal settings included a timestep of 10 minutes and a fixed episode length of 48 hours. To ensure generalized learning across conditions, the start day and start hour of training episodes were randomized. The target performance requirement for the agent was defined by the SLA threshold, which was set to 0.6. This value was

Table 3.4: PPO training hyperparameters, configuration and rationale

Parameter	Value	Rationale
Total timesteps	1,000,000	Ensures sufficient exploration of the long-term state space.
Environment vectorization	`DummyVecEnv`	Standard setup for single-process environments.
Normalization	`VecNormalize`	Stabilizes training by normalizing observations and rewards.
Reward clipping	5.0	Prevents destabilizing gradient spikes from large reward values.
Policy architecture	`MlpPolicy`	Suitable for low-dimensional, structured input spaces.
Rollout length (`n_steps`)	`MAX_STEPS` (48h episode)	Captures full long-term dynamics across the entire episode.
Batch size	2048	Provides stable gradient updates.
Learning rate	3×10^{-4}	Balances convergence speed and stability.
Discount factor (γ)	0.997	Prioritizes long-term returns for multi-day processes.
Entropy coefficient (`ent_coef`)	0.01	Encourages exploration and prevents premature convergence.
Clipping range (`clip_range`)	0.15	Maintains PPO stability by limiting policy updates.
Checkpoint frequency	every 50,000 steps	Periodically preserves training progress.
Evaluation frequency	every 10,000 steps	Tracks performance and saves best model.

chosen based on the median of the empirical confidence distribution sampled by the KDE models derived from the parameter study. The exact parameters used for training the agent are shown in Table 3.4.

3.5 Evaluation Strategy

3.5.1 Evaluation Scenarios and Metrics

Evaluating the agent's strategy requires focusing on its core objectives: survival and maintaining the highest possible detection confidence. Therefore, the chosen measures of success must directly reflect these goals. The *key evaluation metrics* selected are: survival time, battery downtime, SLA satisfaction, and prediction performance/confidence [26, 21, 44]. Beyond quantitative results, the evaluation also aims to assess the agent's desirable, qualitative behavior, such as adaptive scaling.

To showcase how the agents and baselines perform under various conditions, a *set of specific evaluation scenarios* was defined. This approach, which tests adaptation strategies across different seasons and lighting conditions, is frequently used in related literature (loosely inspired by [45, 26, 22]). The selection was motivated by the need to include

diverse seasonal and weather conditions, a stress test, and dynamic environments that specifically test the agent's ability to scale up. The six final scenarios that were defined are summarized in Table 3.5:

Table 3.5: Defined evaluation scenarios

Scenario Name	Start Day	Start Hour	Clouds	Objective / Highlight
Summer_Clear	172	6	Clear	Optimal Conditions: Tests maximum performance under the longest day and highest solar irradiation.
Winter_Clear	355	6	Clear	Seasonal Test: Evaluates operation during short winter days with limited solar energy.
Winter_Cloudy	355	6	Overcast	Stress Test: Measures system endurance during consistently low light conditions.
Early_Morning	200	4	Clear	Dynamic Scaling: Evaluates the agent's ability to scale performance with increasing solar input after sunrise.
Cloud_Ramp	200	6	Ramp	Dynamic Adaptation: Tests response to fluctuating light levels during a partially cloudy day.
Night_Start	200	22	Clear	Survival Test: Examines overnight endurance until the next sunrise for recharge recovery.

3.5.2 Baselines

The performance of the trained PPO agent was evaluated against a set of *four non-learning baselines*: one random and three static policies. These baselines are designed to validate the effectiveness of the learned, adaptive strategy and to define the practical limits of the system's performance. The defined baselines are:

- **RANDOM:** This agent chooses a model variant and resolution randomly at every step. Its primary purpose is to serve as a minimal reference point, confirming that the PPO agent developed an actual strategy superior to arbitrary decision-making.
- **STATIC 1:** This policy is fixed to the least power-hungry settings possible: the YOLOv11 nano model (YOLOv11n) and the 256-pixel resolution. This baseline

represents the maximum achievable survival time and minimum power consumption within the environment's action space.

- **STATIC 3:** This policy is fixed to the most power-hungry settings: the YOLOv11 small model (YOLOv11s) and the 640-pixel resolution. This baseline establishes the upper bound for prediction performance and SLA satisfaction but represents the maximum power consumption and minimum survival time.

- **STATIC 2:** This policy is a combination of conservative and aggressive settings, fixed to the YOLOv11 small model (YOLOv11s) but using the mid-tier 320-pixel resolution. It serves as an intermediate benchmark for evaluating the trade-off space between the two extreme static policies.

The STATIC 1 and STATIC 3 baselines together *define the operational range of possibility* regarding power consumption and prediction performance in the simulated environment.

3.5.3 Evaluation Procedure

The agent and baselines were rigorously evaluated across the defined six scenarios. The procedure involved two simulation lengths: a 24-hour simulation and a 48-hour simulation, to assess both dynamic and long-term behaviours. For statistical robustness, 50 evaluation episodes were run for every unique combination of agent, baseline, scenario, and simulation length. Apart from setting the specific scenarios and the episode length, the evaluation environment was configured in the exact same way as the training environment. From these 50 evaluation episodes per configuration, both aggregated results (average metrics) and average traces (battery and confidence over time) were extracted for discussion in the following chapter.

3.5.4 Summary

This chapter detailed the methodology used to transition from empirical hardware measurements to a validated adaptive policy.

The process began with a *parameter study* that quantified the power and performance trade-offs on the Raspberry Pi across model variant, resolution, and FPS. *Statistical analysis* confirmed that model and resolution were the highly significant operational factors, while FPS was subsequently fixed to 1 for the RL agent's action space.

These insights were used to design the *custom simulation environment*. The environment utilizes Gaussian KDEs to model the stochastic power and confidence based on real-world data, coupled with validated physical models for solar and battery dynamics. The agent architecture was based on the stable and sample-efficient PPO algorithm, designed around a Constrained Optimization framework to handle the dual objectives of maintaining high detection confidence and long-term survival.

Finally, the *evaluation strategy* was defined. This strategy outlined the necessity of testing the PPO agent against *four non-learning baselines across six challenging, dynamic scenarios*, with all configurations assessed over 24-hour 48-hour episodes and 50 total runs per combination to establish the foundation for robust statistical validation in the subsequent chapter.

CHAPTER 4

Analysis of Adaptive Behaviour and System Efficacy

4.1 Overview and Structure

This chapter addresses RQ2 by presenting the *qualitative and quantitative results* of the evaluation strategy and scenarios discussed in the previous chapter. It begins by detailing the aggregated evaluation measures, followed by an inspection of the trade-off between two key factors: *SLA satisfaction and battery downtime*. Next, the agents' *temporal behaviour* is analysed using 24-hour traces of battery and confidence levels. Finally, the chapter concludes by examining the agents' *long-term behaviour* through 48-hour results.

The evaluation was executed as follows: To obtain the aggregated results presented in the tables and bar charts, each agent was run in every scenario for 50 separate episodes. The mean value across these episodes was then calculated. Trace data was collected similarly. Over the 50 episodes, the agent's battery level and confidence were recorded at every 10-minute time step. The average value per time step was then calculated and displayed as the final average trace.

4.2 Aggregate 24h Performance

This section assesses the overall performance of the agents across all evaluated 24-hour scenarios. The analysis is based on a set of six key performance measures: two focused on object detection performance, three on energy management, and the overall reward. The two main performance metrics discussed, SLA satisfaction and battery downtime, are representative of the overall trends; the other energy conservation measures behave very similarly to dead battery hours, and average confidence behaves similarly to SLA satisfaction.

The specific measures considered include the *average SLA percent* (the average percentage of total time steps where the agent's confidence exceeded the predefined threshold) and *average confidence* for detection performance. Energy management is evaluated using the *average time until first battery death, average total time with dead battery (h), and average total time alive (h).* The final measure is the *average reward.* An example table presenting all six measures for a single representative scenario is provided below in Table 11. For the complete set of aggregated results across all scenarios, the reader is directed to the Appendix. (Note that all time increments are based on the 10-minute timestep of the simulation.)

Table 4.1: Performance metrics for Night_Start scenario

Metric	ppo	random	static1	static2	static3
Avg_Time_Till_First_Death_h	8.52	7.99	24.00	7.50	7.51
Avg_Empty_h	0.66	1.30	0.00	1.83	1.83
Avg_Time_Alive_h	23.34	22.70	24.00	22.17	22.17
Avg_SLA_%	62.21	58.82	0.01	99.99	100.00
Avg_Confidence	0.617	0.607	0.562	0.631	0.648
Avg_Reward	-0.087	-0.101	-0.081	-0.109	-0.106

To illustrate the overall agent behavior and the inherent conflict in objectives, the analysis focuses on *two representative measures: SLA satisfaction and battery downtime.* These are shown in Figures 4.1 and 4.2, respectively. The overall behavior exhibits a clear power-performance trade-off. The power-hungry baselines, STATIC 2 and STATIC 3, consistently outperform all other agents in performance- and SLA-focused measures, while the least power-hungry baseline, STATIC 1, performs the worst. For energy conservation measures, the trend is inverted, with STATIC 1 performing the best.

This trade-off is clearly visible in the figures. Looking at SLA performance (Figure 4.1), STATIC 2 and 3 achieve consistently high SLA percentages, averaging between 99% and 100%. In contrast, STATIC 1 barely exceeds 0% in most scenarios. The PPO agent delivers an average SLA of 45% to 72%, outperforming the RANDOM baseline in every scenario except the low-energy Winter_Cloudy and Winter_Clear environments. While this drop in performance may seem alarming, it reflects the agent's training objective: the PPO agent is explicitly trained to prioritize survival in low-energy contexts (like winter) and maximize performance only in high-energy contexts.

The battery downtime data (Figure 4.2) confirms the PPO agent's conservative strategy. PPO consistently achieves less dead battery time than the RANDOM baseline (and STATIC 2/3), averaging approximately one hour less than the power-hungry static agents. This gap is largest in the two winter scenarios, where PPO reduces downtime by about 1.5 hours. The RANDOM baseline also shows less dead battery time than STATIC 2/3, with a reduction of 19 to 38 minutes on average. This is an expected result, as the RANDOM agent selects the smaller, less power-hungry model with a 50% probability and

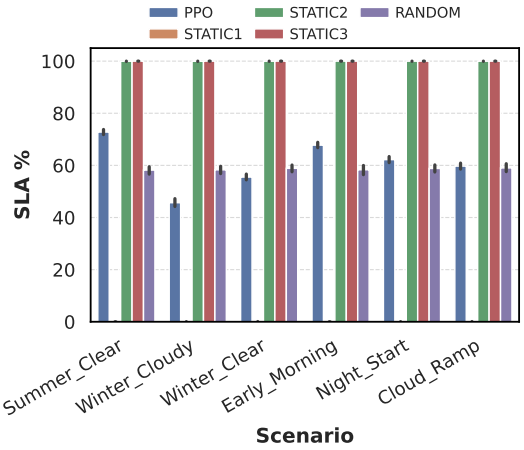


Figure 4.1: Average SLA across agents and scenarios.

