Deep Reinforcement Learning for Autonomous Water Heater Control

Kadir Amasyali 1,*, Jeffrey Munk 2, Kuldeep Kurte 1, Teja Kuruganti 1 and Helia Zandi 1

Abstract: Electric water heaters represent 14% of the electricity consumption in residential buildings. An average household in the United States (U.S.) spends about USD 400–600 (0.45 ¢/L–0.68 ¢/L) on water heating every year. In this context, water heaters are often considered as a valuable asset for Demand Response (DR) and building energy management system (BEMS) applications. To this end, this study proposes a model-free deep reinforcement learning (RL) approach that aims to minimize the electricity cost of a water heater under a time-of-use (TOU) electricity pricing policy by only using standard DR commands. In this approach, a set of RL agents, with different look ahead periods, were trained using the deep Q-networks (DQN) algorithm and their performance was tested on an unseen pair of price and hot water usage profiles. The testing results showed that the RL agents can help save electricity cost in the range of 19% to 35% compared to the baseline operation without causing any discomfort to end users. Additionally, the RL agents outperformed rule-based and model predictive control (MPC)-based controllers and achieved comparable performance to optimization-based control.

Keywords: deep Q-networks; reinforcement learning; heat pump water heater; demand response; smart grid; machine learning; deep learning

1. Introduction

The use of fossil fuels continues to pose adverse environmental impacts on the ecosystem in terms of global warming and pollution. Renewable energy sources, such as solar, wind, biofuels and hydro, are expected to play a key role in transforming the harmful carbon-intensive energy generation systems into the more sustainable ones [1]. Motivated by this fact, the share of renewable energy sources in electricity generation is increasing [2]. For example, the share of renewables in electricity generation in the U.S. increased from 10.5% to 17.5% over the last ten years [1]. This trend is expected to continue for a long time, because many countries have already set ambitious renewable energy targets for climate action [3]. The rise of renewables, however, presents some new challenges to overcome. Primarily, the electricity generation from renewable energy sources is difficult to control and forecast due to their intermittent and stochastic nature, which may sometimes cause an imbalance between electricity supply and demand, and may consequently affect the stability of power grids. To fully exploit renewable energy sources, the potential imbalances between electricity supply and demand should be avoided. Transition from generation on demand to consumption on demand is a way to address this issue. In this regard, the flexibility of the demand side is a very important asset which can be harnessed by demand response (DR) programs.

DR programs incentivize end users to reduce or shift their electricity consumption during peak periods by means of time-based rates or direct load control [4]. In time-based rates programs, local grid operators set higher prices during peak periods when electricity
generation is expensive. In response, end users reduce their electricity consumption in these periods to save money. In direct load control programs, local grid operators are given the ability to shut off equipment during peak periods. In response, end users get discounted electricity bills or similar financial incentives for their participation. In all cases, an efficient control medium is essential to fully harness the flexibility of the demand side (as discussed in Section 2.3). To facilitate this, the CTA-2045 standard was established as a universal communication module for gathering data and dispatching commands from/to devices. The CTA-2045 standard is used for many devices including water heaters, thermostats, electric vehicle chargers, and pool pumps [5].

Being a prominent load in the U.S. residential buildings, electric water heaters are great assets for DR programs. Thanks to their storage tank, water heaters can store energy and provide demand flexibility to utilities when needed [6]. Residential buildings are the largest end-use electricity consumer in the U.S. [7]. Electric water heaters represent 14% of the electricity consumed in the U.S. residential buildings. Water heating is the third largest electricity end-use category, after air conditioning and space heating. On average, 46% of the residential buildings in the U.S. have an electric water heater [8]. Water heaters are not only one of the largest end-use electricity consumers, but also account for a good portion of household expenses. For example, an average household in the U.S. spends about USD 400–600 (0.45 ¢/L–0.68 ¢/L) on water heating every year [9]. For this reason, the use of heat pump water heaters (HPWHs), which is more energy efficient than standard electric water heaters, is on the rise [10].

1.1. Literature Review

In recent years, a significant number of research efforts have been devoted to studying the control of water heater and heating ventilation, and air conditioning (HVAC) systems in the contexts of DR and building energy management system (BEMS) applications. The control of water heater and HVAC systems is a complex problem, as it requires a simultaneous consideration of many factors including cost saving, comfort, safety, and reliability. To date, many unique and powerful control approaches have been proposed, but the main ones can be broadly categorized as rule-based, model predictive control (MPC), and reinforcement learning (RL) approaches.

Rule-based approaches are simple yet can be effective, depending on the expertise and knowledge of the developer. In these approaches, occupant comfort can be maintained while potentially reducing electricity cost and/or reducing electricity consumption [11]. For example, a rule-based approach can simply reduce the electricity cost of a water heater by decreasing its setpoint to the lower comfort band of the user during high price or unoccupied periods. Such rule-based approaches have been implemented by many studies. For example, Vanthournout et al. [12] presented a rule-based controller for a time-of-use (TOU) billing program. Their controller relied on three main rules: turn on the water heater at times when the price is low until it is fully recharged; turn off the water heater at times when the price is high unless the comfort of the user is in danger; and turn on the water heater when the price is high for a very brief period to maintain the comfort of the user, if needed. Péan et al. [13] evaluated a rule-based control strategy that adjusts water heater and indoor temperature setpoints according to electricity price and showed that their controller can save up to 20% through a simulation study. Perera and Skeie [14] presented a set of rule-based controllers that set setpoint and setback temperatures based on predefined occupancy schedules. Delgado et al. [15] provided a rule-based controller for maximizing self-consumption and reducing costs. Rule-based approaches, however, do not take any future information (e.g., future price, future hot water usage) into account, and therefore their efficacy is limited [16].

