Quest Journals
Journal of Electronics and Communication Engineering Research
Volume 12 ~ Issue 6 (Nov. – Dec. 2026) pp: 01-10
ISSN(Online): 2321-5941
www.questjournals.org

Research Paper

Joint Reinforcement Learning Control for Full-Day Solar Street Lighting

Dan Dan-Abia[*1], Kingsley Udofia[2], Dr. Kufre Udofia[3]
[*1]*Department of Electrical/Electronics Engineering, University of Uyo, Uyo, Nigeria*
[2]*Department of Electrical/Electronics Engineering, University of Uyo, Uyo, Nigeria*
[3]*Department of Electrical/Electronics Engineering, University of Uyo, Uyo, Nigeria*
Corresponding Author: Dan Dan-Abia

Abstract
A solar street light fails in a way that residents notice at once: the lamp goes dark. For a battery-backed node with a small panel, avoiding that outcome means spreading a fixed store of energy across the day while still sensing and lighting well enough to be useful. This paper examines how learned controllers manage that budget. A simulated street-lighting node with a 100,000 J battery, a 100 mW peak panel, and an LED drawing 0.2, 0.8, or 1.6 W is run by a policy that picks one of 27 joint actions at each step, combining a TinyML model, a sensing frequency, and an LED brightness. Tabular Q-learning, linear function approximation, and a deep Q-network (DQN) were compared with five baseline policies over 20 greedy evaluation episodes of 1,440 minutes. The learned controllers earned cumulative rewards of 298.7 to 349.6, while the baselines scored between −355.6 and −822.7. DQN kept the node alive for the whole day with 94.9% accuracy, and linear function approximation reached 94.7%. The Greedy baseline emptied the battery at minute 741. An energy-envelope calculation explains why: a lamp held at its brightest level would drain a full battery in about 1,040 minutes, so the real task is allocation. The differences were significant at the 95% confidence level. The results support service continuity as a primary outcome measure for solar lighting controllers.

Keywords: *Deep Q-network, edge IoT, energy autonomy, energy harvesting, reinforcement learning, service continuity, smart street lighting, TinyML*

*Received 26 Oct., 2026; **R**evised 04 Nov., 2026; **A**ccepted 06 Nov., 2026 © The author(s) 2026.*
P**ublished with open access at **www.questjournals.org

I. INTRODUCTION

Street lighting is one of the most visible services a city provides, and one of the easiest to judge. When a lamp works, nobody notices; when it goes dark, everybody does. Solar street lights are popular where extending the grid is costly, and many of them now carry sensors and small processors so that they can respond to traffic and ambient light [1], [2]. A battery and a small photovoltaic (PV) panel carry the load [3]. The arrangement removes the cable but adds a constraint that the lamp has to live with every day: the energy available is whatever the battery holds plus whatever the panel manages to add.

The difficulty is one of timing. Demand for light is greatest at night, when the panel produces nothing, so the battery must carry most of the day's energy forward [3]. The usual design goal is energy-neutral operation, in which consumption does not exceed harvest over the long run [4], and the usual tools are thresholds, duty cycling, and scheduling rules [5], [6]. Those tools are simple and often adequate, but they are tuned for a site and a season, and they respond to the present state of the battery with little regard for the hours still ahead.

Reinforcement learning (RL) is attractive for exactly that reason. A learned policy is trained against the consequences of its decisions over a whole day and can learn to hold back energy early because it will be needed later [7]. Earlier RL work on harvesting nodes has mostly controlled one quantity at a time, such as a duty cycle or a sampling rate [8]–[11]. A street-lighting node has several levers that draw from the same battery: the lamp, the sensing front end, and the model used for on-device inference. They are normally set separately, yet they compete for the same joules, and choosing one without regard to the others can overspend the budget.

This paper studies the joint control of those three levers and judges it by one outcome that matters to a city: whether the light stays on. A solar street-lighting node is simulated with a 100,000 J battery and a 100 mW peak panel. At every decision step the controller selects one of 27 joint actions, each fixing a TinyML model, a

sensing frequency, and an LED brightness. Three RL controllers (tabular Q-learning, linear function approximation, and a deep Q-network) are compared with five baseline policies over 20 greedy evaluation episodes, each covering a full day of 1,440 minutes. if the node that shows, from its stated parameters, how little room a day of lighting leaves for waste; (ii) a joint formulation of lamp, sensing, and model selection as a single control problem with a 27-action space; (iii) an outcome-based comparison of three RL families and five baselines, with service continuity as the headline measure; and (iv) evidence that learned controllers keep the node alive through the day where a greedy rule does not, without giving up accuracy. The remainder of the paper is organized as follows. Section II (Body Text) covers the background, the system model, the controllers, and the experimental setup. Section III presents and discusses the results, and Section IV gives the conclusions and next steps.