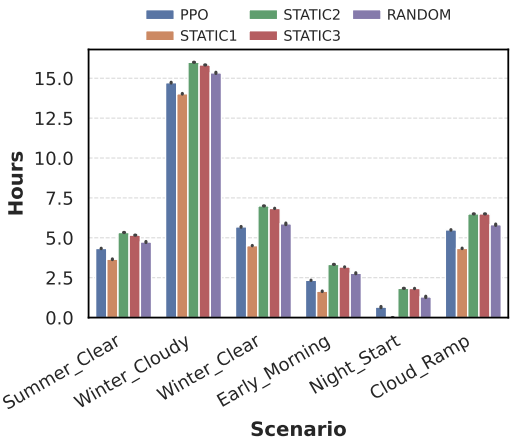


Figure 4.2: Average empty hours across agents and scenarios.

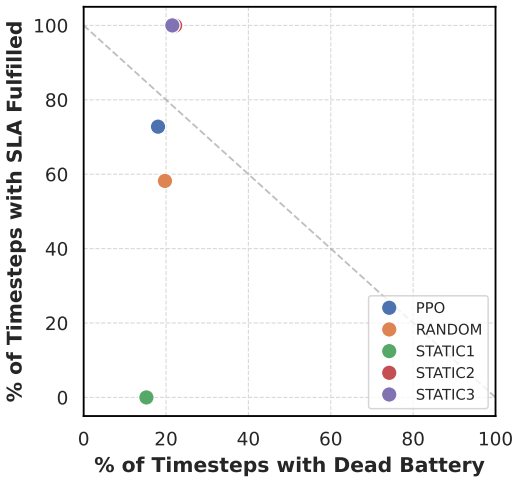
the smallest resolution with a $\frac{1}{3}$ probability, giving it an inherent, albeit non-strategic, chance of saving energy. This aggregated data already highlights the fundamental SLA vs. battery downtime trade-off, which is explored in greater detail in the subsequent section.

4.3 Trade-off Analysis

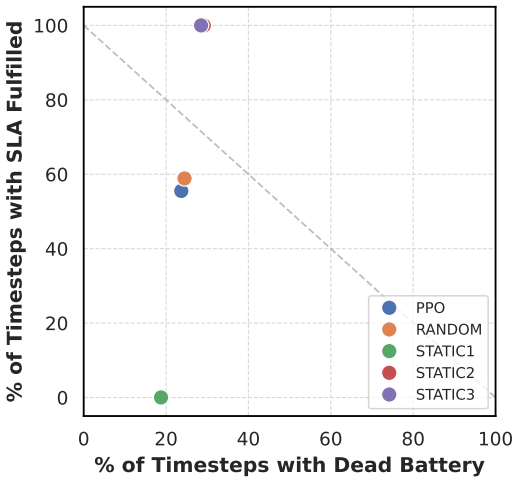
The nature of the optimization problem means that there is an inherent trade-off between maintaining high confidence and SLA satisfaction and minimizing battery downtime. This relationship is explored in detail in Figure 4.3, which directly plots the average percentage of the 24-hour episode spent with a dead battery against the average SLA percentage fulfilled. This visualization utilizes the percentage of total time steps spent with a dead battery (instead of the absolute number of hours) to provide a more intuitive scatter plot comparison against the SLA percentage.

In the figure, the percentage of dead battery time is mapped to the x-axis, and the percentage of fulfilled SLA is mapped to the y-axis. This means, agents positioned further left are more energy conservative, while agents further up are more performance and confidence focused. The dotted grey line represents the theoretical case where the SLA percentage is equal to the percentage of time the battery is alive (SLA = 1 − Dead Battery %). Points above this line indicate a stronger focus on SLA, while those below suggest a greater prioritization of energy conservation.

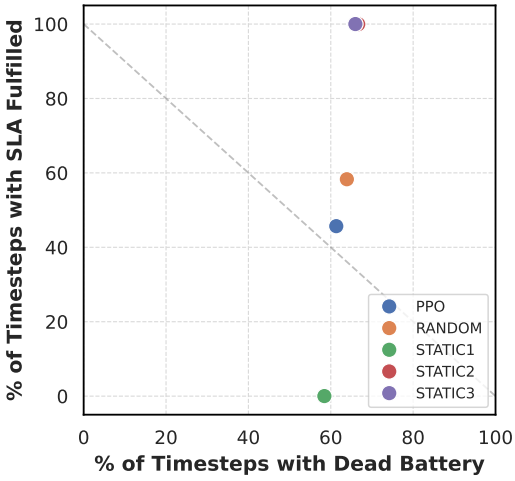
The scatter plots reveal several patterns. First, the overall difference in dead battery percentage between the most and least power-hungry agents is not very large, which provides important context for the 20 minutes to 1.5 hours difference previously seen in the aggregated results. As expected, the STATIC 1 baseline consistently sits in the



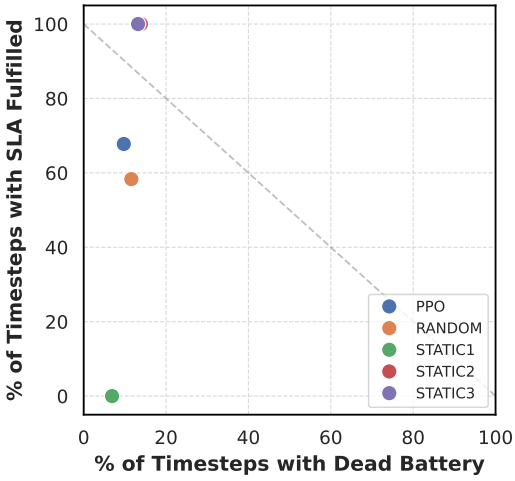
(a) Summer_Clear



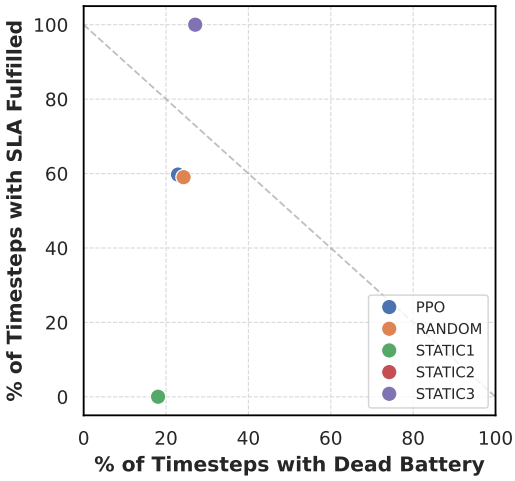
(b) Winter_Clear



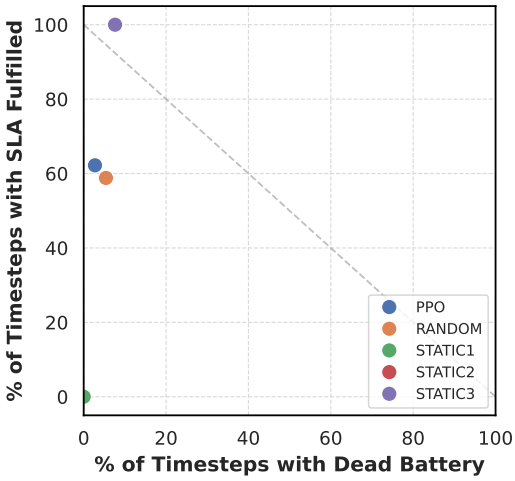
(c) Winter_Cloudy



(d) Early_Morning



(e) Cloud_Ramp



(f) Night_Start

Figure 4.3: Trade-off between SLA fulfillment and battery survival by scenario

bottom-left corner of every plot, achieving essentially no SLA satisfaction while having the least battery downtime. In contrast, the STATIC 2 and STATIC 3 baselines are positioned at the top and farthest right out of all agents with almost perfect SLA satisfaction at the cost of the highest battery downtime. The RANDOM baseline maintains a relatively stable average SLA around 60 percent across scenarios as well consistently slightly more dead time steps than the PPO agent.

The PPO agent's position varies, showing its adaptability. In *high-energy scenarios*, most noticeably Summer_Clear and Early_Morning, the PPO agent overtakes the RANDOM baseline in SLA. This hints at the agent's ability to successfully scale up its performance when energy is readily available, a concept explored further in the next section. In *low-energy contexts* (night, winter, or cloudy), PPO shifts to leverage survival more. This survival-first behavior is intentional and directed, which is what differentiates it from the undirected RANDOM baseline.

4.4 Temporal Behavior (24h)

Beyond the high-level quantitative performance metrics, it is crucial to confirm that the trained agent reacts to changing conditions as intended. The desired behavior is that the agent should scale up its object detection performance when solar energy is abundant and scale down when lighting conditions are poor, such as during cloudy periods or darkness.

The figures in this section illustrate the agents' qualitative behavior throughout a 24-hour period. Two representative scenarios were chosen to best demonstrate this dynamic behavior: Early_Morning and Cloud_Ramp. Both are dynamic scenarios that feature significant shifts in lighting conditions: from night to full day, and from sunny to cloudy periods, making them ideal for judging the agent's learned responsiveness.

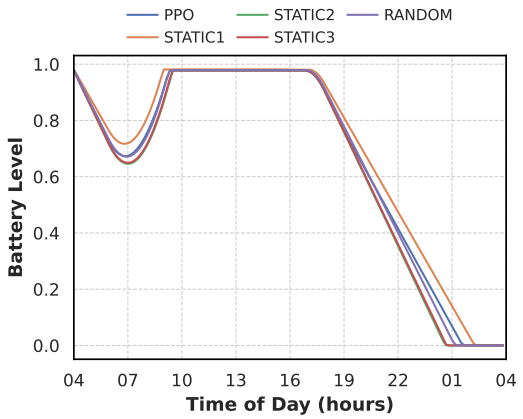


Figure 4.4: 24h battery traces for Early_Morning.

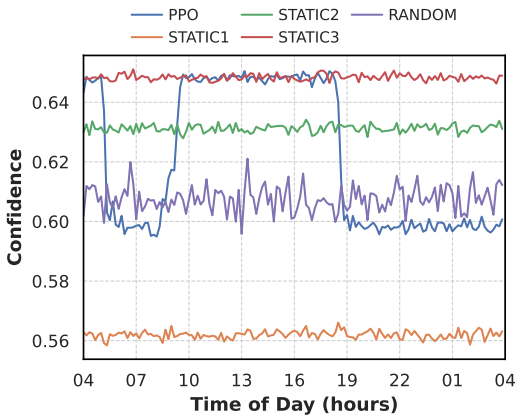


Figure 4.5: 24h confidence traces for Early_Morning.