MPC approaches consist of three main steps: modeling, predicting, and controlling. Modeling refers to the development of a model that represents the characteristics of the thermal system (e.g., water heater, HVAC) to be controlled. Predicting refers to the prediction of the disturbances (e.g., water draw, outdoor temperature). Controlling refers to
the optimization of the control using the model and the prediction data resulting from the first two steps [11]. Unlike rule-based approaches, MPC approaches can have multiple objectives and constraints. MPC approaches are the most popular among DR and BEMS applications and have been used for many applications in the area of water heater and HVAC control [17]. For example, Tarragona [18] proposed an MPC strategy to improve the operation of a space-heating system coupled with renewable resources. Gholamibozanjani et al. [19] applied an MPC strategy for controlling a solar-assisted active HVAC system to minimize heating costs while providing the required comfortable temperature. Starke et al. [20] proposed an MPC-based control approach for heat pump water heaters to reduce electricity cost while maintaining the comfort and reducing the cycling of the water heater. Their results showed that the MPC controller prevented power use during the high price and critical peak periods while maintaining the comfort of the end users. Wang et al. [21] proposed an MPC-based controller to minimize the electricity bill while maintaining comfort for a real-time pricing (RTP) market under uncertain hot water demand and ambient temperature. Nazemi et al. [22] presented an incentive-based multi-objective nonlinear optimization approach for load scheduling of time-shiftable assets (e.g., dishwasher, washing machine) and thermal assets (e.g., water heater, HVAC). MPC approaches, however, have two main limitations. First, these approaches require a significant amount of time and effort for accurate modeling of the system to be controlled and failure to do so results in a poor control strategy. Additionally, due to the limited generalizability of the models in representing the characteristics of different thermal systems, an individual model needs to be developed for each system. For this reason, despite its good performance, the use of MPC may be inconvenient for some applications. Second, MPC optimizes an objective function at each control time step, which is a computationally expensive task that requires relatively powerful computational resources for real-time applications. Additionally, in the case of non-convex problems, the implementation of MPC can be challenging [23].

RL approaches can help addressing the many limitations associated with rule-based and MPC approaches. Parallel to the advancements in big data, machine learning algorithms, and computing infrastructure, RL approaches have become an important alternative for supporting DR and BEMS applications. On one hand, there exist many successful RL studies focusing on HVAC systems. These studies have already proved the feasibility and applicability of RL approaches to the various problems in the contexts of DR and BEMS. For example, [24–26] developed RL-based controllers to control HVAC systems. On the other hand, water heater control still remains largely untapped, except for only a few studies [17]. For example, Ruelens et al. [6] applied the fitted Q-iteration (FQI) to the control of an electric water heater for reducing the energy cost of the water heater and achieved 15% saving in 40 days. Kazmi et al. [27] proposed a model-based RL algorithm to optimize energy consumption of a set of water heaters and achieved a 20% saving while maintaining occupant comfort. Al-jabery et al. [28] proposed a Q-learning-based approach to solving the multi-objective optimization problem of minimizing the total cost of the power consumed, reducing the power demand during the peak period, and achieving customer satisfaction. Zsombinszki et al. [29] developed an RL-based controller to reduce the energy demand for heating, cooling and domestic hot water and compared the RL-based controller with a rule-based controller.

1.2. Research Gaps

Despite the importance of the aforementioned studies, there are still three main research gaps in RL approaches focusing on water heater control. First, there is a lack of studies focusing the control of HPWHs. The existing RL studies focused on controlling standard electric water heaters only. Unlike standard electric water heaters, HPWHs include both a heat pump and electric elements. The overall efficiency of HPWHs depends on the use of electric elements and the coefficient of performance (COP) of the heat pump. Thus, the control of HPWHs is more complex than that of standard electric water heaters, as it requires the consideration of both minimizing the use of electric elements and maximizing
the performance of the heat pump. Second, there is a lack of studies using a DR command standard (e.g., CTA-2045 standard) to control a water heater. The existing RL studies considered a binary action space that includes only turn on (recharge) or turn off. However, most of the water heaters available in the market today do not have the ability to perform these actions remotely. Instead, they receive DR commands (e.g., shed, load up). Given that the ease of implementation is imperative for the widespread adoption of such controllers, developing a controller that is based on a command standard is essential. Third, there is a lack of studies that use model-free RL algorithms for water heaters. The existing RL studies on water heater control used either model-based or batch RL algorithms. On one hand, the model-based algorithms require an accurate water heater model, which may not be available or difficult to obtain. On the other hand, the batch RL algorithms are offline algorithms and often inefficient for large neural networks as they involve repeated trainings of the networks [30,31].