II. BACKGROUND

2.1 Background and Related Work

2.1.1 Solar Street Lighting and the Night-Time Energy Problem

A solar luminaire stores in daylight what it spends after dark. The panel charges the battery while the sun is up, and the lamp, along with any sensors and the controller, draws it down through the night. The mismatch between when energy arrives and when it is needed is structural, and a run of overcast days can leave the battery empty [3]. Adaptive lighting reduces the load by dimming when traffic is light, and street lights are a natural target for urban sensing and control more generally [1]. A recent system adjusts streetlight intensity from video-based detection on an embedded GPU and reports lower late-night energy use [12]. That work attacks the demand side with a supervised perception model; it does not treat the battery as a budget to be planned across the day.

2.1.2 Energy-Neutral Power Management

The theory of energy-neutral operation for harvesting nodes grew out of sensor-network research. Kansal et al. showed that a node which adapts its activity to predicted harvest can sustain more work than one running at a fixed rate [4], and adaptive duty cycling follows the same logic [5]. Harvest predictors based on weighted averages are common [13], and scheduling methods can give timing guarantees under energy limits [6]. Model predictive control handles constraints explicitly, at the price of solving an optimization problem online, which strains a small microcontroller [14]. All of these depend on a good model of the harvest or the system, and they tend to degrade when conditions drift away from the ones assumed at design time.

2.1.3 Learning-Based Control of Harvesting Nodes

RL removes the need for an explicit model by learning from interaction. Hsu et al. used it for time-of-day power provisioning [8], and Shresthamali et al. for power management of a solar node [9]. RLMan, an actor-critic energy manager, targets energy-neutral operation [10]. ACES configures sensor parameters automatically [11], a later study shares one policy across similar nodes to cut training effort [15], and tinyMAN fits a lightweight energy manager onto low-end microcontrollers [16]. Table 1 lists what each of these controls.

Table 1: Quantities Controlled in Closely Related Work and in This Study

Ref.	Node type	Quantity controlled
[8]	Harvesting sensor node	Node activity through time-of-day provisioning
[9]	Solar sensor node	Power use within the energy budget
[10]	Harvesting sensor network node	Energy use of the node
[11]	Harvesting sensor	Sensor configuration
[15]	Indoor solar sensor	Sampling rate
[16]	Harvesting wearable device	Energy use of the device
[12]	Street light (non-RL)	Lamp intensity
This study	Solar street-lighting node	LED brightness, sensing frequency, and TinyML model together

2.1.4 What Is Missing

Two things stand out in this body of work. The controlled quantity is nearly always a single one, so the interaction between lamp, sensing, and inference has had little attention. And the outcome that matters to the owner of a street light, whether it stays lit for the whole night, is rarely the headline result. Studies tend to report reward, utility, or energy saved, and their baselines, harvest traces, and metrics differ enough that comparison across papers is difficult [17], [18]. This paper reports continuity directly and gives it the same prominence as reward.

2.1.5 On-Device Models as an Energy Decision

Choosing a TinyML model is not only an accuracy decision. Inference runtimes for microcontrollers and optimized kernels for Cortex-M processors make small networks practical on a lamp controller [19], [20], and a developer can pick from models of different size and cost [21]. Compression by quantization and pruning narrows the choice further [22], [23]. Each option has an energy price, and the price is dominated as much by moving data through memory as by arithmetic [24], [25]. A heavier model may classify more accurately and drain the battery faster. On a node that has to last a day, the model therefore belongs in the control problem, next to the lamp and the sensing rate, and not outside it as a fixed design choice.

2.2 System Model and Problem Formulation

2.2.1 Energy Flow of the Node

Fig. 1 shows how energy moves through the simulated node. The PV panel charges the battery, and the battery supplies three loads: the LED lamp, the sensing front end, and the microcontroller that runs the TinyML model. A learned controller watches the state of the node and, at each step, chooses the settings of all three loads at once.