Figures 4.4 and 4.5 show the average battery and confidence traces for the Early_Morning

scenario. This scenario is set on a summer sunny day, but begins at 4:00 AM, with the sun rising around 6:00 AM and setting at 6:00 PM. In the battery trace (Figure 4.4), the initial period shows a dip as all agents draw power during the dark hours. Battery recovery begins around 7:00 AM and remains near full until sundown. All agents show similar general behavior, though the PPO agent exhibits the least deep initial dip and achieves the longest survival time (after the energy-conservative STATIC 1). The confidence traces (Figure 4.5) offer the most interesting insight into the learned behaviour. The static baselines naturally remain at their constant confidence levels (STATIC 3 around 0.65, STATIC 2 around 0.63, and STATIC 1 around 0.56). The RANDOM agent fluctuates broadly around 0.60 - 0.62. The PPO agent, however, starts on par with STATIC 3 but quickly dips to around 0.59 as it registers the lack of sunlight. Then, around 8:30 AM, the agent successfully scales up its performance to match the level of STATIC 3 and maintains this high confidence until sundown. This trace confirms the basic learned behavior: prioritizing energy during darkness and maximizing performance when energy is available.

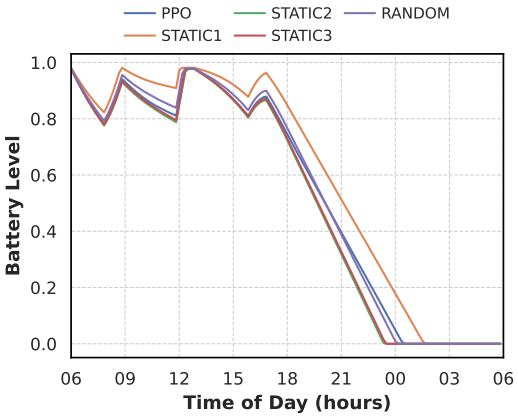


Figure 4.6: 24h battery traces for Cloud_Ramp.

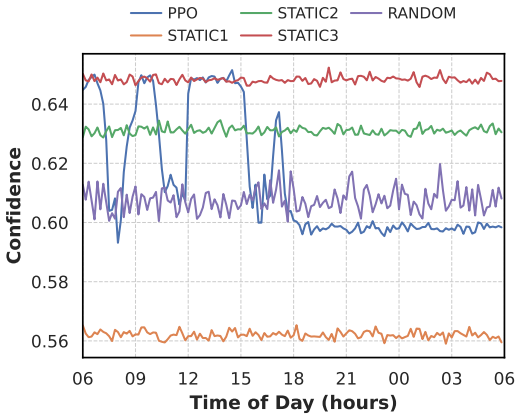


Figure 4.7: 24h confidence traces for Cloud_Ramp.

Figures 4.6 and 4.7 show the average traces for the Cloud_Ramp scenario. This scenario simulates a dynamic environment characterized by a cloudy day broken up by regular periods of sunshine (four hours of clouds followed by one hour of sun, repeated). The effect of these solar changes is clearly visible in the dips and spikes of the battery trace (Figure 4.6). Similar to the previous scenario, only STATIC 1 manages to outlast PPO and recover faster from cloudy periods. While the RANDOM agent appears close in this trace, its non-strategic nature is exposed when examining the confidence levels. In the confidence traces (Figure 4.7), the PPO agent again starts at the level of STATIC 3 before dipping in response to the initial cloudy period. It then consistently comes back up in the periods where sunlight becomes available (notably around 9:00 AM, 12:00 PM, and a final, less pronounced scale-up late in the day). This demonstrates the agent's ability to react to short-term energy flux by dynamically adjusting its resource commitment. T*his*

confirms that the PPO agent has learned the necessary qualitative behavior to manage resources efficiently in a dynamic environment.

The dynamic scaling behavior observed in the traces provides additional explanation for the PPO agent's results in the aggregated figures and trade-off visualization. The ability to scale up only when energy is abundant and scale down to conserve power directly leads to the PPO agent' position on the trade-off curve.

4.5 Long-Term Evaluation (48h)

The analysis of agent behavior is concluded by *examining the 48-hour results.* As the PPO agent was specifically trained on 48-hour episodes, this sustained duration represents the primary validation of its resource management strategy. The expectation is that the adaptive, conservative decisions observed in the 24-hour analysis should compound, resulting in significantly improved long-term sustainability compared to static or random baselines. To evaluate this long-term energy pacing and conservative behavior, a scenario with low but stable lighting conditions, Winter_Clear, is selected for illustration in Figures 4.8 and 4.9.

The 48-hour battery trace (Figure 4.8) confirms the compounding benefit. During the initial 24 hours, the PPO agent's performance might appear unremarkable, lasting only slightly longer than the RANDOM baseline. *However, in the second half of the cycle, the cumulative effect of PPO's energy-aware actions becomes clear: it manages to start recharging earlier and achieves a higher maximum charge level than STATIC 2, STATIC 3, and RANDOM.* Furthermore, in both the peak battery charge and overall survival time on Day 2, PPO significantly pulls away from the RANDOM baseline and nearly matches the longevity of the most conservative agent, STATIC 1.

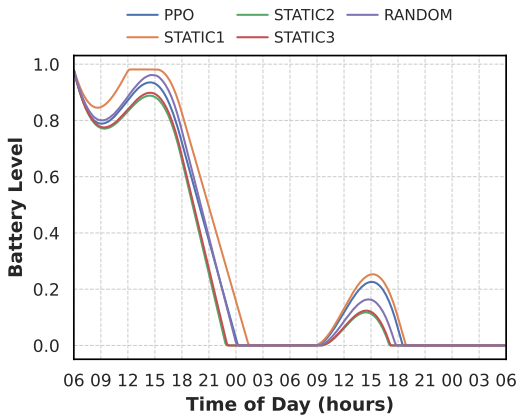


Figure 4.8: 48h battery traces for Winter_Clear.

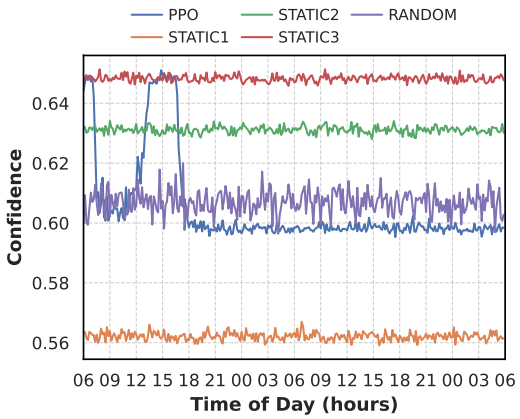


Figure 4.9: 48h confidence traces for Winter_Clear.

The corresponding 48-hour confidence trace (Figure 4.9) demonstrates the PPO agent's

strict adherence to its survival objective in this extreme environment. After sundown on the first day, the agent never chooses to scale up again. It correctly remains at its baseline energy conservation settings, achieving approximately 0.59 confidence. This behavior is expected because the agent starts Day two with a low battery, and the limited solar input in this winter scenario is insufficient to sustain high performance (even the survival benchmark, STATIC 1, only reaches a maximum of about 30 percent charge on Day 2). The agent correctly identifies this extra-low energy context and prioritizes survival over performance, confirming the success of its learned, adaptive strategy. This 48-hour analysis confirms that the PPO agent's ability to compound small, correct energy decisions across sustained operational cycles is the key to its success, providing the necessary sustainable management required for persistent operation.

4.6 Summary of Results

This chapter presented the *quantitative and qualitative results* of the agent evaluation, demonstrating the performance of the trained PPO agent against various baselines.

The aggregated 24-hour results established the core performance-energy trade-off: Static baselines excelled in either SLA satisfaction or energy conservation, *while the PPO agent achieved a balanced, competitive performance.* This trade-off was further visualized in the scatterplots, which confirmed that the PPO agent successfully positioned itself away from the extreme inefficiency of the RANDOM baseline and closer to the optimal region of the trade-off curve. Specifically, PPO demonstrated the ability to scale up performance in high-energy scenarios (e.g., Summer_Clear) while prioritizing survival in low-energy contexts.

The temporal analysis confirmed how the agent operates. Traces from dynamic scenarios like Early_Morning and Cloud_Ramp *proved that the agent learned the intended behavior*: dynamically scaling up performance when solar energy was available and scaling down to conserve power when conditions were poor.

Finally, the 48-hour long-term analysis validated the agent's sustainable resource management. By *demonstrating the compounding benefit of its conservative strategy*, particularly in low-energy environments like Winter_Clear, the PPO agent was shown to significantly increase its second-day survival and battery charge relative to the RANDOM baseline. This ultimately confirms that the learned, adaptive approach provides the necessary longevity required for persistent operation.

Key Results RQ2

1. **Optimal Trade-off Policy:** The PPO agent achieved a *balanced performance on the survival-vs-confidence trade-off*, surpassing the non-learning baselines.
2. **Dynamic Adaptation:** The policy *successfully learned to scale performance* based on real-time energy availability.

3. **Goal Prioritization:** The agent demonstrated *contextual decision-making*, prioritizing long-term survival in low-energy scenarios and maximizing performance in high-energy scenarios resulting in up to 1.5 hours less battery downtime.

4. **Sustainable Longevity:** The 48-hour analysis validated the strategy's *compounding benefit*, confirming the necessary energy conservation for persistent multi-day operation

CHAPTER 5

Discussion and Limitations

This chapter discusses the *limitations* encountered during the system design and evaluation, and outlines avenues for future research to extend the applicability and robustness of the adaptive agent.

5.1 Limitations of the Current Prototype

The results presented, while validating the PPO agent's strategy, are subject to several inherent limitations related to the simulation environment and agent complexity.

5.1.1 Environment Simplification and Stochasticity

The current simulation environment represents a simplified version of the real world. Although the power and confidence models are derived from empirical data, the environment's overall stochasticity is limited, especially when specific weather scenarios are fixed for evaluation. This simplification means that while the agent learns the intended reactive behavior (scale up in light, scale down in dark), this behavior might appear overly deterministic, potentially suggesting that a simple rule-based system could suffice. This is likely a result of the environment's inherent predictability rather than a limitation of the policy learning capacity.

5.1.2 Scope and Generalizability

The agent was trained and evaluated under environmental conditions specific to the latitude of Vienna, and its solar modelling is parameterized accordingly. Consequently, the agent's policy may not generalize effectively to different geographic locations with very different seasonal solar profiles or circadian cycles without specific fine-tuning. Furthermore, the model does not account for the additional energy overhead consumed

by the power management system itself, which in a real-world scenario would introduce a constant, irreducible power floor.

5.1.3 Baselines

The agent's performance was compared only against static and random baselines. A more comprehensive evaluation would typically include a second, distinct Reinforcement Learning agent (e.g., A2C or DQN) or a complex rule-based baseline that attempts to mirror the adaptive logic (e.g., a policy that switches based purely on battery charge thresholds). The absence of these comparison agents restricts the ability to definitively quantify the performance of the PPO algorithm's learned policy over alternative heuristic strategies.

5.2 Future work

The following research directions are proposed to enhance the system's realism, robustness, and generality.