1.3. Study Contributions

Towards addressing these research gaps, this study proposes a model-free RL approach that aims to minimize the electricity cost of a HPWH under a TOU electricity pricing policy by only using standard DR commands (e.g., shed, load up). In this approach, a set of RL agents, with different look ahead periods, were trained using the deep Q-networks (DQN) algorithm and their performance were tested on an unseen pair of price and hot water usage profiles. Additionally, the RL agents were compared to rule-based, MPC-based, and optimization-based controllers to further validate the performance of the proposed RL approach. This paper contributes to the body of knowledge in two main respects. First, this paper presents a model-free controller for a water heater based on standard DR commands to optimize its operation under TOU electricity pricing policy. Second, this paper compares the performance of the proposed model-free controller against a set of state-of-the-art rule-based, MPC-based and optimization-based controllers.

The paper is structured as follows. Section 2 provides a background about RL, HPWHs and the CTA-2045 standard that the DR commands are based on. Section 3 describes the methodology that was followed to develop the RL agents, the rule-based, the MPC-based, and the optimization-based controllers. Section 4 presents the training and testing results of the RL agents and provides a comparison between the RL agents and other controllers. Finally, Section 5 concludes the paper and discusses the limitations of this study.

2. Background

2.1. Reinforcement Learning

RL is a data-driven approach to understand and automate learning and decision-making aiming towards a goal. Unlike other computational approaches, RL approaches can learn by interacting with its environment and does not require any supervision [30]. The interaction between an RL agent and its environment can be formalized using the formal framework of Markov decision process (MDP). An MDP includes four elements: a set of states $S$, a set of actions $A$, a reward function $r : S \times A$, and transition probabilities between the states $P : S \times A \times S' \in [0, 1]$ [32].

The goal of RL is to find an optimal policy $\pi^*$ that maximizes the expected future reward from time $t : G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots$, where $0 \leq \gamma \leq 1$ is the discount rate parameter that determines how much to prioritize the future reward. Almost all RL algorithms either estimate a state-value function $V^\pi(s)$ to evaluate how good it is to be in the state $s$ or estimate an action-value function $Q^\pi(s,a)$ to evaluate how good it is to take the action $a$ at the state $s$. The way to achieve this depends on whether the characteristics of the environment are known. On one hand, if the characteristics of the environment are known, approaches such as model-based RL or dynamic programming (DP) can be taken. In these approaches, the RL agent learns or is given the characteristics
of the environment, then finds the optimal policy. The use of a state value function \( V^\pi(s) \), as defined in Equation (1) is more applicable for model-based approaches [33].

\[
V^\pi(s) = \mathbb{E}[G_t | S_t = s] = \mathbb{E}_{\pi} \left[ \sum_{k=0}^{\infty} \gamma^k r_{t+k} | S_t = s \right] \tag{1}
\]

where \( \mathbb{E} \) refers to the expected return (i.e., the discounted cumulative future rewards) of the agent when starting from the state \( s \) and following the policy \( \pi \).

On the other hand, model-free RL approaches do not need the characteristics of the environment. Such approaches discover an optimal policy by simply interacting with the environment. The use of action-value function \( Q^\pi(s,a) \), as defined in Equation (2), is more applicable for model-free approaches [33].

\[
Q^\pi(s,a) = \mathbb{E}_{\pi}[G_t | S_t = s, A_t = a] = \mathbb{E}_{\pi} \left[ \sum_{k=0}^{\infty} \gamma^k r_{t+k} | S_t = s, A_t = a \right] \tag{2}
\]

There are three main model-free RL approaches: policy-based, value-based, and actor–critic, which is a combination of policy- and value-based approaches. The policy-based approaches use optimization techniques and directly search for an optimal policy. In these approaches, the gradient of the objective function is calculated using Equation (3) and then the weight matrix that represents the policy \( \theta \) is updated as in Equation (4).

\[
\nabla_{\theta} J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \left[ \sum_{t=0}^{T} \nabla_{\theta} \log \pi_{\theta}(a_{i,t} | s_{i,t}) \sum_{t=0}^{T} r(s_{i,t}, a_{i,t}) \right] \tag{3}
\]

\[
\theta \leftarrow \theta + \alpha \nabla_{\theta} J(\theta) \tag{4}
\]

Value-based approaches include SARSA (on-policy) and Q-learning (off-policy), algorithms. These approaches learn the action-value function without explicitly representing the policy function. Both SARSA and Q-learning are tabular approaches and use a look up table (Q-table) to store the values of state-action pairs (Q-values).

The SARSA algorithm evaluates policies by constructing their value functions and uses these value functions to find improved policies [34]. The SARSA algorithm updates the Q-values as follows.

\[
Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha [r_t + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)] \tag{5}
\]

where \( \alpha \) refers to the learning rate. As such, the update in Equation (5) is performed after every transition from \( S_t \) to \( S_{t+1} \), given that \( S_{t+1} \) is not a terminal state. If \( S_{t+1} \) is the terminal state, then \( Q(S_{t+1}, A_{t+1}) \) is defined as zero. This algorithm is called SARSA because it uses \( S_t, A_t, R_{t+1}, S_{t+1}, A_{t+1} \).