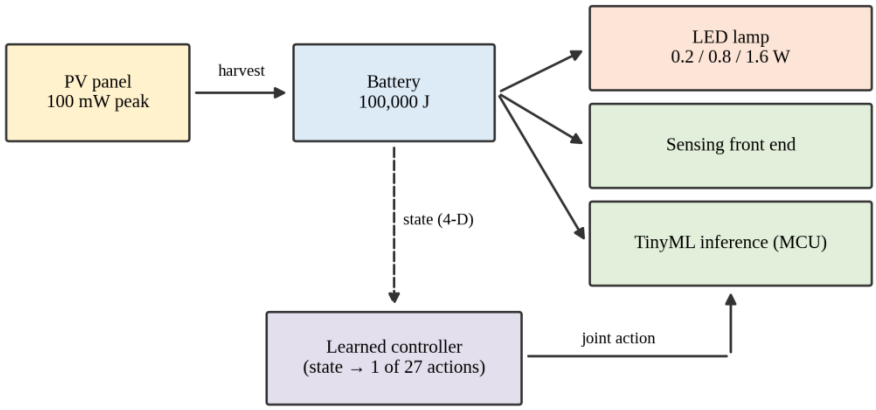


Figure 1: Energy flow and control loop of the solar street-lighting node.

The panel has a peak output of 100 mW, and its harvested power follows the usual conversion model

$$P_{h,t} = \eta_{pv} \cdot A \cdot G_t \cdot (1 - \xi_t) \quad (1)$$

where G_t is the irradiance, A the panel area, η_{pv} the conversion efficiency, and ξ_t a loss term for soiling, shading, and temperature. The battery holds at most E_{max} = 100,000 J and is updated each step by

$$E_{t+1} = \min\{ E_{max}, \max\{ 0, E_t + \eta_c P_{h,t} \Delta t - P_{l,t} \Delta t / \eta_d \} \} \quad (2)$$

with charge and discharge efficiencies η_c and η_d. The load power is the sum of the three consumers:

$$P_{l,t} = P_{LED,t} + P_{sense,t} + P_{inf,t} \quad (3)$$

The LED term takes one of three values, 0.2 W, 0.8 W, or 1.6 W. The sensing and inference terms depend on the sensing frequency and on the TinyML model the controller has selected.

2.2.2 The Joint Action Space

Each decision picks one setting for each of three levers. The TinyML model determines how accurate the node's inference is and how much energy each inference costs. The sensing frequency determines how fresh the node's information is and how much the front end draws. The LED brightness determines how well the street is lit and by far the largest share of the load. Combined, the choices form a space of 27 joint actions (Fig. 2).

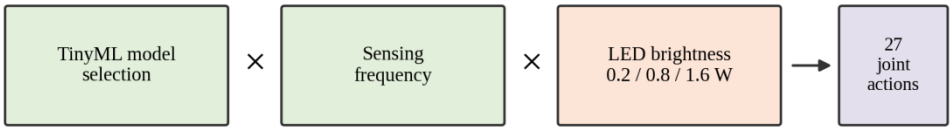


Figure 2: The joint action space. Each decision fixes a model, a sensing frequency, and an LED level.

Treating the levers jointly matters because they trade against each other. A controller that can only dim the lamp has one way to save energy. A controller that can also slow the sensing or move to a lighter model has several, and some of them may cost less in service than dimming does.

2.2.3 State and Reward

The control problem is a Markov decision process with a four-dimensional state and the 27-action space above [7]. The reward trades service against energy and penalizes running the battery down to its floor:

$$r_t = w_s s_t - w_p p_t - w_d \cdot \mathbb{1}[\, E_t \le E_{\min} \,] \tag{4}$$

where s_t is the service delivered, p_t the energy spent, and the last term the depletion penalty. The weights shape behavior strongly. A weak depletion penalty invites a controller to spend early, and a strong one invites it to sit in the dark [26].

2.2.4 Service Continuity and the Energy Envelope

The main outcome measure is service continuity, the number of minutes the node operates before the battery is first exhausted, capped at the length of the episode:

$$T_c = \min\{\, 1440, \min\{\, t : E_t = 0 \,\} \,\} \tag{5}$$

A value of 1,440 therefore means the lights stayed on for the whole day. To see what a controller is up against, consider how long a full battery would last if the lamp alone were held at one level with no harvest at all:

$$\tau_L = E_{\max} / (\, 60 \cdot P_L \,) \quad [\text{min}] \tag{6}$$

Table 2 evaluates this for the three LED levels. It uses only the stated parameters of the node. The result is sobering. At the brightest level the battery lasts about 1,042 minutes, which is less than a day, and a full day at that level would need 138.2% of the battery. Even at the middle level a day of constant lighting would take more than two thirds of the stored energy. On the supply side, a panel that delivered its full 100 mW for all 1,440 minutes would add only 8,640 J, or 8.6% of the battery, and in practice the panel delivers far less.