5.2.1 Enhancing Environment Realism

Future work should focus on integrating more real-world elements and improving stochasticity. This could involve using a live Solar Output API for highly realistic weather and solar data, or gathering significantly more empirical power meter and prediction confidence data to better train the KDE models. Furthermore, one could integrate the device's actual input streams by using real power meter data and recorded prediction confidence over time, which would introduce greater real-life stochasticity into the environment.

5.2.2 Improving Agent Design and Evaluation

The current reward function, while functional and backed by literature, could certainly be improved. Future work should explore advanced reward structures beyond potential-based shaping to encourage more aggressive performance seeking without violating constraints. Additionally, training and comparing the policy against more diverse RL agents (e.g., DQ, A2C) would provide a stronger foundation for validating the PPO approach.

5.2.3 Policy Generalization

To address the limited generalizability, future research should investigate training the agent using scenarios that span multiple, highly varied geographical locations. Techniques like Domain Randomization or Meta-RL could be explored to train a policy capable of generalizing across different solar profiles and latitudes.

CHAPTER 6

Conclusion

The research first addressed the foundational question (RQ1): quantifying the influence of operational parameters on performance and power. This involved a parameter study based on real-world data generated in the lab, measuring power consumption and detection confidence across different FPS, resolution, and model variants.

The statistical evaluation confirmed the specific trade-offs inherent to the system. For power consumption, resolution increase significantly raises consumption up to 320px, but a clear plateau effect exists between 320px and 640px, where the difference is no longer statistically significant. This effect was most distinct at low frame rates; for instance, the yolo11n model at 1fps spanned a 1.29W difference in median power consumption, ranging from 3.78W (at 128px) to 5.07W (at 640px). At high FPS, these differences vanished, with median consumption clustered tightly near 5.0W ($\sim$ 5.03W to $\sim$ 5.15W). The largest median difference across all configurations was 1.37W. For prediction confidence, both the model variant and image resolution had a statistically significant influence, with a plateau effect also observed between the two largest resolutions and median differences up to 26 percent points.

To address the second research question (RQ2), the insights from RQ1 were utilized to first build a custom data-driven RL environment, and then train a PPO-based agent. This included modelling all four core simulation components and designing the custom RL environment. The agent was evaluated against static and random baselines in six scenarios designed to test adaptability and goal balancing. Quantitative evaluation showed promising results: the PPO agent consistently managed the trade-off better than the simple baselines, successfully outliving the more power-hungry policies while delivering better confidence than the least power-hungry baseline.

The qualitative evaluation of the battery and confidence traces clearly demonstrated that the agent learned a goal-oriented strategy: it scales up and uses its resources when power is abundant, achieving confidence on par with the most performant STATIC 3 baseline,

and knows to scale down and conserve its battery when incoming power is limited. The agent shines especially in the 48-hour long-term evaluation, where it demonstrated the learned ability to pace its energy consumption for the entire second-day cycle.

The implemented policy successfully validated the feasibility of using a learning-based approach to manage the dual objectives of survival and performance. While the agent is not claimed to be universally optimal, it robustly achieved the fundamental goals established for its design by demonstrably learning the basic dynamics of the simulated energy environment. Despite its current limitations, this work shows the potential for the implemented approach to be a promising strategy for dynamic power management in solar-powered Edge AI systems. Given the inherent practical constraints of the initial study, including the simplified stochasticity and the bounds of the empirically derived data, the achieved performance level represents a practical boundary for this specific prototype. This research, therefore, serves as a necessary and solid foundation for subsequent exploration into advanced adaptive policies for resource-constrained edge systems.

Bibliography

[1] Wazir Zada Khan, Ejaz Ahmed, Saqib Hakak, Ibrar Yaqoob, and Arif Ahmed. "Edge computing: A survey". In: *Future Generation Computer Systems* 97 (Aug. 2019), pp. 219–235. ISSN: 0167-739X. DOI: 10.1016/j.future.2019.02.050. URL: https://www.sciencedirect.com/science/article/pii/S0167739X18319903 (visited on 05/28/2024).

[2] Md Maruf Hossain Shuvo. "Edge AI: Leveraging the Full Potential of Deep Learning". en. In: *Recent Innovations in Artificial Intelligence and Smart Applications*. Ed. by Mostafa Al-Emran and Khaled Shaalan. Cham: Springer International Publishing, 2022, pp. 27–46. ISBN: 978-3-031-14748-7. DOI: 10.1007/978-3-031-14748-7_2. URL: https://doi.org/10.1007/978-3-031-14748-7_2 (visited on 05/28/2024).

[3] Raghubir Singh and Sukhpal Singh Gill. "Edge AI: A survey". In: *Internet of Things and Cyber-Physical Systems* 3 (Jan. 2023), pp. 71–92. ISSN: 2667-3452. DOI: 10.1016/j.iotcps.2023.02.004. URL: https://www.sciencedirect.com/science/article/pii/S2667345223000196 (visited on 03/26/2024).

[4] Congfeng Jiang, Tiantian Fan, Honghao Gao, Weisong Shi, Liangkai Liu, Christophe Cérin, and Jian Wan. "Energy aware edge computing: A survey". In: *Computer Communications* 151 (Feb. 2020), pp. 556–580. ISSN: 0140-3664. DOI: 10.1016/j.comcom.2020.01.004. URL: https://www.sciencedirect.com/science/article/pii/S014036641930831X (visited on 11/13/2025).

[5] Yu-Jen Ku, Sabur Baidya, and Sujit Dey. "Adaptive Computation Partitioning and Offloading in Real-Time Sustainable Vehicular Edge Computing". en. In: *IEEE Transactions on Vehicular Technology* 70.12 (Dec. 2021), pp. 13221–13237. ISSN: 0018-9545, 1939-9359. DOI: 10.1109/TVT.2021.3119585. URL: https://ieeexplore.ieee.org/document/9573263/ (visited on 03/12/2024).

[6] Sheng Zhang, Can Wang, Yibo Jin, Jie Wu, Zhuzhong Qian, Mingjun Xiao, and Sanglu Lu. "Adaptive Configuration Selection and Bandwidth Allocation for Edge-Based Video Analytics". In: *IEEE/ACM Transactions on Networking* 30.1 (Feb. 2022), pp. 285–298. ISSN: 1558-2566. DOI: 10.1109/TNET.2021.3106937. URL: https://ieeexplore.ieee.org/document/9525630 (visited on 10/09/2025).

[7] Yohann Rioual, Yannick Le Moullec, Johann Laurent, Muhidul Islam Khan, and Jean-Philippe Diguet. *Design and Comparison of Reward Functions in Reinforcement Learning for Energy Management of Sensor Nodes*. arXiv:2106.01114 [eess]. June 2021. DOI: 10.48550/arXiv.2106.01114. URL: http://arxiv.org/abs/2106.01114 (visited on 10/04/2025).

[8] Arief Setyanto, Theopilus Bayu Sasongko, Muhammad Ainul Fikri, and In Kee Kim. "Near-Edge Computing Aware Object Detection: A Review". In: *IEEE Access* 12 (2024), pp. 2989–3011. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2023.3347548. URL: https://ieeexplore.ieee.org/abstract/document/10374363 (visited on 11/26/2025).

[9] Qing Zhang, Xiaoyong Lin, Yongsheng Hao, and Jie Cao. "Energy-Aware Scheduling in Edge Computing Based on Energy Internet". In: *IEEE Access* 8 (2020), pp. 229052–229065. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.3044932. URL: https://ieeexplore.ieee.org/abstract/document/9294029 (visited on 11/13/2025).

[10] Alessandro Tundo, Marco Mobilio, Shashikant Ilager, Ivona Brandić, Ezio Bartocci, and Leonardo Mariani. "An Energy-Aware Approach to Design Self-Adaptive AI-based Applications on the Edge". In: *2023 38th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. ISSN: 2643-1572. Sept. 2023, pp. 281–293. DOI: 10.1109/ASE56229.2023.00046. URL: https://ieeexplore.ieee.org/abstract/document/10298366 (visited on 04/23/2024).

[11] Shreshth Tuli, Shashikant Ilager, Kotagiri Ramamohanarao, and Rajkumar Buyya. "Dynamic Scheduling for Stochastic Edge-Cloud Computing Environments Using A3C Learning and Residual Recurrent Neural Networks". en. In: *IEEE Transactions on Mobile Computing* 21.3 (Mar. 2022), pp. 940–954. ISSN: 1536-1233, 1558-0660, 2161-9875. DOI: 10.1109/TMC.2020.3017079. URL: https://ieeexplore.ieee.org/document/9169832/ (visited on 05/02/2024).

[12] Richard S. Sutton and Andrew Barto. *Reinforcement learning: an introduction*. en. Second edition. Adaptive computation and machine learning. Cambridge, Massachusetts London, England: The MIT Press, 2020. ISBN: 978-0-262-03924-6.

[13] Rahima Khanam and Muhammad Hussain. *YOLOv11: An Overview of the Key Architectural Enhancements*. arXiv:2410.17725 [cs]. Oct. 2024. DOI: 10.48550/arXiv.2410.17725. URL: http://arxiv.org/abs/2410.17725 (visited on 05/26/2025).

[14] *Gymnasium Documentation*. en. URL: https://gymnasium.farama.org/introduction/basic_usage.html (visited on 11/15/2025).

[15] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. *You Only Look Once: Unified, Real-Time Object Detection*. arXiv:1506.02640 [cs]. May 2016. DOI: 10.48550/arXiv.1506.02640. URL: http://arxiv.org/abs/1506.02640 (visited on 11/14/2025).

[16] Muhammad Farhan Hanif, Rauf Ahmad, Abdul Bari Farooq, Xiangtao Liu, Akhlaq Ahmad, Shahid Iqbal, and Jianchun Mi. "The Solar AI Nexus: Reinforcement Learning Shaping the Future of Energy Management". en. In: *WIREs Energy and Environment* 14.3 (2025). _eprint: https://wires.onlinelibrary.wiley.com/doi/pdf/10.1002/wene.70012, e70012. ISSN: 2041-840X. DOI: 10.1002/wene.70012. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/wene.70012 (visited on 11/09/2025).

[17] Danny Weyns. *An Introduction to Self-Adaptive Systems | Wiley Online Books*. English. 1st ed. John Wiley & Sons Ltd, 2020. ISBN: 978-1-119-57491-0. URL: https://onlinelibrary.wiley.com/doi/book/10.1002/9781119574910 (visited on 04/25/2024).

[18] Raspberry Pi Ltd. *Raspberry Pi 4 Model B*. en-GB. URL: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/ (visited on 11/28/2025).

[19] Danilo Greco, Majid Fasihiany, Ali Varasteh Ranjbar, Francesco Masulli, Stefano Rovetta, and Alberto Cabri. "Computer Vision Algorithms on a Raspberry Pi 4 for Automated Depalletizing". en. In: *Algorithms* 17.8 (Aug. 2024). Publisher: Multidisciplinary Digital Publishing Institute, p. 363. ISSN: 1999-4893. DOI: 10.3390/a17080363. URL: https://www.mdpi.com/1999-4893/17/8/363 (visited on 11/14/2025).

