ECE 1508 Report: Room Temperature Control with Sensor Noise

Weilin Fu, Yupeng Zhang

August 2024

1 Introduction

This project explores the efficacy of reinforcement learning (RL) algorithms in maintaining a stable indoor temperature despite fluctuating external conditions and sensor inaccuracies. This project addresses the challenge of achieving a constant set-point temperature in a room, considering the dynamic nature of external temperatures and the added complexity of sensor noise, which simulates real-world measurement inaccuracies. This project addresses several key research questions. Firstly, it explores the impact of sensor noise on the learning process and performance of non-deep and deep RL agents. It examines how inaccuracies in temperature readings affect the algorithms' ability to achieve and maintain the desired set-point temperature. Secondly, it assesses the robustness of different RL algorithms to sensor noise, identifying which approach is more resilient and understanding the reasons behind their robustness. Thirdly, it examines the sensitivity of various parameters, such as learning rate, discount factor, and exploration strategy, on the agent's learning efficiency in noisy environments, aiming to optimize the performance of the RL agents. Finally, it compares the performance and stability of deep RL algorithms with traditional non-deep RL methods, providing insights into how deep learning influences the agents' ability to handle noisy, imperfect observations and their applicability in real-world scenarios. Through these investigations, the project seeks to determine the most effective RL strategies for controlling room temperature amidst environmental uncertainties and sensor noise.

2 Method

The project environment meticulously simulates the dynamic interplay between internal and external temperatures within a room, integrating sensor noise to reflect real-world measurement inaccuracies accurately. The setup encompasses the precise definition of environment dynamics, the observation space, and the reward function, ensuring a comprehensive and realistic simulation.

2.1 Environment Dynamics

The dynamics of the environment are described by the following equations:

2.1.1 Internal Temperature Update

$$T_{int}(t+1) = T_{int}(t) + s \cdot (T_{ext}(t) - T_{int}(t)) + a(t) \tag{1}$$

where:

- $T_{int}(t)$ [Celsius]: The internal room temperature at time t.
- $s = 0.1$: The insulation factor for the room.
- $T_{ext}(t)$ [Celsius]: The external temperature at time t.
- $a(t)$ [Celsius]: The action applied by the air conditioning system at time t, which can be either $+0.1$ or -0.1 Celsius.

2.1.2 External Temperature Variation

$$T_{ext}(t) = 20 + 10 \sin\left(\frac{2\pi t}{24 \times 60}\right) \tag{2}$$

2.1.3 Initial Internal Temperature

$$T_{int}(0) = 15 \text{ Celsius} \tag{3}$$

The simulation runs for a total duration of 1440 minutes (24 hours).

2.1.4 Revision on Action Space

The action space has been revised to include a wider range of actions: $[-2, -1, -0.5, -0.1, 0.0, 0.1, 0.5, 1, 2]$. This adjustment was made to better handle the peak and bottom extremes of the external temperature variations. The original action space, which included more limited action values, posed challenges in effectively controlling the internal temperature, particularly during periods of significant external temperature fluctuations. By expanding the range of possible actions, the RL agent can now make more substantial adjustments to the air conditioning system, ensuring better stability and control over the internal temperature, even under extreme external conditions. **Additional justification has been included in the Appendix**.

2.2 Observation Space

The RL agent receives observations that include noisy sensor readings of both the internal and external temperatures:

$$S(t) = [\hat{T}_{int}(t), \hat{T}_{ext}(t)] \tag{4}$$

where the noisy sensor readings are given by:

$$\hat{T}_{int}(t) = T_{int}(t) + \epsilon_{int} \tag{5}$$

$$\hat{T}_{ext}(t) = T_{ext}(t) + \epsilon_{ext} \tag{6}$$

The sensor noise ϵ_{int} and ϵ_{ext} are modeled as Gaussian distributions:

$$\epsilon_{int} \sim \mathcal{N}(0, \sigma^2_{int}) \tag{7}$$

$$\epsilon_{ext} \sim \mathcal{N}(0, \sigma^2_{ext}) \tag{8}$$

This setup aims to create a realistic environment to evaluate the performance of reinforcement learning agents under conditions of sensor inaccuracies and dynamic temperature changes.

2.3 Reward Function

The reward function is designed to ensure that the internal temperature remains as close to the set-point temperature as possible, minimizing the impact of external temperature fluctuations and sensor noise. The reward function is defined as follows:

$$R(t) = -|T_{int}(t) - T_{SP}| \tag{9}$$

where:

- $R(t)$: The reward at time t.
- $T_{int}(t)$: The internal room temperature at time t.
- T_{SP}: The temperature set-point.

The primary purpose of this reward function is to control the internal temperature such that it remains stable around the set-point temperature. By penalizing the absolute deviation of the internal temperature from the set-point, the function encourages the RL agent to minimize the influence of external temperature variations and sensor noise on the internal temperature.

This approach ensures that the internal temperature stays within a desirable range, providing a stable and comfortable environment despite the presence of external disturbances and measurement inaccuracies.