Table 2: Energy Envelope of the Node (Lamp Load Only, No Harvest)

LED level	Endurance on a full battery	Energy for 1,440 min at that level	Share of battery
0.2 W	8,333 min (5.8 days)	17,280 J	17.3%
0.8 W	2,083 min (34.7 h)	69,120 J	69.1%
1.6 W	1,042 min (17.4 h)	138,240 J	138.2%
Panel at 100 mW for 1,440 min	—	8,640 J harvested (upper bound)	8.6%

Two consequences follow. First, a day-long episode is dominated by the stored energy, and harvest plays a supporting role. Second, a controller cannot survive by reacting to a low battery after the fact: by the time the battery looks low, the energy is already gone. It has to allocate the budget in advance. That is the behavior the experiments are designed to test.

2.2.5 A Bound on Average Power

The envelope can be turned into a simple feasibility condition. Over a day of 86,400 s, a node that starts with energy E_0 and harvests at most $E_{\text{h,max}}$ can sustain an average total load of no more than

$$\bar{P}_{\max} = (\, E_0 + E_{\text{h,max}} \,) / 86{,}400 \tag{7}$$

Starting from a full battery with the panel at its 100 mW peak for the entire day, this gives about 1.26 W. With no harvest it falls to 1.16 W. The sustainable average therefore lies between the 0.8 W and 1.6 W lamp levels, and it has to cover the sensing and inference loads as well as the lamp. In other words, the brightest setting can be used only for part of the day, and a controller that survives must share its time between brighter and dimmer settings. The calculation assumes a full initial charge and ignores conversion losses , so it is an upper bound and not a prediction.

2.2.6 Notation

Table 3 collects the symbols used in the rest of the paper.

Table 3: Notation

Symbol	Meaning
E_t, E_{max}, E_{min}	Battery energy at step t, its capacity (100,000 J), and the battery floor enforced by the supervisor
$P_{h,t}$	Harvested power from the PV panel (100 mW peak)
$P_{l,t}$	Total load power: LED, sensing, and inference
T_c	Service continuity: minutes until the battery is first exhausted, capped at 1,440
τ_L	Endurance of a full battery with the lamp held at level L and no harvest
$\bar{P}_{max}$	Highest average total load a node can sustain over a day
s_t, p_t, r_t	Service delivered, energy spent, and reward at step t
γ, α	Discount factor and learning rate

2.3 Controllers

2.3.1 Learned Controllers

Three RL controllers are compared. Tabular Q-learning keeps one action value for each discretized state and action and updates it after every transition [27]. Linear function approximation (Linear FA) replaces the table with a weight vector over 12 engineered features, which lets what is learned in one state carry over to its neighbors. The deep Q-network (DQN) approximates the action values with a small neural network and trains with experience replay and a periodically updated target network [28]. All three bootstrap from the same target,

$$y = r + \gamma \max_{a'} Q(s',a') \quad (8)$$

and differ in how the action value Q is represented. Time of day enters the 12-feature vector as a sine and cosine pair so that the end of one day joins smoothly to the start of the next. Training and architecture details are given in Table 4.

2.3.2 Baseline Policies

Five baselines supply the reference. The Greedy baseline chooses the action with the best immediate service and gives no weight to the energy left for later. a fixed-brightness rule, a state-of-charge threshold rule, a time-schedule rule, and a random policy.] Simple rules of this kind are what many installed lights actually run, so they are the fair comparison for a learned controller.

2.3.3 Safety Supervisor

Between the policy and the lamp sits a supervisor that enforces a battery floor and a minimum service level whatever action the policy proposes [29]. It applies to every policy equally, so the comparison measures the quality of the decisions and not the presence of a safety net.

2.4 Experimental Setup and Outcome Measures

2.4.1 Simulation Environment

The experiments run in a Python simulator built on NumPy, with Matplotlib for plotting. Every policy, learned or rule-based, acts on the same dynamics. An episode covers 1,440 minutes, one full day. The RL controllers were trained in this environment and frozen before evaluation. Table 4 lists the configuration.

Table 4: Node, Episode, and Training Configuration

Item	Value
Battery capacity	100,000 J (about 27.8 Wh)
Solar panel peak output	100 mW
LED power levels	0.2 W, 0.8 W, 1.6 W
Episode length	1,440 min (24 h)
State dimension / engineered features	4 / 12
Action space	27 joint actions (TinyML model, sensing frequency, LED level)
Greedy evaluation episodes	20
Confidence level	95%
Initial battery charge	[fraction of E_{max} at the start of each episode]
Irradiance model	[clear-sky envelope with stochastic cloud attenuation, or measured trace]

2.4.2 Evaluation Protocol

After training, each RL controller was run greedily, with exploration off, for 20 evaluation episodes, and the baselines were run under the same protocol. Repeated episodes are needed because reinforcement learning results can vary from run to run [18]. Differences are judged at the 95% confidence level.