[20] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. *Proximal Policy Optimization Algorithms*. arXiv:1707.06347 [cs]. Aug. 2017. DOI: 10.48550/arXiv.1707.06347. URL: http://arxiv.org/abs/1707.06347 (visited on 11/05/2025).

[21] Jason Hsu, Sadaf Zahedi, Aman Kansal, Mani Srivastava, and Vijay Raghunathan. "Adaptive Duty Cycling for Energy Harvesting Systems". In: *ISLPED'06 Proceedings of the 2006 International Symposium on Low Power Electronics and Design*. Oct. 2006, pp. 180–185. DOI: 10.1145/1165573.1165616. URL: https://ieeexplore.ieee.org/document/4271832 (visited on 11/09/2025).

[22] Leander Hörmann, Thomas Hölzl, Christian Kastl, Peter Priller, Hans Peter Bernhard, Philipp Peterseil, and Andreas Springer. "Evaluation of Solar-based Energy Harvesting for Indoor IoT Applications". In: *European Test and Telemetry Conference (ettc2022)* (May 2022). DOI: 10.5162/ettc2022/8.1. URL: https://www.ama-science.org/proceedings/details/4221 (visited on 11/06/2025).

[23] Kevin Abas, Katia Obraczka, and Leland Miller. "Solar-powered, wireless smart camera network: An IoT solution for outdoor video monitoring". In: *Computer Communications* 118 (Mar. 2018), pp. 217–233. ISSN: 0140-3664. DOI: 10.1016/j.comcom.2018.01.007. URL: https://www.sciencedirect.com/science/article/pii/S0140366417305236 (visited on 11/06/2025).

[24] Meghana Tedla, Shubham Kulkarni, and Karthik Vaidhyanathan. *EcoMLS: A Self-Adaptation Approach for Architecting Green ML-Enabled Systems*. en. arXiv:2404.11411 [cs]. Apr. 2024. URL: http://arxiv.org/abs/2404.11411 (visited on 05/03/2024).

[25] Fabian Kaup, Philip Gottschling, and David Hausheer. "PowerPi: Measuring and modeling the power consumption of the Raspberry Pi". In: *39th Annual IEEE Conference on Local Computer Networks*. ISSN: 0742-1303. Sept. 2014, pp. 236–243. DOI: 10.1109/LCN.2014.6925777. URL: https://ieeexplore.ieee.org/abstract/document/6925777 (visited on 05/22/2025).

[26] Michal Prauzek, Jaromir Konecny, Jakub Hlavica, and Petr Musilek. "Self-learning for Day-night Mode Energy Strategy for Solar Powered Environmental WSN Nodes". In: *2020 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*. ISSN: 2576-7046. Aug. 2020, pp. 1–5. DOI: 10.1109/CCECE47787.2020.9255790. URL: https://ieeexplore.ieee.org/document/9255790 (visited on 11/06/2025).

[27] Qinglin Yang and Peng Li. "Deep Reinforcement Learning based Energy Scheduling for Edge Computing". In: *2020 IEEE International Conference on Smart Cloud (SmartCloud)*. Nov. 2020, pp. 175–180. DOI: 10.1109/SmartCloud49737.2020.00041. URL: https://ieeexplore.ieee.org/abstract/document/9265959 (visited on 11/09/2025).

[28] Amanda Jayanetti, Saman Halgamuge, and Rajkumar Buyya. "Deep reinforcement learning for energy and time optimized scheduling of precedence-constrained tasks in edge–cloud computing environments". In: *Future Generation Computer Systems* 137 (Dec. 2022), pp. 14–30. ISSN: 0167-739X. DOI: 10.1016/j.future.2022.06.012. URL: https://www.sciencedirect.com/science/article/pii/S0167739X22002230 (visited on 11/09/2025).

[29] Md. Shirajum Munir, Sarder Fakhrul Abedin, Nguyen H. Tran, Zhu Han, Eui-Nam Huh, and Choong Seon Hong. "Risk-Aware Energy Scheduling for Edge Computing With Microgrid: A Multi-Agent Deep Reinforcement Learning Approach". In: *IEEE Transactions on Network and Service Management* 18.3 (Sept. 2021), pp. 3476–3497. ISSN: 1932-4537. DOI: 10.1109/TNSM.2021.3049381. URL: https://ieeexplore.ieee.org/abstract/document/9313066 (visited on 11/09/2025).

[30] Yang Wang, Kun Yang, Weixiang Wan, and Haibo Mei. "Adaptive energy saving algorithms for Internet of Things devices integrating end and edge strategies". en. In: *Transactions on Emerging Telecommunications Technologies* 32.8 (2021). _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/ett.4122, e4122. ISSN: 2161-3915. DOI: 10.1002/ett.4122. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.4122 (visited on 03/12/2024).

[31] Technische Universität Wien. *HPC Research Group*. 2025. URL: https://hpc.ec.tuwien.ac.at/ (visited on 05/26/2025).

[32] *Camera - Raspberry Pi Documentation*. en. URL: https://www.raspberrypi.com/documentation/accessories/camera.html (visited on 11/14/2025).

[33] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. *Microsoft COCO: Common Objects in Context*. arXiv:1405.0312 [cs]. Feb. 2015. DOI: 10.48550/arXiv.1405.0312. URL: http://arxiv.org/abs/1405.0312 (visited on 05/26/2025).

[34] S. S. SHAPIRO and M. B. WILK. "An analysis of variance test for normality (complete samples)†". In: *Biometrika* 52.3-4 (Dec. 1965), pp. 591–611. ISSN: 0006-3444. DOI: 10.1093/biomet/52.3-4.591. URL: https://doi.org/10.1093/biomet/52.3-4.591 (visited on 11/28/2025).

[35] W. J. Conover. *Practical Nonparametric Statistics*. en. Google-Books-ID: n_39DwAAQBAJ. John Wiley & Sons, Jan. 1999. ISBN: 978-0-471-16068-7.

[36] Jean Dickinson Gibbons and Subhabrata Chakraborti. "Nonparametric Statistical Inference". en. In: *International Encyclopedia of Statistical Science*. Springer, Berlin, Heidelberg, 2011, pp. 977–979. ISBN: 978-3-642-04898-2. DOI: 10.1007/978-3-642-04898-2_420. URL: https://link.springer.com/rwe/10.1007/978-3-642-04898-2_420 (visited on 11/28/2025).

[37] Olive Jean Dunn. "Multiple Comparisons Using Rank Sums". In: *Technometrics* 6.3 (Aug. 1964). Publisher: ASA Website _eprint: https://www.tandfonline.com/doi/pdf/10.1080/00401706.1964.10490181, pp. 241–252. ISSN: 0040-1706. DOI: 10.1080/00401706.1964.10490181. URL: https://www.tandfonline.com/doi/abs/10.1080/00401706.1964.10490181 (visited on 11/28/2025).

[38] Duffie and Beckmann. "Solar Radiation". en. In: *Solar Engineering of Thermal Processes*. Section: 1 _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781118671603.ch1. John Wiley & Sons, Ltd, 2013, pp. 3–42. ISBN: 978-1-118-67160-3. DOI: 10.1002/9781118671603.ch1. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118671603.ch1 (visited on 11/02/2025).

[39] Eitan Altman. *Constrained Markov Decision Processes*. New York: Routledge, Dec. 2021. ISBN: 978-1-315-14022-3. DOI: 10.1201/9781315140223.

[40] Andrew Y. Ng, Daishi Harada, and Stuart J. Russell. "Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping". In: *Proceedings of the Sixteenth International Conference on Machine Learning*. ICML '99. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., June 1999, pp. 278–287. ISBN: 978-1-55860-612-8. (Visited on 11/04/2025).

[41] E. Wiewiora. "Potential-Based Shaping and Q-Value Initialization are Equivalent". In: *Journal of Artificial Intelligence Research* 19 (Sept. 2003). arXiv:1106.5267 [cs], pp. 205–208. ISSN: 1076-9757. DOI: 10.1613/jair.1190. URL: http://arxiv.org/abs/1106.5267 (visited on 11/04/2025).

[42] *DLR-RM/stable-baselines3*. original-date: 2020-05-05T05:52:26Z. Nov. 2025. URL: https://github.com/DLR-RM/stable-baselines3 (visited on 11/28/2025).

[43] Neil De La Fuente and Daniel A. Vidal Guerra. *A Comparative Study of Deep Reinforcement Learning Models: DQN vs PPO vs A2C*. arXiv:2407.14151 [cs]. July 2024. DOI: 10.48550/arXiv.2407.14151. URL: http://arxiv.org/abs/2407.14151 (visited on 11/05/2025).

[44] Can Wang, Sheng Zhang, Yu Chen, Zhuzhong Qian, Jie Wu, and Mingjun Xiao. "Joint Configuration Adaptation and Bandwidth Allocation for Edge-based Real-time Video Analytics". In: *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*. ISSN: 2641-9874. July 2020, pp. 257–266. DOI: 10.1109/INFOCOM41043.2020.9155524. URL: https://ieeexplore.ieee.org/abstract/document/9155524 (visited on 11/06/2025).

[45] Navin Sharma, Jeremy Gummeson, David Irwin, and Prashant Shenoy. "Cloudy Computing: Leveraging Weather Forecasts in Energy Harvesting Sensor Systems". In: *2010 7th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*. ISSN: 2155-5494. June 2010, pp. 1–9. DOI: 10.1109/SECON.2010.5508260. URL: https://ieeexplore.ieee.org/abstract/document/5508260 (visited on 11/25/2025).

Overview of Generative AI Tools Used

Generative AI tools were utilized throughout this Diploma Thesis strictly as an aid for structure, refinement, and technical assistance. The core research, methodology, empirical data collection, statistical analysis, and final conclusions were the result of independent intellectual effort, and the content and composition of the final text are entirely the author's work. Specifically, Gemini Pro 2.5 flash provided support in standardizing the tone and phrasing across all chapters, ensuring style consistency, assisting with formulation, and performing essential spellchecking. For technical documentation, ChatGPT 5.1 was employed to streamline the presentation of empirical findings by generating LaTeX code for tables and figures from raw data inputs, while ChatGPT 4.0 served as a utility for code review and debugging. All usage adheres to the principle that the author's original intellectual and creative efforts remain predominant in this work.