2.4 Non-Deep RL Method - Q-Learning

Q-Learning algorithm has been employed to develop a traditional reinforcement learning agent aimed at controlling room temperature under conditions of sensor noise. The Q-Learning algorithm is chosen for its effectiveness in environments with discrete state and action spaces, which aligns with our discretized temperature control problem.

2.4.1 Initialization

The agent is initialized with key parameters:

- **Learning Rate (α)**: Determines the step size for updating Q-values.
- **Discount Factor (γ)**: Balances immediate and future rewards.
- **Exploration Rate (ϵ)**: Governs the probability of choosing random actions to explore the environment.
- **Exploration Decay**: Reduces the exploration rate over time.
- **Bins**: Discretize the continuous state space, enabling tabular Q-learning.

2.4.2 State Discretization

The continuous states are converted into discrete bins, facilitating the use of a finite Q-table. This transformation is achieved through a discretization function.

2.4.3 Action Selection

The agent employs an ϵ-greedy policy:

- **Exploration**: With probability ϵ, the agent selects a random action.
- **Exploitation**: With probability $1 - \epsilon$, the agent selects the action with the highest Q-value for the current state.

2.4.4 Q-Value Update

The core of the Q-Learning algorithm is the Temporal Difference (TD) learning rule. For a given state-action-reward-next state tuple (s, a, r, s'):

- **TD Target Calculation**:
  $$\text{TD Target} = r + \gamma \max_a Q(s', a)$$
  where r is the received reward, γ is the discount factor, and $\max_a Q(s', a)$ is the maximum Q-value for the next state.
- **TD Error**:
  $$\text{TD Error} = \text{TD Target} - Q(s, a)$$
- **Q-Value Update**:
  $$Q(s, a) \leftarrow Q(s, a) + \alpha \times \text{TD Error}$$
  The current Q-value is adjusted towards the TD target, scaled by the learning rate.

By systematically updating the Q-values based on state transitions and rewards, the Q-Learning algorithm enables the agent to iteratively improve its policy, leading to optimal decision-making over time.

2.5 Deep RL Method - DQN

The Deep Q-Network (DQN) algorithm is utilized to develop a reinforcement learning agent capable of controlling room temperature in the presence of sensor noise. DQN extends the Q-Learning algorithm by incorporating deep neural networks to approximate the Q-values, allowing it to handle environments with high-dimensional state spaces.

2.5.1 Neural Network Architecture

The DQN consists of a neural network with three fully connected layers:

- **Input Layer**: The input dimension equals to the size of the state space.
- **Hidden Layers**: Two hidden layers, each with 24 neurons, use the ReLU activation function to introduce non-linearity.
- **Output Layer**: The output dimension equals to the number of possible actions, providing the Q-values for each action given the current state.

2.5.2 Q-Value Update

The agent employs two neural networks, the policy network is updated every step, while the target network is updated periodically to stabilize learning.

For a given batch of transitions (s, a, r, s', d):

- **Current Q-Values**:
$$Q(s, a) = \text{policy\_net}(s)[a]$$
- **Target Q-Values** (using the target network):
$$Q_{\text{target}} = r + \gamma \max_a Q_{\text{target\_net}}(s', a) \cdot (1 - d)$$
where γ is the discount factor, and d is a binary indicator of whether the episode has terminated.
- **Loss Calculation**: The Mean Squared Error (MSE) loss between the current Q-values and the target Q-values:
$$\text{Loss} = \text{MSE}(Q(s, a), Q_{\text{target}})$$
- **Optimization**: The policy network is optimized by backpropagating the loss and updating the network parameters using the Adam optimizer.

2.5.3 Experience Replay

The agent uses an experience replay buffer to improve sample efficiency and decorate training data. Transitions are stored in the buffer and sampled in mini-batches during training. This technique stabilizes training by breaking the temporal correlations between consecutive transitions.

2.5.4 Exploration vs. Exploitation

The agent uses an ϵ-greedy policy to balance exploration and exploitation:

- **Exploration**: With probability ϵ, a random action is selected.
- **Exploitation**: With probability $1 - \epsilon$, the action with the highest Q-value from the policy network is selected.

The exploration rate ϵ decays over time to shift the focus from exploration to exploitation as the agent learns.

2.5.5 Target Network Update

To enhance stability, the target network is periodically updated to match the policy network. This decoupling of the target from the policy network reduces the risk of divergence during training.

By employing these mechanisms, the DQN agent is capable of learning effective control policies for maintaining the room temperature at the setpoint, despite the presence of sensor noise and varying external conditions.

3 Result

3.1 Impact of Noise

To explore the impact of sensor noise on the learning process and performance of non-deep and deep RL agents. Experiments have been separately conducted on traditional and deep RL models. With all other conditions remaining the same, three experiments were conducted for each model. Experiment 1: scale σ^2_{ext}, Experiment 2: scale σ^2_{int}, and Experiment 3: scale σ^2_{int} and σ^2_{ext}.