The Q-learning algorithm begins with a random action-value function and updates to an improved action-value function in an iterative process until reaching the optimal action-value function. The Q-learning algorithm updates the Q-values as follows.

\[
Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[ r_t + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t) \right] \tag{6}
\]

The value-based approaches can also use deep neural networks (DNN) to estimate the Q-values instead of the Q-table when the size of the combinations of state-action pairs is very high. In such cases, the DNNs represented by \( \theta \) are trained to minimize the loss functions. The loss functions for deep SARSA and DQN are given in Equations (7) and (8), respectively.

\[
L_i(\theta_i) = \mathbb{E} \left[ (r_t + \gamma Q(S_{t+1}, A_{t+1}, \theta_{i-1}) - Q(S_t, A_t, \theta_i))^2 \right] \tag{7}
\]

\[
L_i(\theta_i) = \mathbb{E} \left[ (r_t + \gamma \max_a Q(S_{t+1}, a, \theta_{i-1}) - Q(S_t, A_t, \theta_i))^2 \right] \tag{8}
\]
Finally, actor–critic approaches combine value-based and policy-based approaches and use two agents: an actor and a critic. For actor–critic approaches, by replacing the second term with the action-value function (Q-value), Equation (3) can be rewritten as follows:

$$
\nabla_{\theta} J(\theta) = E \left[ \sum_{t=0}^{T} \nabla_{\theta} \log \pi_{\theta}(a_{t}|s_{t})Q_{w}(s_{t}, a_{t}) \right]
$$

where the critic estimates the action-value function parameterized by $w$ and the actor updates the policy parameters $\theta$.

2.2. Heat Pump Water Heaters

Unlike standard electric water heaters that only include a single or two electric elements to generate heat directly, HPWHs include both a heat pump and electric elements. These water heaters use only the heat pump most of the time and use electric elements as backup during periods of high hot water demand [35]. Because heat pumps move heat from one place to another rather than generating it directly, HPWHs are more efficient than standard electric water heaters [36]. HPWHs can achieve savings more than 50% compared to standard electric water heaters [37].

The overall efficiency of HPWHs depends on two main factors: the use of electric elements and the COP of the heat pump. The higher the use of electric elements, the lower the overall efficiency is. However, operating the water heater with a very narrow deadband to avoid element usage is also not efficient, due to increased losses and higher average heat sink temperatures. Therefore, the lower the water temperature (i.e., heat sink temperature), the higher the heat pump efficiency is. Thus, developing control strategies for HPWHs that maximize heat pump efficiency and minimize the use of relatively inefficient electric elements is critical.

2.3. The CTA-2045 Standard

An efficient control medium is essential to fully harness the flexibility of the demand side. However, the control of the flexible loads is not an easy task, because the DR operators have to coordinate a large number of devices with different manufacturers and communication protocols. To facilitate this, the CTA-2045 standard was established as a universal communication module for gathering data and dispatching commands from/to devices. To date, the CTA-2045 standard has been used for many devices including water heaters, thermostats, electric vehicle chargers, and pool pumps [5].

In the case of water heaters, the CTA-2045 standard enables a water heater with a CTA-2045 communication module to be monitored and controlled remotely. For example, DR operators can monitor the properties (e.g., water heater size and rated power) and operation data (e.g., total and present energy storage capacity) of water heaters and can send several DR commands (e.g., shed and load up) to control water heaters. The use of CTA-2045-compatible water heaters is expected to increase due to the incentives provided to the manufacturers and utilities. For example, it is expected that CTA-2045-enabled water heaters will represent 91% of all water heaters in Washington and Oregon by 2039 [38]. If only 26.5% of these electric water heaters are enrolled for a DR program in these states, 301 MW of demand response potential can be achieved [39]. As the penetration of CTA-2045 grows, it is imperative to develop control strategies that use the DR commands of the CTA-2045 to control water heaters.

3. Methodology

An RL agent can be deployed in an environment to derive an optimal policy through continuously interacting with the environment without any prior knowledge. Although this is feasible for online RL algorithms (e.g., DQN), many researchers have opted to use simulators to train and test the algorithms to avoid the long learning periods and the poor control performance during the learning process [11,40]. Similarly, this study used a simu-
lator for training and testing the RL agents. Figure 1 summarizes the methodology of this study. Accordingly, the research methodology consists of four main steps: (1) developing a CTA-2045-compatible HPWH simulator, (2) formulating the main components (e.g., state, action and reward) of the RL control problem, (3) training the RL agents, (4) developing rule-based, MPC-based, and optimization-based controllers, and (5) performance testing.

Figure 1. The framework of the methodology.

3.1. Simulator Development

A 6-node HPWH simulator with two electric elements and a heat pump, adapted from [35,41] was developed. The diagram of the water heater is shown in Figure 2. The 6-node water heater splits the water tank into six equal volumes and assumes that the temperature of each node is homogeneous and the only heat transfer between the nodes is via advection (i.e., bulk motion of the water from one node to another during hot water draws). The simulator calculates the temperature change in each node based on the heat added from the heat pump and/or an element, heat loses through the tank walls, and water draws.

Figure 2. The diagram of the water heater.