List of Figures

List of Tables

Appendix A: Extended 24h Results

Table 1: Performance metrics for Summer_Clear scenario

Metric	ppo	random	static1	static2	static3
Avg_Time_Till_First_Death_h	19.83	19.43	20.51	18.83	19.00
Avg_Empty_h	4.33	4.74	3.66	5.33	5.17
Avg_Time_Alive_h	19.67	19.26	20.34	18.67	18.83
Avg_SLA_%	72.78	58.19	0.00	99.96	100.00
Avg_Confidence	0.625	0.607	0.562	0.631	0.648
Avg_Reward	-0.105	-0.118	-0.107	-0.124	-0.120

Table 2: Performance metrics for Winter_Clear scenario

Metric	ppo	random	static1	static2	static3
Avg_Time_Till_First_Death_h	18.48	18.29	19.66	17.17	17.33
Avg_Empty_h	5.69	5.88	4.51	7.00	6.84
Avg_Time_Alive_h	18.31	18.12	19.49	17.00	17.16
Avg_SLA_%	55.51	58.88	0.03	99.97	100.00
Avg_Confidence	0.612	0.608	0.562	0.631	0.648
Avg_Reward	-0.130	-0.136	-0.121	-0.148	-0.144

Table 3: Performance metrics for Winter_Cloudy scenario

Metric	ppo	random	static1	static2	static3
Avg_Time_Till_First_Death_h	9.45	8.83	10.14	8.17	8.33
Avg_Empty_h	14.72	15.34	14.03	16.00	15.83
Avg_Time_Alive_h	9.28	8.66	9.97	8.00	8.17
Avg_SLA_%	45.68	58.29	0.01	99.96	100.00
Avg_Confidence	0.601	0.607	0.562	0.631	0.648
Avg_Reward	-0.240	-0.248	-0.233	-0.256	-0.254

Table 4: Performance metrics for Early_Morning scenario

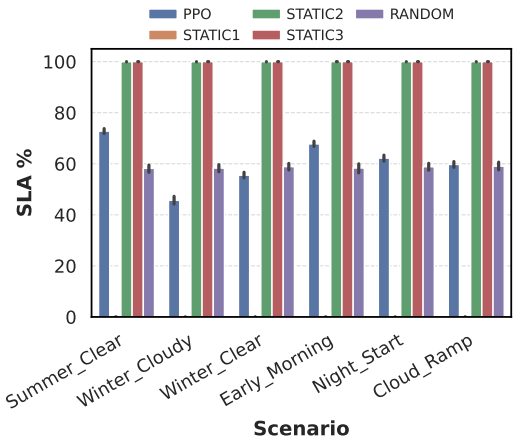
Metric	ppo	random	static1	static2	static3
Avg_Time_Till_First_Death_h	21.83	21.39	22.51	20.83	21.00
Avg_Empty_h	2.34	2.78	1.65	3.33	3.17
Avg_Time_Alive_h	21.66	21.22	22.35	20.67	20.83
Avg_SLA_%	67.76	58.32	0.00	100.00	100.00
Avg_Confidence	0.621	0.607	0.562	0.631	0.649
Avg_Reward	-0.092	-0.105	-0.093	-0.112	-0.108

Table 5: Performance metrics for Night_Start scenario

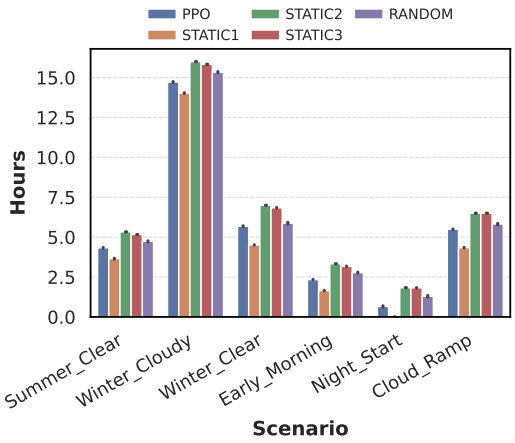
Metric	ppo	random	static1	static2	static3
Avg_Time_Till_First_Death_h	8.52	7.99	24.00	7.50	7.51
Avg_Empty_h	0.66	1.30	0.00	1.83	1.83
Avg_Time_Alive_h	23.34	22.70	24.00	22.17	22.17
Avg_SLA_%	62.21	58.82	0.01	99.99	100.00
Avg_Confidence	0.617	0.607	0.562	0.631	0.648
Avg_Reward	-0.087	-0.101	-0.081	-0.109	-0.106

Table 6: Performance metrics for Cloud_Ramp scenario

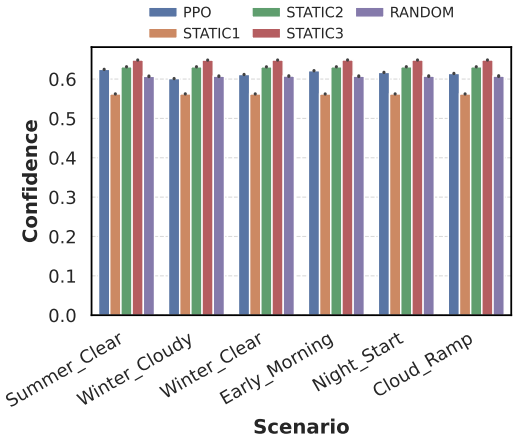
Metric	ppo	random	static1	static2	static3
Avg_Time_Till_First_Death_h	18.67	18.34	19.83	17.67	17.67
Avg_Empty_h	5.50	5.82	4.34	6.50	6.50
Avg_Time_Alive_h	18.50	18.18	19.66	17.50	17.50
Avg_SLA_%	59.72	59.03	0.03	99.97	100.00
Avg_Confidence	0.614	0.607	0.562	0.631	0.648
Avg_Reward	-0.136	-0.144	-0.126	-0.153	-0.150



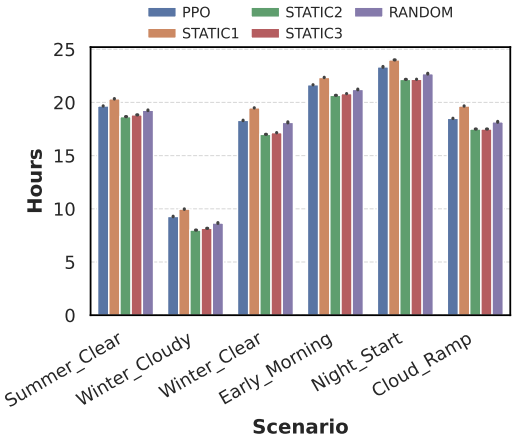
(a) Avg SLA %



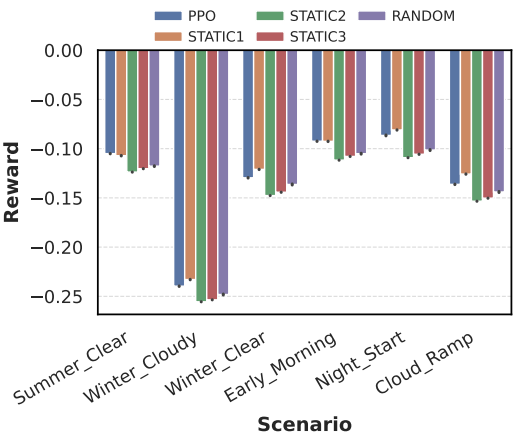
(b) Avg empty hours



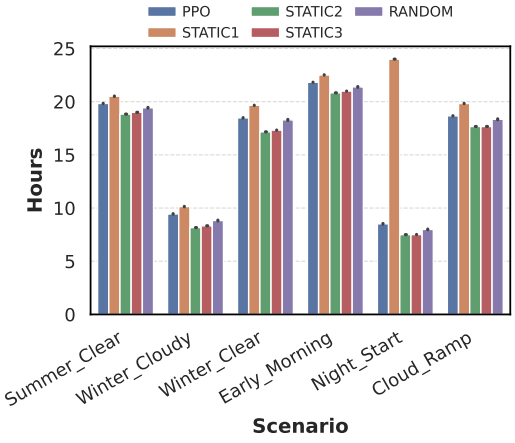
(c) Avg confidence



(d) Avg total time alive

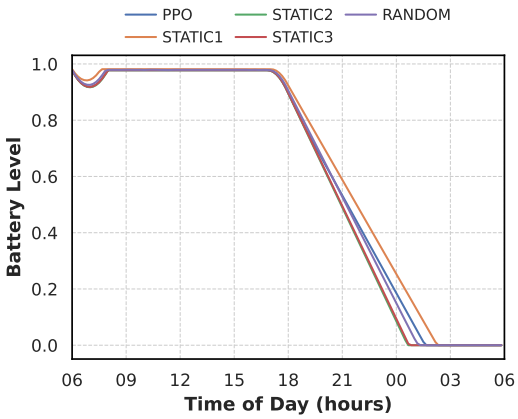


(e) Avg reward

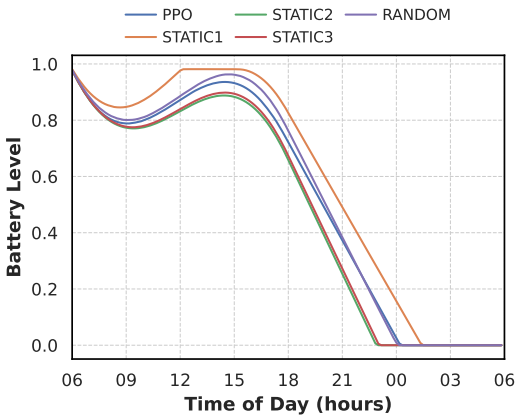


(f) Avg time until first battery death

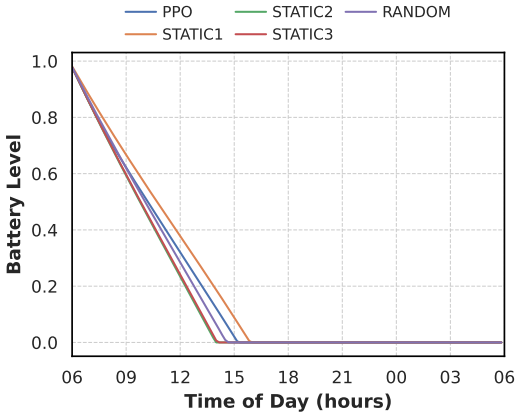
Figure 1: 24h aggregated evaluation metrics