3.1.1 Q-Learning (Traditional RL)

Experiment 1: Varying External Noise (σ^2_{ext})

Experiment	Train Reward	Test Reward
Ext 0.0, Int 0.5	-3832.38	-2703.23
Ext 0.2, Int 0.5	-3910.09	-2870.68
Ext 0.4, Int 0.5	-3811.98	-2781.38
Ext 0.5, Int 0.5	-2725.23	-2774.53
Ext 0.6, Int 0.5	-3928.76	-3070.66
Ext 0.8, Int 0.5	-3776.23	-2810.17
Ext 1.0, Int 0.5	-3752.21	-2638.81
Ext 10.0, Int 0.5	-3989.28	-2725.80

Table 1: Experiment 1: scale σ^2_{ext}, Q-Learning trained 1000 Episodes

The highest training reward is observed for $\sigma^2_{ext} = 0.5$ (-2725.23), indicating better learning efficiency at this noise level. The testing rewards follow a similar trend, with the best performance at $\sigma^2_{ext} = 0.5$ (-2774.53). Higher noise levels ($\sigma^2_{ext} = 10.0$) result in poorer training (-3989.28) and testing rewards (-2725.80), suggesting that very high external noise degrades the agent's performance. However, the lowest external noise ($\sigma^2_{ext} = 0.0$) does not yield the best rewards, implying that a moderate level of noise might aid learning.

Experiment 2: Varying Internal Noise (σ^2_{int})

Experiment	Train Reward	Test Reward
Ext 0.5, Int 0.0	-3994.18	-2899.08
Ext 0.5, Int 0.2	-3916.12	-2950.15
Ext 0.5, Int 0.4	-3894.52	-2836.07
Ext 0.5, Int 0.5	-2725.23	-2774.53
Ext 0.5, Int 0.6	-3815.00	-2577.15
Ext 0.5, Int 0.8	-3759.98	-2548.65
Ext 0.5, Int 1.0	-3743.78	-2420.20
Ext 0.5, Int 10.0	-4713.80	-2481.10

Table 2: Experiment 2: scale σ^2_{int}, Q-Learning trained 1000 Episodes

The highest training reward is observed for $\sigma^2_{int} = 0.5$ (-2725.23), indicating optimal performance at this noise level. The best testing reward is at $\sigma^2_{int} = 1.0$ (-2420.20), reinforcing the idea that moderate internal noise improves the agent's performance. Extremely high internal noise ($\sigma^2_{int} = 10.0$) results in the worst training (-4713.80) and testing rewards (-2481.10), indicating significant performance degradation. As internal noise increases, the testing rewards tend to improve slightly up to a point ($\sigma^2_{int} = 1.0$) but worsen at very high noise levels.

Experiment 3: Varying Both Internal and External Noise (σ^2_{int} and σ^2_{ext})

Experiment	Train Reward	Test Reward
Ext 0.0, Int 0.0	-3947.70	-2959.21
Ext 0.2, Int 0.2	-3949.62	-2934.24
Ext 0.4, Int 0.4	-3877.08	-2851.39
Ext 0.5, Int 0.5	-2725.23	-2774.53
Ext 0.6, Int 0.6	-3830.84	-2632.29
Ext 0.8, Int 0.8	-3727.19	-2580.36
Ext 1.0, Int 1.0	-3615.11	-2454.80
Ext 10.0, Int 10.0	-6293.28	-3429.61

Table 3: Experiment 3: scale σ^2_{int} and σ^2_{ext}, Q-Learning trained 1000 Episodes

The best training reward is observed for moderate noise levels ($\sigma^2_{int} = \sigma^2_{ext} = 0.5$) (-2725.23). Testing rewards are optimal at the same noise levels ($\sigma^2_{int} = \sigma^2_{ext} = 0.5$) (-2774.53). Extremely high noise levels ($\sigma^2_{int} = \sigma^2_{ext} = 10.0$) result in significantly worse training (-6293.28) and testing rewards (-3429.61), emphasizing the detrimental effects of excessive noise. The results suggest that a balance of internal and external noise can facilitate better learning and performance.

3.1.2 DQN (Deep RL)

Experiment 1: Varying External Noise (σ^2_{ext})

Experiment	Train Reward	Test Reward
Ext 0.0, Int 0.5	-606.19	-619.62
Ext 0.2, Int 0.5	-855.53	-869.49
Ext 0.4, Int 0.5	-554.98	-552.64
Ext 0.5, Int 0.5	-730.70	-728.80
Ext 0.6, Int 0.5	-536.87	-523.53
Ext 0.8, Int 0.5	-785.44	-765.49
Ext 1.0, Int 0.5	-557.63	-547.12
Ext 10.0, Int 0.5	-949.89	-960.17

Table 4: Experiment 1: scale σ^2_{ext}, DQN trained 100 Episodes

The highest training reward is observed for $\sigma^2_{ext} = 0.6$ (-536.87), indicating better learning efficiency at this noise level. The best testing reward is also at $\sigma^2_{ext} = 0.6$ (-523.53), suggesting that the agent performs well with moderate external noise. Extremely high noise levels ($\sigma^2_{ext} = 10.0$) result in poorer performance, both in training (-949.89) and testing (-960.17). The results suggest that moderate noise levels aid in better policy learning and generalization.