The temperature change in each node ($\Delta T_w(N)$) at every minute was calculated using the following equation:

$$
\Delta T_w(N) = \frac{\dot{Q}_{HP} \times C_{HP}(N) + \dot{Q}_E \times C_E(N) - UA(N) \times (T_w(N) - T_{amb}) + m \times c_p \times (T_w(N+1) - T_w(N))}{c_p \times m(N) \times \beta}
$$

(10)

where $\dot{Q}_{HP}$ and $\dot{Q}_E$ are the heats added by the heat pump and an element, respectively, $C_{HP}(N)$ and $C_E(N)$ are the fractions of the heat added to node $N$ by the heat pump and element, respectively (Equation (11)), $UA(N)$ is the standby heat loss coefficient of node $N$, $T_w(N)$ is the water temperature of node $N$, $T_{amb}$ is the ambient temperature of the room where the water heater is located, $m$ is the mass flow rate, $c_p$ is the specific heat of water, $T_w(N+1)$ is the water temperature of node $N + 1$, and $\beta$ is the fraction of heat added to node $N$ by the element.
where $s$ is the perceived distance between the lower and upper tank, $T_{set}$ is the temperature setpoint, and $p$ is a parameter to tune the concavity of $T_w(N)$ reaching the setpoint.

The heat added to the tank by the heat pump ($\dot{Q}_{HP}$) or an element ($\dot{Q}_{E}$) and the resulting electricity consumption ($E_{WH}$) in a minute are calculated using the following equations.

$$\dot{Q}_{HP} = \frac{P_{HP} \times \text{COP}_{HP} \times u_{HP}}{60}$$

$$\dot{Q}_{E} = \frac{P_{E} \times \text{COP}_{E} \times u_{E}}{60}$$

$$\dot{Q}_{E} = \frac{P_{E} \times \text{COP}_{E} \times u_{E}}{60}$$

where $P_{HP}$ and $P_{E}$ are the rated powers, COP$_{HP}$ and COP$_{E}$ are the coefficients of performance, $u_{HP}$ and $u_{E}$ are the status (on/off) of the heat pump and an element, respectively. The water heater uses a deadband control logic which uses the weighted average of the temperatures of node 2 and node 5 (Equation (15)). When there is no DR command, the deadbands for the heat pump and the upper element are 5 °C and 10 °C, respectively. For example, if the weighted average drops 5 °C below the setpoint, the heat pump is turned on to heat the water back to the setpoint. If the temperature of node 2 drops 10 °C below the setpoint, first the upper element is turned on to heat the water back to the 4 °C below the setpoint, and then the upper element is turned off and lower element is turned on to heat the water back to the setpoint. The upper and lower elements cannot be on at the same time. However, whenever any element is turned on, the heat pump is also automatically turned on. The DR commands can either increase or decrease the deadband for the heat pump but cannot change the control logic described above.

$$T_{wp} = 0.75 \times T_{w}(2) + 0.25 \times T_{w}(5)$$

### 3.2. Reinforcement Learning Problem Formulation

Four RL agents were formulated. In formulating these RL agents, four different state variable configurations were considered, but the action space and the reward function of the four agents were kept identical. To develop the proposed agents, it was assumed that a CTA-2045-compatible water heater (or simulator) for gathering data and dispatching commands from/to the water heater, were kept identical. To develop the proposed agents, it was assumed that a CTA-2045-compatible water heater (or simulator) for gathering data and dispatching commands from/to the water heater, were kept identical. To develop the proposed agents, it was assumed that a CTA-2045-compatible water heater (or simulator) for gathering data and dispatching commands from/to the water heater, were kept identical.
next two hours, respectively. Consequently, their state variables included 13, 17 and 25 variables, respectively. Table 1 summarizes the state variables of all RL agents.

Table 1. State variables of the RL agents.

<table>
<thead>
<tr>
<th>RL Agent #</th>
<th>State Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$T_w(1)<em>t, \ldots, T_w(6)<em>t, s1_t, s2_t, s3_t, mean(\lambda</em>{t+15}), \text{mean}(\lambda</em>{t+15} + 2 \times 15)$</td>
</tr>
<tr>
<td>2</td>
<td>$T_w(1)<em>t, \ldots, T_w(6)<em>t, s1_t, s2_t, s3_t, \text{mean}(\lambda</em>{t+15}), \text{mean}(\lambda</em>{t+15} + 2 \times 15)$</td>
</tr>
<tr>
<td>3</td>
<td>$T_w(1)<em>t, \ldots, T_w(6)<em>t, s1_t, s2_t, s3_t, \text{mean}(\lambda</em>{t+15}), \ldots, \text{mean}(\lambda</em>{t+15} + 6 \times 15)$</td>
</tr>
<tr>
<td>4</td>
<td>$T_w(1)<em>t, \ldots, T_w(6)<em>t, s1_t, s2_t, s3_t, \text{mean}(\lambda</em>{t+15}), \ldots, \text{mean}(\lambda</em>{t+15} + 8 \times 15)$</td>
</tr>
</tbody>
</table>

$T_w(N)_t$: water temperature of node $N$ at time $t$, $s1_t$: heat pump status at time $t$, $s2_t$: lower element status at time $t$, $s3_t$: upper element status at time $t$, $\lambda_t$: electricity price at time $t$, $V_t$: hot water usage volume at time $t$.