2.4.3 Outcome Measures

Four measures are used. *Service continuity* is T_c from Eq. (5). *Cumulative reward* is the episode sum of Eq. (4). *Accuracy* is the average inference delivered by the selected TinyML models over the episode]. *End-of-day energy margin* is the battery charge remaining at minute 1,440 as a fraction of E_{max}, which shows how close a surviving controller came to the edge.

2.4.4 Implementation Notes

One code base holds the simulator, the learners, and the baselines, so every policy sees the same state construction, action set, and reward. Evaluation episodes come from the same environment definition for every policy, and the learned policies are frozen during evaluation. Differences in outcome therefore reflect the policies, not the conditions they met.

III. RESULTS AND DISCUSSION

3.1 Service Continuity

The clearest result is the one residents would see. DQN kept the node running for all 1,440 minutes. The Greedy baseline emptied the battery at minute 741, which is 51.5% of the day, and left 699 minutes uncovered (Table 5). The gap between those two figures is the practical difference between a street light that lasts the night and one that does not.

Table 5: Service Continuity over a 1,440-Minute Day

Policy	Continuity (min)	Share of the day
DQN	1,440	100%
Greedy baseline	741	51.5%

3.2 Reward

The reward results tell the same story at larger scale. Over the 20 evaluation episodes, the three RL controllers earned between 298.7 and 349.6, and the five baselines between −355.6 and −822.7 (Fig. 3). The weakest RL result is 654.3 reward units above the best baseline. The spread is also telling: the three RL controllers fall within a band of 50.9 units, whereas the baselines are scattered over 467.1. Learned control was not only better but more consistent in outcome across algorithms, which suggests that the benefit comes from the approach and not from a particular architecture. DQN achieved the strongest performance within the RL group.

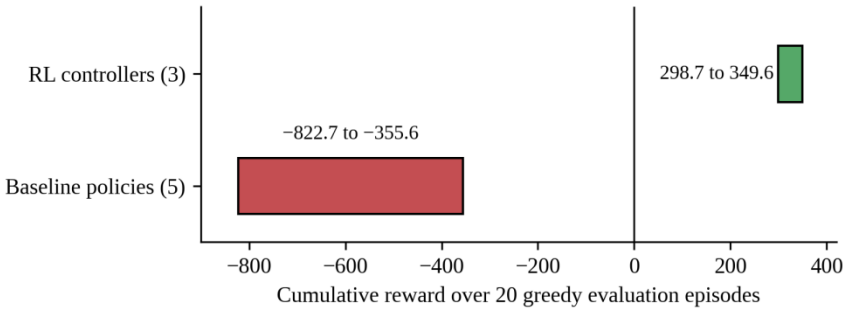


Figure 3: Range of cumulative reward for the RL controllers and the baseline policies.

3.3 Staying Alive Without Losing Accuracy

Survival would be a hollow result if the controllers achieved it by shutting down useful work. They did not. DQN reached 94.9% accuracy while covering the full day, and Linear FA reached 94.7%, only 0.2 percentage points lower. Continuity and accuracy therefore did not have to be traded against each other in this setting, at least among the learned controllers. Table 6 collects the outcomes together.

Table 6: Accuracy and Continuity of the Learned Controllers

Controller	Accuracy (%)	Continuity (min)
DQN	94.9	1,440
Linear FA	94.7	1440
Tabular Q-learning	93.2	1440

3.4 Observed Continuity Against the Energy Envelope

Fig. 4 sets the observed results against the envelope of Section 2.2.4. Holding the lamp at 1.6 W would empty the battery in about 1,042 minutes, so a controller that covers 1,440 minutes must have run the lamp below its brightest level for a meaningful part of the day, assuming it began with a full battery . DQN did exactly that, and it also kept its accuracy high. The Greedy baseline, in contrast, failed at minute 741, earlier than even a constant brightest setting would have allowed. The arithmetic is informative. Emptying a full 100,000 J battery in 741 minutes implies an average net draw of about 2.25 W, which is above the 1.6 W maximum of the lamp alone.