Experiment 2: Varying Internal Noise (σ^2_{int})

Experiment	Train Reward	Test Reward
Ext 0.5, Int 0.0	-773.76	-769.28
Ext 0.5, Int 0.2	-596.05	-602.13
Ext 0.5, Int 0.4	-1006.62	-988.30
Ext 0.5, Int 0.5	-730.70	-728.80
Ext 0.5, Int 0.6	-619.17	-611.27
Ext 0.5, Int 0.8	-893.14	-896.57
Ext 0.5, Int 1.0	-765.93	-813.21
Ext 0.5, Int 10.0	-1410.31	-1555.45

Table 5: Experiment 2: scale σ^2_{int}, DQN trained 100 Episodes

The highest training reward is observed for $\sigma^2_{int} = 0.6$ (-619.17), indicating better learning efficiency. The best testing reward is also at $\sigma^2_{int} = 0.6$ (-611.27), suggesting that moderate internal noise improves the agent's performance. Very high internal noise ($\sigma^2_{int} = 10.0$) results in significantly worse training (-1410.31) and testing rewards (-1555.45), showing the negative impact of excessive noise. These results suggest that moderate internal noise levels facilitate better policy learning and generalization.

Experiment 3: Varying Both Internal and External Noise (σ^2_{int} and σ^2_{ext})

Experiment	Train Reward	Test Reward
Ext 0.0, Int 0.0	-420.46	-431.86
Ext 0.2, Int 0.2	-505.03	-499.05
Ext 0.4, Int 0.4	-562.29	-559.10
Ext 0.5, Int 0.5	-730.70	-728.80
Ext 0.6, Int 0.6	-714.59	-709.96
Ext 0.8, Int 0.8	-623.33	-622.63
Ext 1.0, Int 1.0	-731.84	-734.88
Ext 10.0, Int 10.0	-3103.06	-3290.65

Table 6: Experiment 3: scale σ^2_{int} and σ^2_{ext}, DQN trained 100 Episodes

The best training reward is observed for low noise levels ($\sigma^2_{int} = \sigma^2_{ext} = 0.0$) (-420.46). Testing rewards are also optimal at low noise levels (-431.86). High noise levels ($\sigma^2_{int} = \sigma^2_{ext} = 10.0$) result in the worst training (-3103.06) and testing rewards (-3290.65), indicating substantial performance degradation. These findings suggest that while low noise levels facilitate the best performance, moderate noise levels also balance learning efficiency and policy generalization.

3.1.3 Insight and Comparison

In Q-learning agent, moderate noise levels ($\sigma^2_{int} = \sigma^2_{ext} = 0.5$) yield the best performance, suggesting that a certain amount of noise can help the learning process.

In DQN agent, both low and moderate noise levels facilitate better learning and performance. Excessive noise significantly degrades performance in both algorithms.

Both Q-Learning and DQN agents perform optimally at moderate noise levels, but DQN appears more resilient to noise variations and achieves higher rewards across experiments.

3.2 Robustness of Algorithms

Based on our experimental results, comparing the robustness of Q-Learning to DQN under varying noise conditions reveals some interesting insights.

For Q-Learning (Traditional RL), the agent performs best at a moderate external noise level ($\sigma^2_{ext} = 0.5$), with both training and testing rewards peaking at this point. Higher noise levels ($\sigma^2_{ext} = 10.0$) result in significant performance degradation, indicating that Q-Learning struggles with excessive external noise. Interestingly, the lowest external noise ($\sigma^2_{ext} = 0.0$) does not yield the best rewards, suggesting that a moderate amount of noise might be beneficial for learning. Similarly, when varying internal noise (σ^2_{int}), the Q-Learning agent performs best with moderate internal noise ($\sigma^2_{int} = 0.5$). Extremely high internal noise ($\sigma^2_{int} = 10.0$) significantly degrades performance, indicating sensitivity to high noise levels. When both internal and external noise are varied simultaneously ($\sigma^2_{int} = \sigma^2_{ext}$), the agent shows optimal performance at moderate noise levels ($\sigma^2_{int} = \sigma^2_{ext} = 0.5$), with performance deteriorating at both low and high noise levels. This indicates that Q-Learning benefits from a balanced noise level.

In contrast, the DQN (Deep RL) agent performs best at moderate noise levels ($\sigma^2_{ext} = 0.6$), achieving higher training and testing rewards compared to other noise levels. High external noise ($\sigma^2_{ext} = 10.0$) leads to poorer performance, although the degradation is less severe compared to Q-Learning. When varying internal noise (σ^2_{int}), the DQN agent performs optimally at moderate internal noise levels ($\sigma^2_{int} = 0.6$), with significant performance drops at very high noise levels ($\sigma^2_{int} = 10.0$). This pattern is similar to the external noise experiment, indicating a preference for moderate noise. When both internal and external noise are varied simultaneously ($\sigma^2_{int} = \sigma^2_{ext}$), the DQN agent performs best at low noise levels ($\sigma^2_{int} = \sigma^2_{ext} = 0.0$), but also shows good performance at moderate levels. Performance significantly degrades at high noise levels ($\sigma^2_{int} = \sigma^2_{ext} = 10.0$).