The action space for all RL agents included three standard DR commands: shed, normal, and load up. Shed is a curtailment event that increases the deadband for the heat pump to 10 °C. Under shed command, the heat pump turns on when the weighted average given in Equation (15) drops 10 °C below the setpoint to heat up the water back to the setpoint. Load up command, on the other hand, forces the water heater to heat up to the setpoint temperature by reducing the deadband for the heat pump. Under the load up command, the heat pump turns on when the weighted average drops 1 °C below the setpoint to heat up the water back to the setpoint. Normal command cancels any existing DR command and run the water heater with the default deadband for the heat pump.

Finally, the reward ($r_t$) that the agents receive after taking an action at each control time step is given in Equation (16). The reward function only includes a term about electricity cost and does not include any term about comfort due to two reasons. First, user comfort is already maintained because the actions do not change the water temperature setpoint set by users. Second, the upper nodes are reserved for end users by the manufacturer and are not available for any DR command. Regardless of the received DR command, the water heater control logic guarantees the availability of hot water by heating the water in upper nodes when needed.

$$r_t = -c_t = -E_{WH,t+15} \times \lambda_{t+15}$$  \hspace{1cm} (16)

where $c_t$ is the electricity cost calculated by multiplying the electricity consumed in previous control time step ($E_{WH,t+15}$) by the electricity price in previous control time step ($\lambda_{t+15}$).

3.3. Reinforcement Learning Training

For training the RL agents, the DQN algorithm (Figure 3) was used. The DQN algorithm, presented in Algorithm 1, has two important ingredients. First, the DQN algorithm includes two networks: policy network ($\theta$) and target network ($\hat{\theta}$). The policy network is constantly updated and determines the actions to take while interacting with the environment. The target network is updated by taking the weights from the policy network at regular intervals and predicts the Q-value of the next state. Second, the DQN algorithm includes a replay memory to store previous experiences. The policy network updates its weights, using a random batch (of experiences) taken from the replay memory. The replay memory helps with breaking the correlation between consecutive observations [40]. The algorithm uses the following loss function:

$$L_t(\theta_i) = E \left[ (r_t + \gamma \max_{\hat{\theta}} Q(S_{t+1}, A_t, \hat{\theta}) - Q(S_t, A_t, \theta_i))^2 \right]$$  \hspace{1cm} (17)
### 3.4. Rule-Based, MPC-Based, and Optimization-Based Controllers

To better evaluate the performance of the proposed RL approach, rule-based, MPC-based and optimization-based controllers were also developed as benchmarks. The rule-based controller followed the logic provided in Algorithm 2. This controller sends load up to heat up the water when the price is low and there is an upcoming water draw; sends normal when the price is low and there is no upcoming water draw; and sends shed in other cases to operate the water heater in lower temperatures for higher COP values.

![An overview of the DQN algorithm.](image)

**Algorithm 1** DQN training

<table>
<thead>
<tr>
<th>Algorithm 1 DQN training</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialize replay memory with capacity of ( N )</td>
</tr>
<tr>
<td>Initialize policy network with random weights</td>
</tr>
<tr>
<td>Initialize the counter: ( \xi = 0 )</td>
</tr>
<tr>
<td>for episode = 1, ( M ) do</td>
</tr>
<tr>
<td>Get the initial state ( s_t )</td>
</tr>
<tr>
<td>Copy the weights of policy network to target network at every ( K ) episode</td>
</tr>
<tr>
<td>for ( t = 1, dt, T ) do</td>
</tr>
<tr>
<td>Scale state values between ([0, 1])</td>
</tr>
<tr>
<td>Update the value of ( \epsilon ): ( \epsilon = \epsilon_{start} + (\epsilon_{start} - \epsilon_{end}) \times e^{-\frac{t}{\epsilon_{decay}}} )</td>
</tr>
<tr>
<td>Select a random action ( a_t ) with probability of ( \epsilon )</td>
</tr>
<tr>
<td>Otherwise select action ( a_t ), using the policy network</td>
</tr>
<tr>
<td>Execute action ( a_t ) in the water heater model</td>
</tr>
<tr>
<td>Observe reward ( r_t ) and next state ( s_{t+1} )</td>
</tr>
<tr>
<td>Store transitions ((s_t, a_t, r_t, s_{t+1})) in replay memory</td>
</tr>
<tr>
<td>Move to next state: ( s_t := s_{t+1} )</td>
</tr>
<tr>
<td>Sample a batch of transitions ((s, a, r, s')) from replay memory</td>
</tr>
<tr>
<td>Calculate state action values (Q(s, a, \theta)) using the policy network</td>
</tr>
<tr>
<td>Calculate the expected state action values ( y = \begin{cases} r &amp; \text{for terminal} \ r + \gamma \max_{a'} Q(s', a', \theta') &amp; \text{for non terminal} \end{cases} )</td>
</tr>
<tr>
<td>Calculate the loss between the calculated (Q(s, a, \theta)) and ( y ) values using Equation (17)</td>
</tr>
<tr>
<td>Perform a gradient descent and update the weights of ( \theta )</td>
</tr>
<tr>
<td>Update the counter: ( \xi := \xi + 1 )</td>
</tr>
<tr>
<td>end</td>
</tr>
<tr>
<td>end</td>
</tr>
</tbody>
</table>