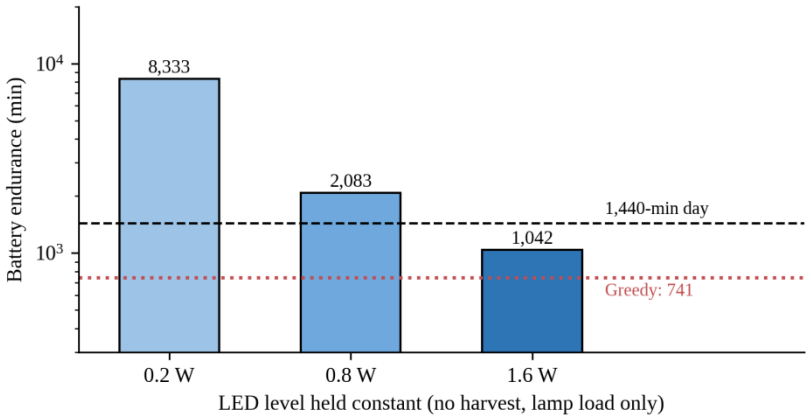


Figure 4: Battery endurance at constant LED levels (no harvest) against the 1,440-minute day and the Greedy result.

3.5 Average Draw Implied by the Results

Continuity results carry information about average power even without the logs. Because DQN covered all 1,440 minutes, its average total draw over the day cannot have exceeded the bound of Eq. (7), about 1.26 W, if it began with a full battery. That figure is below the 1.6 W lamp level, so DQN must have lowered the lamp, the sensing load, or both for much of the day. The Greedy baseline sits at the other extreme. Its implied average draw of about 2.25 W is roughly 1.8 times the sustainable ceiling, which is why it ran out of energy before the day was half over.

3.6 Summary of Outcomes

Table 7: Summary of Principal Outcomes

Outcome	Evidence
Learned control beats every baseline	Cumulative reward 298.7 to 349.6, against −355.6 to −822.7
The best controller covers the whole day	DQN: 1,440 of 1,440 minutes at 94.9% accuracy
A greedy rule fails within the day	Battery exhausted at minute 741 (51.5% of the day)
Linear FA nearly matches DQN on accuracy	94.7% against 94.9%
No controller can light freely all day	Constant 1.6 W empties the battery in about 1,042 min

3.7 Discussion

3.7.1 The Problem Is Allocation

The energy envelope changes how the results should be read. With the lamp alone able to drain the battery in under a day at full brightness and the panel adding at most a few percent of the stored energy, a day of lighting is a budgeting exercise. The question is not whether to spend but when. A greedy rule spends where the reward is immediate, and the battery, which cannot recover within the episode, pays for it later. The learned controllers earned their rewards by doing the opposite. They accepted lower service at some moments in exchange for having energy at others, which is what an RL objective with a long horizon rewards [7].

3.7.2 Why Joint Control May Help

We did not run an ablation that compares joint control with lamp-only control, so the benefit of the joint action space is a hypothesis in this paper and not a finding. The reasoning behind it is straightforward. Dimming is a blunt instrument, because it changes the light the street receives. Slowing the sensing or moving to a lighter model saves energy without touching the lamp. A controller that can choose among these has more ways to stay within budget, and it can shed the cheapest service first. Testing this directly is a priority for further work (Section IV).

3.7.3 Reading Accuracy and Continuity Together

DQN led on both accuracy and continuity, but Linear FA was within 0.2 percentage points on accuracy. For an operator whose main concern is a light that does not fail, the two controllers look much alike on the outcomes measured here. What separates them is cost of deployment.

3.7.4 What an Operator Can Ask of a Controller

Table 8 turns the results into the questions a lighting operator would ask before adopting a controller. It also records where the evidence is still thin.

Table 8: Operator Questions and What This Study Offers

Question	Evidence from this study	Status
Will the light stay on all day?	DQN covered 1,440 of 1,440 min; Greedy failed at minute 741	Shown in simulation
Is service quality kept?	Accuracy of 94.9% (DQN) and 94.7% (Linear FA)	Shown in simulation
Is there a safety backstop?	A supervisor enforces the battery floor [29]	Built into the design
Does it survive bad weather?	Multi-day and low-irradiance stress tests not yet run	Open

3.7.5 Continuity as a Reporting Standard

Continuity is easy to measure, easy to explain to a non-specialist, and tied directly to the thing the service exists to provide. It deserves a place next to reward in studies of RL for edge control, and it is more informative than reward alone, because a reward is a weighted sum that hides how and when a node fails. Reporting it also makes baselines easier to compare across studies, which has been a persistent difficulty [17].