Comparing the two, both Q-Learning and DQN agents perform optimally at moderate noise levels. However, DQN achieves higher rewards across the board, indicating better learning efficiency and robustness. DQN is more resilient to noise variations, showing less severe performance degradation at higher noise levels compared to Q-Learning. Additionally, DQN consistently achieves higher rewards in both training and testing phases, suggesting superior learning capabilities.

In conclusion, while both Q-Learning and DQN benefit from moderate noise levels, DQN demonstrates greater robustness and higher overall performance. This resilience to noise and enhanced learning efficiency make DQN a more robust choice for environments with varying noise conditions.

3.3 Parameter Sensitivity

3.3.1 Q-Learning (Traditional RL)

Category	**0.001**	**0.01**	**0.1**	**0.999**	**0.99**	**0.90**	**ER 1.0**	**0.5**	**ED 0.999**	**0.995**
Reward	-9663	**-2069**	-3790	-7750	-2200	**-2138**	-2138	**-2080**	-3347	**-2139**

Table 7: Experiment: Parameter Sensitivity. Parameter included: Learning Rate (first three columns), Discount Factors (second three columns), Exploration Rate, and Exploration Decay. The second column is reward from the experiment, highest reward from specific experiment is marked in bold. (**Round Up** to the nearest) Each experiment is conducted on Q-Learning trained 1000 Episodes.

3.3.2 DQN (Deep RL)

Category	**LR 0.0001**	**0.001**	**0.01**	**Gamma 0.9**	**0.95**	**0.99**	**ED 0.99**	**0.995**	**0.999**
Reward	-2201	**-598**	-815	**-482**	-660	-864	**-616**	-664	-900

Table 8: Experiment: Parameter Sensitivity. Parameter included: Learning Rate, Gamma, and Epsilon Decay. The second column is reward from the experiment, highest reward from specific experiment is marked in bold. (**Round Up** to the nearest) Each experiment is conducted on DQN trained 100 Episodes.

3.3.3 Insight and Comparison

The parameter sensitivity tests for Q-Learning and DQN reveal the most effective parameter configurations for each method.

For Q-Learning, the best learning rate is 0.01, achieving a reward of -2069, significantly better than other learning rates tested. Among the discount factors, 0.90 yields the best result with a reward of -2138. Regarding the exploration rate and decay, the optimal values are 0.5 and 0.995, respectively, with rewards of -2080 and -2139.

In DQN, the optimal learning rate is 0.001, achieving the highest reward of -598. The best gamma value is 0.9, resulting in a reward of -482. For epsilon decay, 0.99 provides the best performance with a reward of -616.

In summary, while both Q-Learning and DQN benefit from carefully tuned parameters, DQN shows a greater sensitivity to these parameters, achieving higher overall rewards. The best set of parameters for Q-Learning is a learning rate of 0.01, a discount factor of 0.90, and an exploration rate of 0.5 with an exploration decay of 0.995. For DQN, the best set of parameters includes a learning rate of 0.001, a gamma of 0.9, and an epsilon decay of 0.99.

3.4 Comparison of Deep vs. Non-Deep RL

3.4.1 Comparative Analysis

- **Performance under Noise**: Both Q-Learning and DQN agents perform optimally at moderate noise levels. However, DQN achieves higher rewards across the board, indicating better learning efficiency and robustness. DQN is more resilient to noise variations, showing less severe performance degradation at higher noise levels compared to Q-Learning.

- **Learning Efficiency**: DQN consistently achieves higher rewards in both training and testing phases, suggesting superior learning capabilities compared to Q-Learning.

- **Parameter Sensitivity**: Both Q-Learning and DQN benefit from carefully tuned parameters, but DQN shows greater sensitivity to these parameters, achieving higher overall rewards.

3.4.2 Justification

The experimental results indicate that DQN outperforms Q-Learning in terms of robustness to noise and overall learning efficiency. DQN's ability to handle noise variations more effectively and achieve higher rewards demonstrates its superiority in dynamic environments. The higher sensitivity of DQN to parameter tuning allows for more precise optimization, further enhancing its performance. These findings justify the preference for DQN over Q-Learning in scenarios where environmental conditions are unpredictable and optimal performance is critical.

4 Conclusion

In our comparison of Deep Reinforcement Learning (DQN) and Non-Deep Reinforcement Learning (Q-Learning), we observed notable differences in their performance, robustness to noise, and sensitivity to parameter tuning.

Both Q-Learning and DQN agents perform optimally at moderate noise levels. However, DQN consistently achieves higher rewards across the board, indicating superior learning efficiency and robustness. Specifically, DQN demonstrates greater resilience to noise variations, showing less severe performance degradation at higher noise levels compared to Q-Learning. This suggests that DQN is better suited for dynamic environments where noise levels can fluctuate unpredictably.