The diagram illustrates the DQN algorithm.
Algorithm 2 Rule-based controller

\[
\text{Algorithm 2 Rule-based controller}
\]

\[
\text{for } t = 1, dt, T \text{ do}
\]

\[
\text{if mean(λ}_{t+dt}^T) < \text{mean}(λ_{1:T}) \text{ and sum}(V_{t+dt}^T) > 0 \text{ then}
\]

\[
\text{send Load up}
\]

\[
\text{else if mean(λ}_{t+dt}^T) < \text{mean}(λ_{1:T}) \text{ and sum}(V_{t+dt}^T) = 0 \text{ then}
\]

\[
\text{send Normal}
\]

\[
\text{else}
\]

\[
\text{send Shed}
\]

For fairness of comparison, the MPC-based controllers used the same prediction and control horizons as the RL agents. Accordingly, MPC-based controllers with three different prediction horizons were developed: MPC with 30 min, an hour, and two hours prediction horizons. Additionally, for all MPC-based controllers, the control horizon of 15 min, same as the RL agents, was implemented. The MPC-based controllers are based on three main steps. First, the controller gets the temperatures of all nodes from the simulator and collects the electricity price and hot water usage volumes for the corresponding prediction horizon. Second, the control variables are optimized by genetic algorithm (GA) for the prediction horizon for a minimum electricity cost. The optimization problem was solved using GA because the water heater simulator is nonlinear and non-convex. For nonlinear models, global search metaheuristic methods such as GA, can achieve near-optimal solutions with a certain probability, which is useful to solve the optimization problems with complicated scenarios [30,31]. Third, the optimal control variables resulting from the optimization were implemented for the control horizon.

For the optimization-based controller, the control variables were optimized for the entire simulation period using GA. This controller assumes that perfect information about the future is available. For example, it assumes that a day-ahead hot water usage volume is available with perfect accuracy. Therefore, this controller may not be applicable for real problems. However, in this study it was implemented only for the sake of creating a golden standard. Both MPC-based and optimization-based controllers were developed using Python on a standard four-core personal computer.

3.5. Performance Testing

The performance testing included two comparisons. First, the resulting RL agents were compared against a baseline that was established by running the simulator without sending any DR commands. As such, the resulting RL agents were deployed on the simulator for 30 days. For these 30 days, the simulator was provided with hot water usage and price profiles different from the ones used in the training stage. The testing of the RL agents was performed in an offline manner. The RL agents were not further trained during the testing. Second, the performance of the RL agents were compared against the rule-based, MPC-based, and optimization-based controllers for five days. The comparison between RL agents and other controllers was conducted for five days because the MPC-based controllers require an optimization in each control time step. Thus, longer simulation periods result in longer computational times. Additionally, because the optimization-based controller optimizes the operation for the entire simulation period, longer simulation periods would result in more variables, which makes the optimization more challenging.

4. Experiments and Results

4.1. Water Heater Experiments and Simulator Results

To ensure the water heater simulator was realistic and represents the behavior of an actual water heater, a set of experiments with an actual 250-l HPHW was conducted to determine the parameters of the simulator. As such, the water heater was instrumented with three thermistors, an ambient temperature sensor, a power meter, and a flow meter. Table 2 summarizes the instruments and their error ranges. The following data were collected from the water heater in minute intervals: inlet water temperature (°C), outlet water temperature (°C), ambient air temperature (°C), lower tank water temperature (°C), upper tank water temperature (°C), hot waterflow (GPM), and electricity demand (W). The resulting dataset was split into training and testing datasets. Using the training dataset, four analyses were performed. First, the values of water mass in a node and specific heat of water were set using common knowledge. Second, parameters including ambient temperature, inlet water temperature, temperature setpoint, heat pump rated power, and element rated power were assumed constant and were set by averaging their values in the training dataset. Third, the COPs of the heat pump and elements were calculated. Finally, the remaining unknown parameters (e.g., UA, p) were
optimized in a way that minimizes the sum of the root mean squared error (RMSE) of lower and upper temperatures. Table 3 shows the resulting parameters of the water heater simulator.

### Table 2. Measurement devices.

<table>
<thead>
<tr>
<th>Instrument Type</th>
<th>Instrument Name</th>
<th>Error Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thermistor</td>
<td>Omega 44031 10 kOhm Precision</td>
<td>±0.01 °C</td>
</tr>
<tr>
<td>Ambient temperature sensor</td>
<td>Campbell Scientific HC2S3</td>
<td>±0.01 °C</td>
</tr>
<tr>
<td>Power meter</td>
<td>WattNode Pulse</td>
<td>±0.5%</td>
</tr>
<tr>
<td>Flow meter</td>
<td>Omega FTB4607</td>
<td>±2.0%</td>
</tr>
</tbody>
</table>