3.7.6 Relation to Energy-Neutral Theory

Energy-neutral operation is defined over the long run [4], and the present study covers one day. The two views are compatible. A controller that cannot survive a single day cannot be energy-neutral over a month, so day-long continuity is a necessary condition, though not a sufficient one. The energy envelope shows that, in this configuration, neutrality over longer horizons would depend on the panel being large enough to replace what each night removes, and that is a sizing question as much as a control question. A controller can only allocate what the hardware provides.

3.7.7 Why Continuity Alone Is Not Enough

A controller can achieve perfect continuity by doing very little. A lamp held at its lowest setting with sensing switched to a minimum would cover the day with ease and serve nobody well. For that reason continuity cannot stand alone, and the study reports it together with accuracy and reward. The combination is what gives the DQN result its weight: the node stayed alive and kept its accuracy at 94.9%. Any future benchmark for solar lighting controllers should pair a survival measure with a service-quality measure and be explicit about how each is defined.

3.7.8 Scope of the Findings

The numbers belong to one simulated node with the parameters of Table 4. Other lamps, panels, and batteries will move the energy envelope, and with it the margin a controller has to work with. A larger panel or battery would relax the allocation problem, and a brighter lamp or a longer night would tighten it. What the study is meant to transfer is the method: compute the envelope first, then ask whether a controller allocates within it. Applying that to other lighting contexts, such as rural roads or campus paths, would only need the parameters of the new node and a harvest trace for its location.

IV. CONCLUSION AND FUTURE WORK

This paper judged learned controllers for a solar street-lighting node by the outcome that matters most to the people who rely on the lamp: whether it stays on. Controlling the TinyML model, the sensing frequency, and the LED brightness jointly through 27 actions, all three RL controllers beat all five baselines on cumulative reward, with scores of 298.7 to 349.6 against −355.6 to −822.7. DQN covered the full 1,440-minute day at 94.9% accuracy, Linear FA reached 94.7%, and the Greedy baseline emptied the battery at minute 741. An energy-envelope calculation from the stated node parameters shows why the task is hard: a lamp held at full brightness would drain the battery in under 1,050 minutes, so surviving the day depends on allocating energy in advance. Learned controllers did that, and a greedy rule did not.

Table 9 sets out the follow-up experiments in the order in which we would run them.

Table 9: Planned Follow-Up Experiments

Experiment	What is varied	What it checks	Primary measure
Low starting charge	Initial battery level	Recovery from a poor start	Continuity
Consecutive cloudy days	Irradiance over several days	Energy-neutral behavior	Continuity, margin
Degraded panel	Panel output and soiling	Robustness to harvest loss	Continuity
Reward-weight sweep	Weights in the reward	Sensitivity of behavior	Reward, continuity
Feature ablation	Subsets of the 12 features	Which features matter	Reward, accuracy
Joint versus lamp-only control	Action space	Value of joint control	Continuity, accuracy
Hardware-in-the-loop	Policy on target MCU	Memory, timing, energy	Measured cost

Several steps would strengthen the evidence. Stress tests with low starting charge, consecutive cloudy days, and reduced panel output would show how robust continuity is. Multi-day episodes would test energy-neutral operation proper. A reward-weight sensitivity study and a feature ablation would show what the results depend on, and an ablation comparing joint with lamp-only control would test the argument for the joint action space. Hardware-in-the-loop validation and a field pilot on a real luminaire would close the gap between simulation and deployment.

V. ACKNOWLEDGEMENTS

The author sincerely acknowledges the guidance, constructive criticisms, and valuable contributions of the research supervisor throughout the development of this study. Special appreciation is extended to the Department of Electrical and Electronic Engineering, University of Uyo, for providing the academic environment, institutional support, and intellectual resources that contributed to the successful completion of this research.The author also appreciates colleagues and fellow researchers whose scholarly contributions provided valuable insights and informed the theoretical and methodological foundations of this study.