In terms of learning efficiency, DQN outperforms Q-Learning by consistently achieving higher rewards in both training and testing phases. This superior performance highlights the advanced learning capabilities of DQN, making it a more effective solution for complex tasks requiring robust decision-making.

When it comes to parameter sensitivity, both Q-Learning and DQN benefit from carefully tuned parameters. However, DQN shows greater sensitivity to these parameters, which translates into higher overall rewards when the optimal parameters are used. This increased sensitivity allows for more precise optimization, further enhancing DQN's performance.

After conducting the parameter sensitivity test, we trained and evaluated the DQN agent using the best hyperparameters: a learning rate of 0.001, epsilon decay of 0.99, and gamma of 0.9. The results were impressive, with the DQN agent achieving an average reward of -449.34 over 100 episodes and a total reward of -469.02. These results indicate that the DQN agent performs exceptionally well under these conditions.

The visual results of the DQN training and evaluation provide further insights:

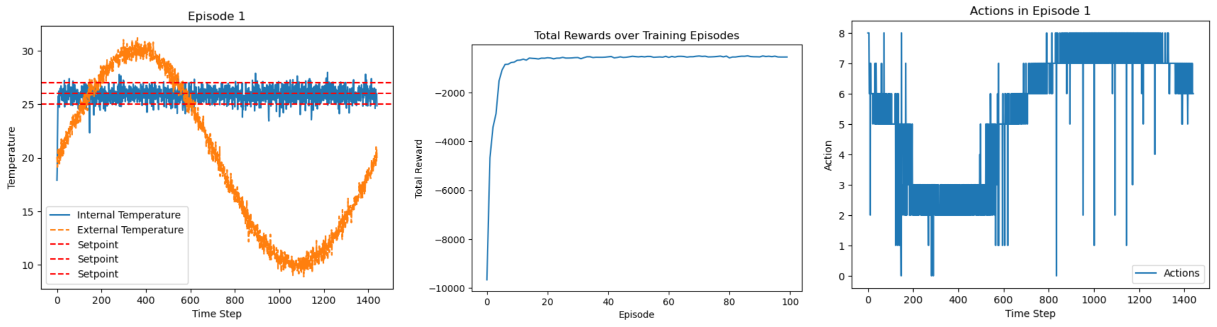

Figure 1: Best DQN Training and Evaluation Results: (a) Internal and External Temperature Tracking, (b) Total Rewards over Training Episodes, (c) Actions Taken in Episode 1

The internal temperature is maintained close to the setpoint despite external temperature fluctuations, indicating effective control by the DQN agent. The learning curve of the DQN agent shows that rewards improve and stabilize over the training episodes, reflecting the agent's learning progress. The actions taken by the DQN agent in Episode 1 to maintain the internal temperature highlight the agent's decision-making process and its ability to adapt to changing conditions. Best DQN's performance at extreme noise level has been included in Appendix. It is clear the Best DQN's performance is better than the original DQN.

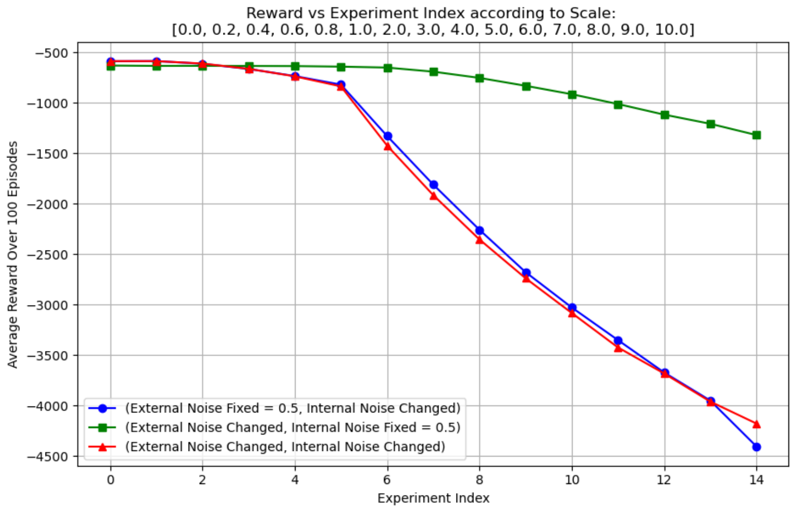

Figure 2: Best DQN Evaluation Result, Trained on Internal and External **Equivalent** at 0.5, Evaluated at a **Changing Scale** of [0.0, 0.2, 0.4, 0.6, 0.8, 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0]

Additionally, we have also conducted experiments to evaluate the Best DQN model trained on internal and external noise levels equivalent at 0.5. The plot illustrates the DQN agent's performance under varying noise conditions. From the plot, the DQN agent fails to set the internal temperature to the setpoint at certain noise levels. Specifically, for the transformation where external noise is fixed at 0.5 and internal noise changes $(0.5, External)$, the agent fails at an internal noise level after 1.0. Similarly, for the transformation where internal noise is fixed at 0.5 and external noise changes $(Internal, 0.5)$, the agent ability smoothly decays.