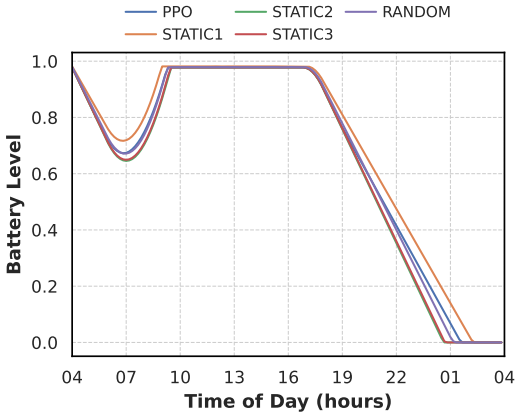
(a) Summer_Clear



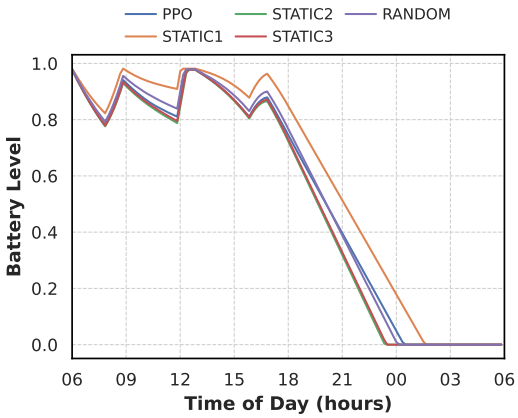
(b) Winter_Clear



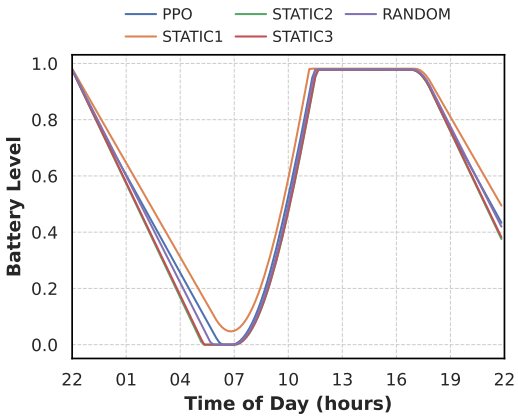
(c) Winter_Cloudy



(d) Early_Morning

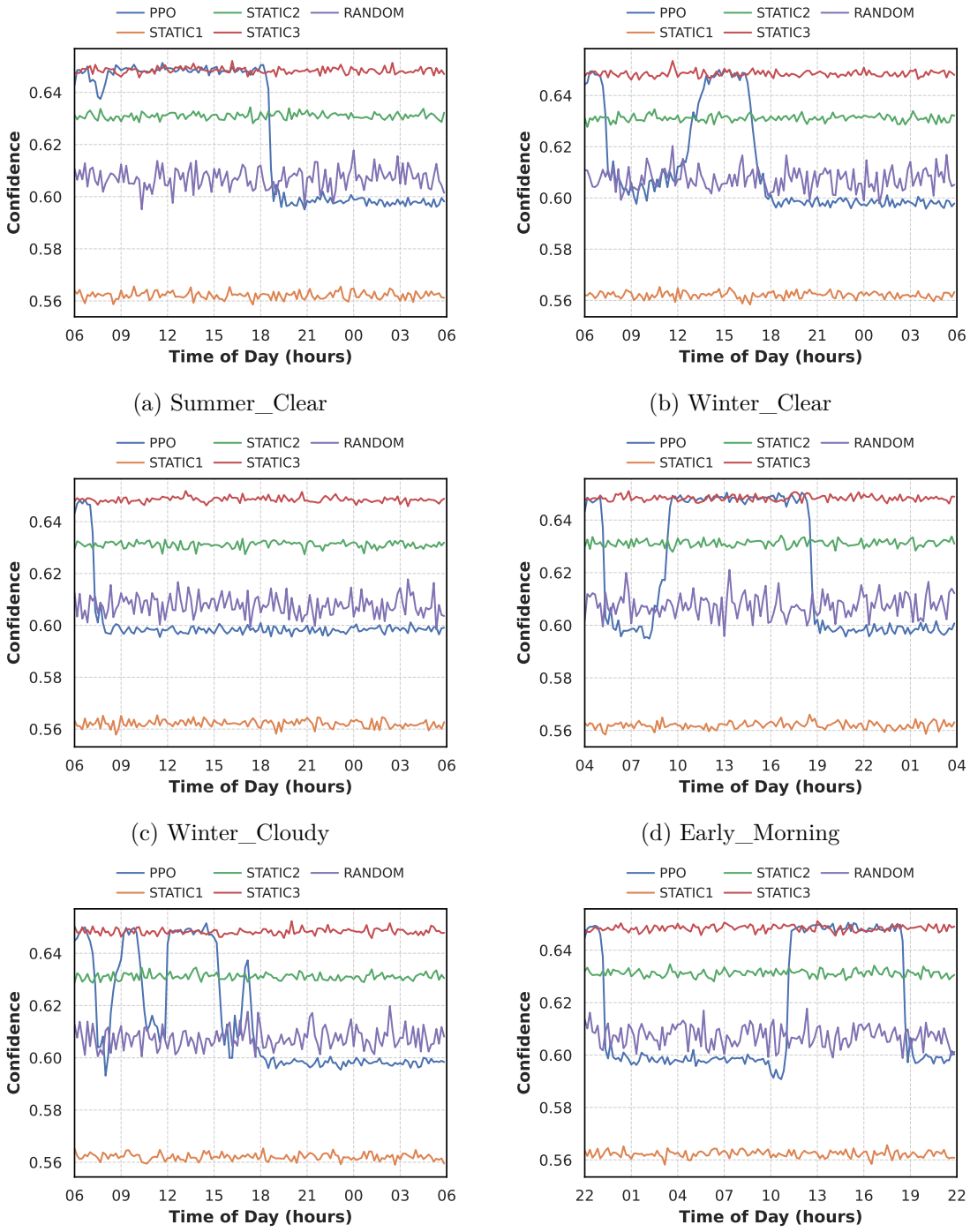


(e) Cloud_Ramp



(f) Night_Start

Figure 2: 24hh battery traces by scenario and agent



(a) Summer_Clear

(b) Winter_Clear

(c) Winter_Cloudy

(d) Early_Morning

(e) Cloud_Ramp

(f) Night_Start

Figure 3: 24h confidence traces by scenario and agent

Appendix B: Extended 48h Results

Table 7: Performance metrics for Summer_Clear scenario

Metric	ppo	random	static1	static2	static3
Avg_Time_Till_First_Death_h	19.83	19.41	20.51	18.83	19.00
Avg_Empty_h	9.66	10.60	8.29	11.83	11.50
Avg_Time_Alive_h	38.34	37.40	39.71	36.17	36.50
Avg_SLA_%	66.58	58.85	0.00	99.99	100.00
Avg_Confidence	0.620	0.608	0.562	0.631	0.648
Avg_Reward	-0.111	-0.123	-0.112	-0.130	-0.126

Table 8: Performance metrics for Winter_Clear scenario

Metric	ppo	random	static1	static2	static3
Avg_Time_Till_First_Death_h	18.48	18.25	19.65	17.17	17.33
Avg_Empty_h	20.04	21.34	18.36	23.40	23.16
Avg_Time_Alive_h	27.96	26.66	29.64	24.60	24.84
Avg_SLA_%	49.45	59.12	0.00	99.97	100.00
Avg_Confidence	0.605	0.608	0.562	0.631	0.648
Avg_Reward	-0.162	-0.170	-0.156	-0.180	-0.177

Table 9: Performance metrics for Winter_Cloudy scenario

Metric	ppo	random	static1	static2	static3
Avg_Time_Till_First_Death_h	9.43	8.81	10.14	8.17	8.33
Avg_Empty_h	38.74	39.35	38.03	40.00	39.83
Avg_Time_Alive_h	9.26	8.65	9.97	8.00	8.17
Avg_SLA_%	43.90	58.58	0.00	99.99	100.00
Avg_Confidence	0.600	0.607	0.562	0.631	0.648
Avg_Reward	-0.270	-0.274	-0.267	-0.278	-0.277

Table 10: Performance metrics for Early_Morning scenario

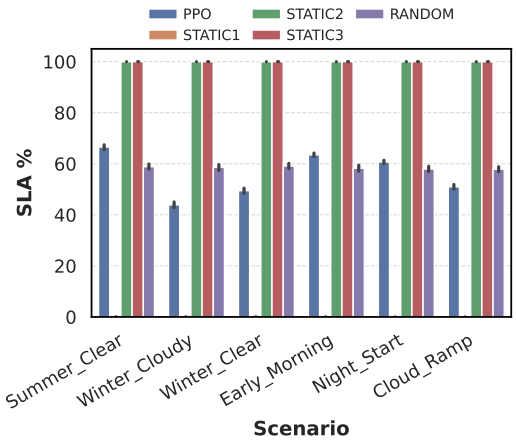
Metric	ppo	random	static1	static2	static3
Avg_Time_Till_First_Death_h	21.83	21.37	22.50	20.83	21.00
Avg_Empty_h	7.68	8.66	6.33	9.83	9.50
Avg_Time_Alive_h	40.32	39.34	41.67	38.17	38.50
Avg_SLA_%	63.50	58.22	0.00	99.96	100.00
Avg_Confidence	0.618	0.607	0.562	0.631	0.648
Avg_Reward	-0.105	-0.117	-0.105	-0.124	-0.120

Table 11: Performance metrics for Night_Start scenario

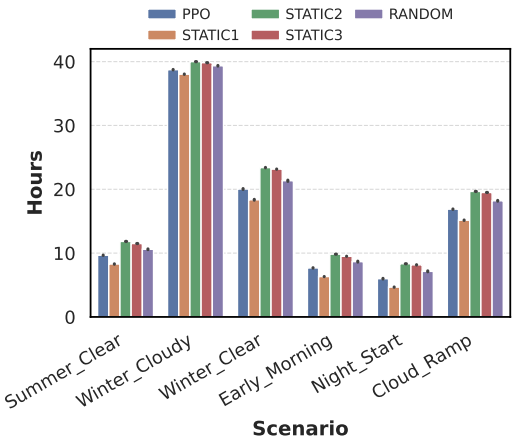
Metric	ppo	random	static1	static2	static3
Avg_Time_Till_First_Death_h	8.53	8.00	28.50	7.50	7.51
Avg_Empty_h	5.99	7.14	4.67	8.33	8.16
Avg_Time_Alive_h	42.01	40.86	43.33	39.67	39.84
Avg_SLA_%	60.70	57.93	0.01	99.98	100.00
Avg_Confidence	0.615	0.607	0.562	0.631	0.648
Avg_Reward	-0.102	-0.115	-0.099	-0.122	-0.119

Table 12: Performance metrics for Cloud_Ramp scenario

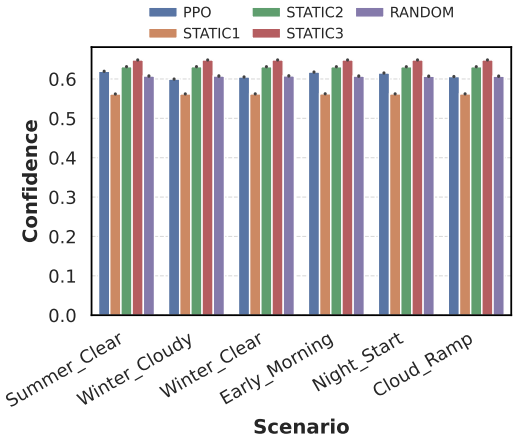
Metric	ppo	random	static1	static2	static3
Avg_Time_Till_First_Death_h	18.66	18.33	19.84	17.67	17.67
Avg_Empty_h	16.89	18.20	15.15	19.67	19.50
Avg_Time_Alive_h	31.11	29.80	32.85	28.33	28.50
Avg_SLA_%	50.99	57.84	0.01	99.97	100.00
Avg_Confidence	0.606	0.607	0.562	0.631	0.648
Avg_Reward	-0.159	-0.169	-0.152	-0.177	-0.175