REFERENCES

[1]. A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of Things for smart cities," *IEEE Internet Things J.*, vol. 1, no. 1, pp. 22–32, Feb. 2014.
[2]. W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.
[3]. S. Sudevalayam and P. Kulkarni, "Energy harvesting sensor nodes: Survey and implications," *IEEE Commun. Surveys Tuts.*, vol. 13, no. 3, pp. 443–461, 3rd Quart. 2011.
[4]. A. Kansal, J. Hsu, S. Zahedi, and M. B. Srivastava, "Power management in energy harvesting sensor networks," *ACM Trans. Embedded Comput. Syst.*, vol. 6, no. 4, Art. 32, 2007.
[5]. J. Hsu, S. Zahedi, A. Kansal, M. Srivastava, and V. Raghunathan, "Adaptive duty cycling for energy harvesting systems," in *Proc. Int. Symp. Low Power Electron. Design (ISLPED)*, 2006, pp. 180–185.
[6]. C. Moser, D. Brunelli, L. Thiele, and L. Benini, "Real-time scheduling for energy harvesting sensor nodes," *Real-Time Syst.*, vol. 37, no. 3, pp. 233–260, 2007.
[7]. R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.
[8]. R. C. Hsu, C.-T. Liu, and H.-L. Wang, "A reinforcement learning-based ToD provisioning dynamic power management for sustainable operation of energy harvesting wireless sensor node," *IEEE Trans. Emerg. Topics Comput.*, vol. 2, no. 2, pp. 181–191, Jun. 2014.
[9]. S. Shresthamali, M. Kondo, and H. Nakamura, "Adaptive power management in solar energy harvesting sensor node using reinforcement learning," *ACM Trans. Embedded Comput. Syst.*, vol. 16, no. 5s, Art. 181, 2017.
[10]. F. A. Aoudia, M. Gautier, and O. Berder, "RLMan: An energy manager based on reinforcement learning for energy harvesting wireless sensor networks," *IEEE Trans. Green Commun. Netw.*, vol. 2, no. 2, pp. 408–417, Jun. 2018.
[11]. F. Fraternali, B. Balaji, D. Sengupta, D. Hong, and R. K. Gupta, "ACES: Automatic configuration of energy harvesting sensors with reinforcement learning," *ACM Trans. Sens. Netw.*, vol. 16, no. 4, Art. 36, 2020.
[12]. M. Asif *et al.*, "Adaptive control of streetlights using deep learning for the optimization of energy consumption during late hours," *Energies*, vol. 15, no. 17, Art. 6337, 2022.
[13]. J. R. Piorno, C. Bergonzini, D. Atienza, and T. S. Rosing, "Prediction and management in energy harvested wireless sensor nodes," in *Proc. Int. Conf. Wireless Commun., Veh. Technol., Inf. Theory Aerosp. Electron. Syst. Technol. (Wireless VITAE)*, 2009, pp. 6–10.
[14]. E. F. Camacho and C. Bordons, *Model Predictive Control*, 2nd ed. London, U.K.: Springer, 2007.
[15]. F. Fraternali *et al.*, "Scaling configuration of energy harvesting sensors with reinforcement learning," in *Proc. Int. Workshop Energy Harvesting Energy-Neutral Sens. Syst. (ENSsys)*, Shenzhen, China, Nov. 2018 (arXiv:1811.11259).
[16]. U. Y. Ogras *et al.*, "tinyMAN: Lightweight energy manager using reinforcement learning for energy harvesting wearable IoT devices," in *Proc. tinyML Res. Symp.*, 2022.
[17]. G. Dulac-Arnold *et al.*, "Challenges of real-world reinforcement learning: Definitions, benchmarks and analysis," *Mach. Learn.*, vol. 110, no. 9, pp. 2419–2468, 2021.
[18]. P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, "Deep reinforcement learning that matters," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 3207–3214.
[19]. R. David *et al.*, "TensorFlow Lite Micro: Embedded machine learning for TinyML systems," in *Proc. Mach. Learn. Syst. (MLSys)*, vol. 3, 2021, pp. 800–811.
[20]. L. Lai, N. Suda, and V. Chandra, "CMSIS-NN: Efficient neural network kernels for Arm Cortex-M CPUs," arXiv:1801.06601, 2018.

[21]. P. Warden and D. Situnayake, *TinyML: Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers*. Sebastopol, CA, USA: O'Reilly Media, 2019.
[22]. S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding," in *Proc. ICLR*, 2016.
[23]. B. Jacob *et al.*, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2018, pp. 2704–2713.
[24]. V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proc. IEEE*, vol. 105, no. 12, pp. 2295–2329, Dec. 2017.
[25]. M. Horowitz, "Computing's energy problem (and what we can do about it)," in *Proc. IEEE Int. Solid-State Circuits Conf. (ISSCC)*, 2014, pp. 10–14.
[26]. J. García and F. Fernández, "A comprehensive survey on safe reinforcement learning," *J. Mach. Learn. Res.*, vol. 16, pp. 1437–1480, 2015.
[27]. C. J. C. H. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, no. 3–4, pp. 279–292, 1992.
[28]. V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
[29]. M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu, "Safe reinforcement learning via shielding," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 2669–2678.