Additionally, for the transformation where both external and internal noise change ($External, Internal$), the agent fails at a noise level after 1.0 for both internal and external noise. These findings suggest that extreme noise levels significantly impact the DQN agent's ability to maintain the setpoint temperature.

Furthermore, according to the plot, we can observe that when only the external temperature noise is scaling, the agent's ability is less affected, while if internal temperature noise exists or internal and external both exist, the agent's ability is significantly affected. Hence, we can determine the DQN agent is more sensitive to the internal noise compared to the external noise.

This consistent drop across all three transformations highlights that high noise levels, irrespective of their source, adversely affect the agent's performance. Therefore, ensuring noise levels remain within a manageable range is crucial for the DQN agent to function effectively in maintaining the setpoint temperature.

Overall, the experimental results clearly indicate that DQN outperforms Q-Learning in terms of robustness to noise and overall learning efficiency. DQN's ability to handle noise variations more effectively and achieve higher rewards demonstrates its superiority in dynamic environments. The higher sensitivity of DQN to parameter tuning allows for more precise optimization, further enhancing its performance. These findings justify the preference for DQN over Q-Learning in scenarios where environmental conditions are unpredictable and optimal performance is critical.

5 Discussion

In our application of Deep Reinforcement Learning (DQN) for room temperature control, several key limitations emerged that are specific to this task. While the DQN model showed promising results in maintaining the desired temperature setpoint and outperformed the Q-Learning model in terms of robustness and efficiency, the following main drawbacks were identified:

1. **Sensor Noise Sensitivity**: The DQN model's performance is sensitive to sensor noise, which is prevalent in real-world applications. Despite achieving optimal performance under moderate noise levels, the model struggles with high levels of sensor noise, potentially leading to suboptimal temperature control in environments where sensor accuracy cannot be guaranteed.

2. **Environmental Changes**: The DQN model is trained on specific environmental conditions and lacks adaptability to sudden changes in external factors such as weather, occupancy, or unforeseen thermal loads. Adapting to non-stationary environments without retraining can be challenging and resource-intensive.

3. **Energy Efficiency**: The current DQN model focuses on maintaining the setpoint temperature but does not explicitly optimize for energy efficiency. Balancing temperature control with energy consumption is vital for practical implementations, and integrating energy efficiency into the model's objectives would add complexity.

4. **Computational Resources**: Implementing DQN for real-time room temperature control is computationally intensive. The model demands considerable computational power for both training and inference, which may not be feasible for all HVAC systems, especially in residential settings with limited hardware capabilities.

In conclusion, while the DQN model demonstrates strong potential for room temperature control, addressing these main drawbacks is crucial for practical deployment. Enhancing the model's robustness to sensor noise and environmental changes, optimizing for energy efficiency, and ensuring feasible computational resource requirements will be key steps in developing a reliable and efficient DQN-based solution for maintaining optimal indoor climate conditions.

6 Appendix

6.1 Extreme Noise Level on Best DQN

6.1.1 Experiment: Internal Noise 10.0, External Noise 10.0

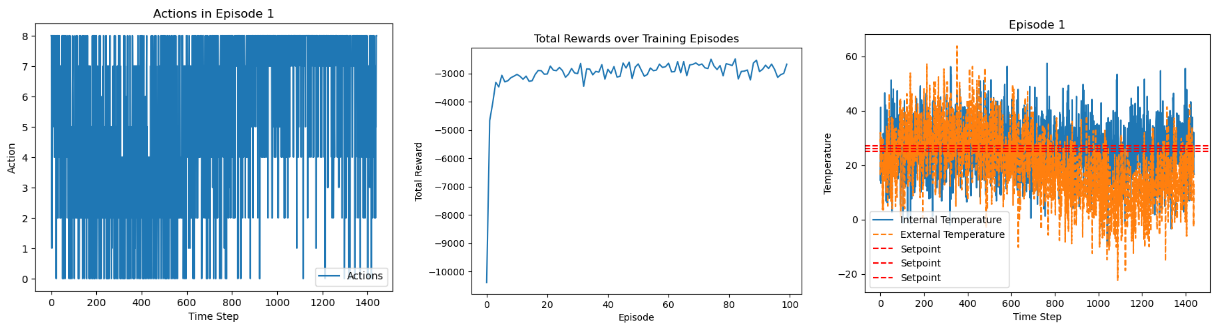

Figure 3: Best DQN Training and Evaluation Results: (a) Internal and External Temperature Tracking, (b) Total Rewards over Training Episodes, (c) Actions Taken in Episode 1