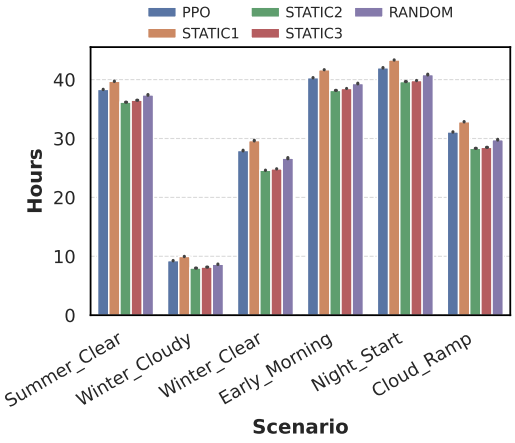
(a) Avg SLA %



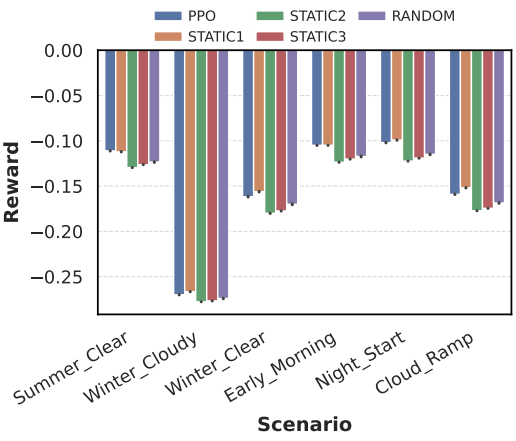
(b) Avg empty hours



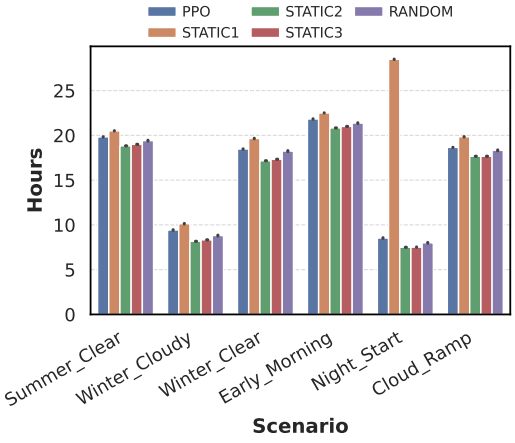
(c) Avg confidence



(d) Avg total time alive

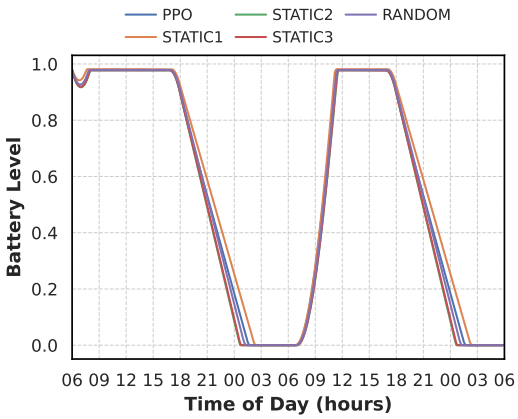


(e) Avg reward

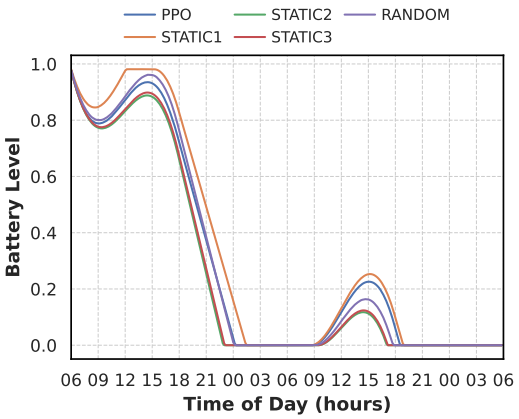


(f) Avg time until first battery death

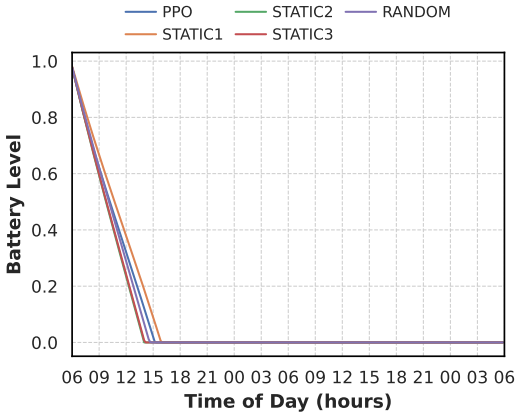
Figure 4: 48h aggregated evaluation metrics



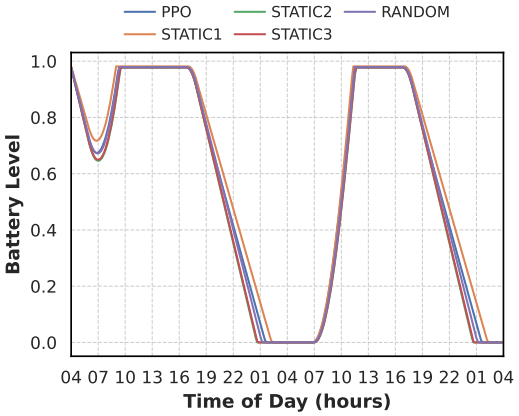
(a) Summer_Clear



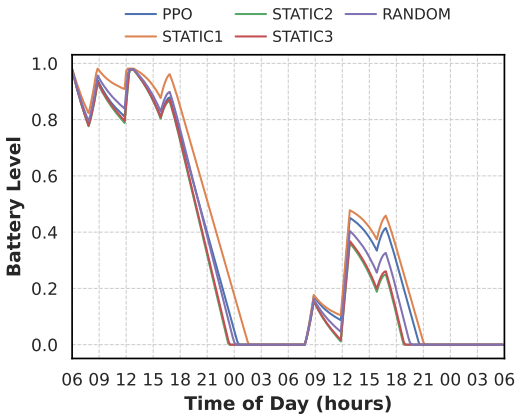
(b) Winter_Clear



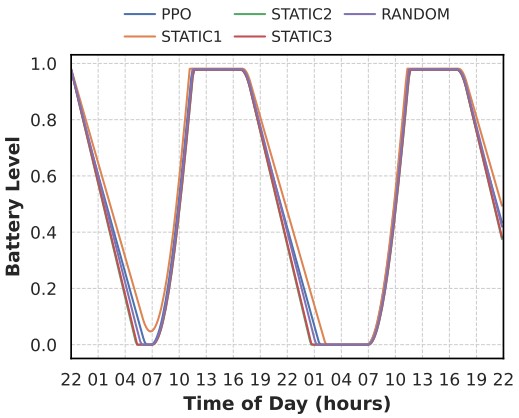
(c) Winter_Cloudy



(d) Early_Morning

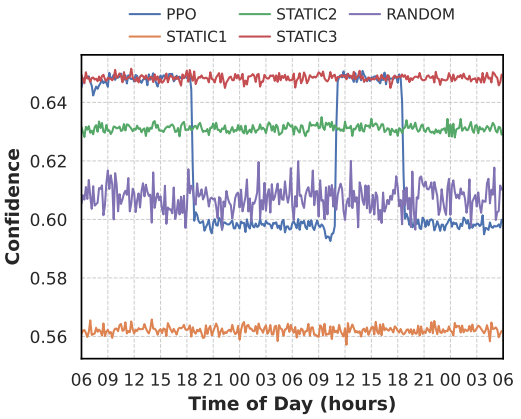


(e) Cloud_Ramp

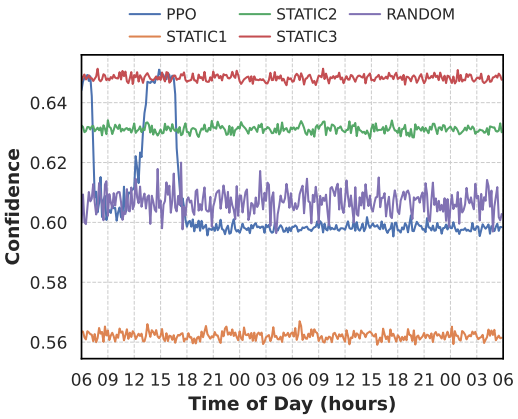


(f) Night_Start

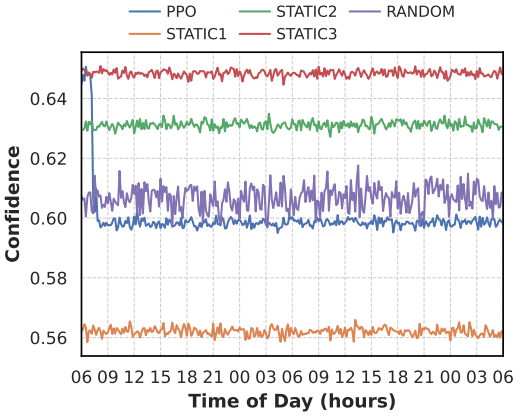
Figure 5: 48h battery traces by scenario and agent



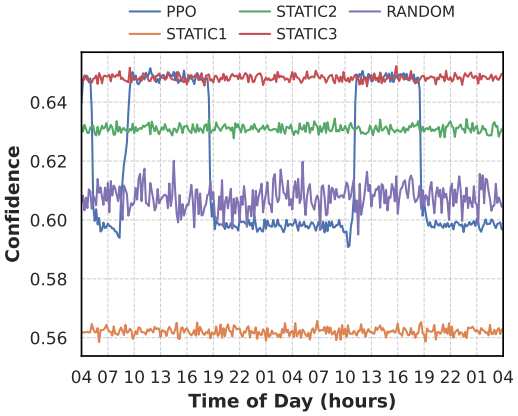
(a) Summer_Clear



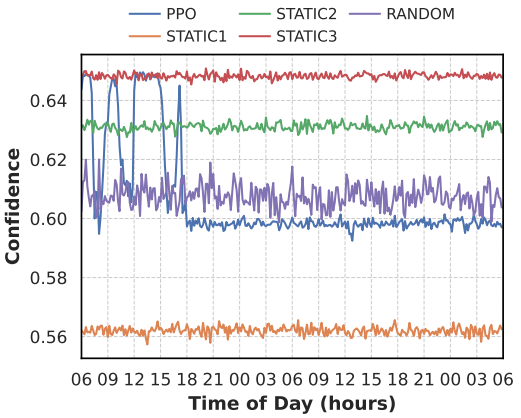
(b) Winter_Clear



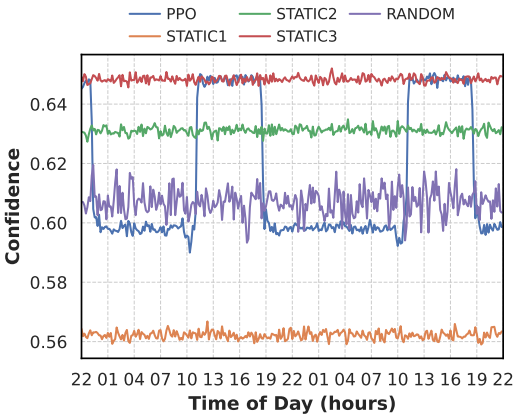
(c) Winter_Cloudy



(d) Early_Morning



(e) Cloud_Ramp



(f) Night_Start

Figure 6: 48h confidence traces by scenario and agent