1. Average Reward over 100 episodes: -2717.23
2. Total Reward: -2688.69

6.1.2 Experiment: Internal Noise 0.5, External Noise 10.0

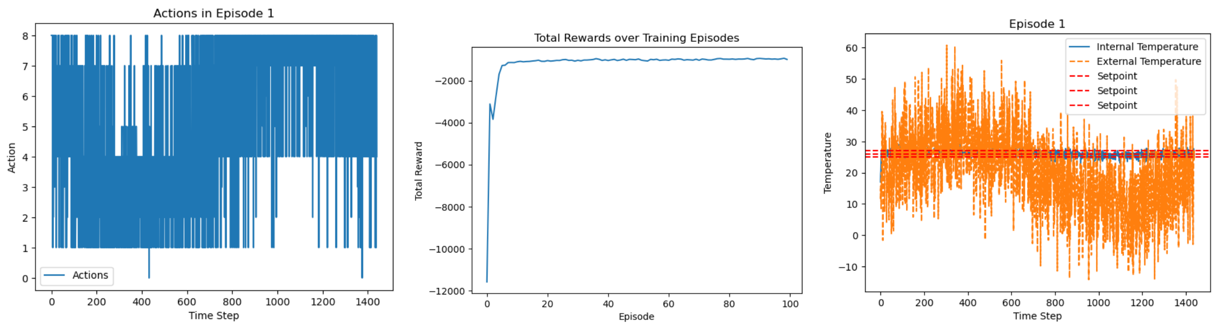

Figure 4: Best DQN Training and Evaluation Results: (a) Internal and External Temperature Tracking, (b) Total Rewards over Training Episodes, (c) Actions Taken in Episode 1

1. Average Reward over 100 episodes: -965.54
2. Total Reward: -954.82

6.1.3 Experiment: Internal Noise 10.0, External Noise 0.5

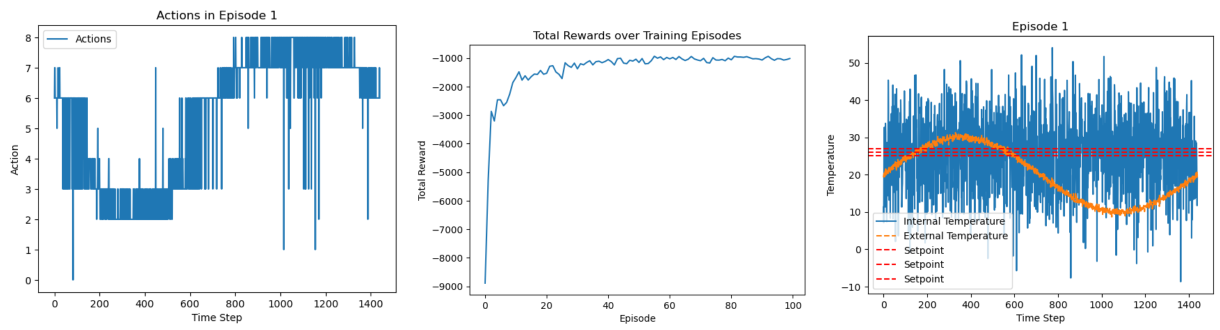

Figure 5: Best DQN Training and Evaluation Results: (a) Internal and External Temperature Tracking, (b) Total Rewards over Training Episodes, (c) Actions Taken in Episode 1

1. Average Reward over 100 episodes: -1043.34
2. Total Reward: -1005.07

6.2 Justification of Modifying Action Space

6.2.1 Critical Example

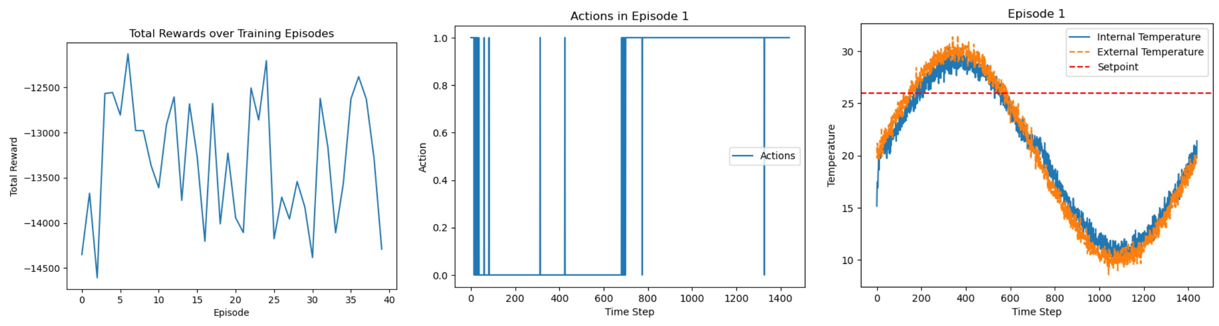

Figure 6: DQN Training and Evaluation Results on Unmodified Environment: (a) Internal and External Temperature Tracking, (b) Total Rewards over Training Episodes, (c) Actions Taken in Episode 1

This is one of the most critical examples that demonstrated the importance of modifying the action value in this project. The agent has conducted nearly all actions correctly; however, the internal temperature movement still proceeds closely with the external temperature.