### Table 3. Parameters of the water heater simulator.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Explanation</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c_p$</td>
<td>Specific heat of water</td>
<td>4.184</td>
<td>J/(g·K)</td>
</tr>
<tr>
<td>$m(N)$</td>
<td>Water mass in a node</td>
<td>41.7</td>
<td>kg</td>
</tr>
<tr>
<td>$T_{amb}$</td>
<td>Ambient temperature</td>
<td>21.5</td>
<td>°C</td>
</tr>
<tr>
<td>$T_{inlet}$</td>
<td>Inlet water temperature</td>
<td>23.9</td>
<td>°C</td>
</tr>
<tr>
<td>$T_{set}$</td>
<td>Temperature setpoint</td>
<td>51</td>
<td>°C</td>
</tr>
<tr>
<td>$P_{HP}$</td>
<td>Heat pump rated power</td>
<td>400</td>
<td>W</td>
</tr>
<tr>
<td>$P_E$</td>
<td>Element rated power</td>
<td>4500</td>
<td>W</td>
</tr>
<tr>
<td>COP_{HP}</td>
<td>COP of heat pump</td>
<td>-0.004 × $T_w(5)^2 + 0.19 × T_w(5) + 3.56</td>
<td>N/A</td>
</tr>
<tr>
<td>COP_{E}</td>
<td>COP of element</td>
<td>0.99</td>
<td>N/A</td>
</tr>
<tr>
<td>$UA(1) \ldots UA(6)$</td>
<td>Standby heat loss coefficients</td>
<td>0.04, 0.03, 0.03, 0.03, 0.03, 0.06</td>
<td>kJ/(min·K)</td>
</tr>
<tr>
<td>$\beta$</td>
<td>Water tank wall coefficient</td>
<td>1.12</td>
<td>N/A</td>
</tr>
<tr>
<td>$s$</td>
<td>Perceived distance</td>
<td>3.7</td>
<td>N/A</td>
</tr>
<tr>
<td>$p$</td>
<td>Concavity parameter</td>
<td>1.06</td>
<td>N/A</td>
</tr>
</tbody>
</table>

#### 4.2. Price and Hot Water Usage Profiles

Separate pairs of price and water usage profiles were used for training and testing the RL agents. In training the RL agents, a typical TOU price signal, consisting of 3 h peak and 21 h off-peak prices per day, was used. The peak price period in TOU price signal was determined randomly for every day. In testing the RL agents, a price signal, consisting of two peaks, a mid-peak, and an off peak, was used to better verify the adaptability of the RL agents to different price signals. For hot water usage profiles, the domestic hot water (DHW) event schedule generator tool provided by the U.S. Department of Energy (DOE) was used [43]. The RL agents were trained on June and July usage profiles and tested on August profile. Figure 5 shows a sample from the price and hot water usage profiles for training and testing.

Then, the simulator was validated using the testing dataset. The simulator was run with the same conditions as the actual water heater. Then, the measurements from the actual and the simulated water heaters were compared. Figure 4 shows the accuracy of the simulator in predicting upper and lower temperatures and power consumption level for a randomly selected day. As shown, the simulator was pretty accurate. It achieved RMSEs of 1.13 °C and 1.34 °C for lower and upper tank temperatures, respectively. Overall, these results showed that the water heater simulator is realistic and therefore capable of representing the behavior of an actual water heater.

#### 4.3. Training Results

All RL agents were trained based on Algorithm 1 using Python with the PyTorch framework [44] on a standard four-core personal computer with parameters as summarized in Table 4. The training of each RL agent was repeated five times. Figure 6 shows the average, maximum, and minimum operation costs of the RL agents per episode. The RL agents reduced the electricity cost per episode as the training advances. Despite all RL agents being able to reduce the electricity cost significantly, the RL agents with more future state variables have achieved more savings in training data. For example, the RL agent #4 was able to reduce the average electricity cost to USD 7.59 while the RL agent #1 could reduce it to USD 10.17. On average, one episode took 3 min to be simulated on a machine with an Intel Xeon CPU @ 3.70 GHz processor and 16.0 GB RAM.
Inlet water temperature 23.9 °C
Temperature setpoint 51 °C
Heat pump rated power 400 W
Element rated power 4500 W
COP of heat pump
\[-0.004 \times T_0(5)^{0.19} \times T_0(5) + 3.56\]
COP of element 0.99 N/A
Standby heat loss coefficients 0.04, 0.03, 0.03, 0.03, 0.03, 0.06 kJ/(min · K)
Water tank wall coefficient 1.12 N/A
Perceived distance 3.7 N/A
Concavity parameter 1.06 N/A

Then, the simulator was validated using the testing dataset. The simulator was run with the same conditions as the actual water heater. Then, the measurements from the actual and the simulated water heaters were compared. Figure 4 shows the accuracy of the simulator in predicting upper and lower temperatures and power consumption level for a randomly selected day. As shown, the simulator was pretty accurate. It achieved RMSEs of 1.13 °C and 1.34 °C for lower and upper tank temperatures, respectively. Overall, these results showed that the water heater simulator is realistic and therefore capable of representing the behavior of an actual water heater.

Figure 4. Simulator performance.

4.2. Price and Hot Water Usage Profiles
Separate pairs of price and water usage profiles were used for training and testing the RL agents. In training the RL agents, a typical TOU price signal, consisting of 3 h peak and 21 h off-peak prices per day, was used. The peak price period in TOU price signal was determined randomly for every day. In testing the RL agents, a price signal, consisting of two peaks, a mid-peak, and an off peak, was used to better verify the adaptability of the RL agents to different price signals. For hot water usage profiles, the domestic hot water (DHW) event schedule generator tool provided by the U.S. Department of Energy (DOE) was used [43]. The RL agents were trained on June and July usage profiles and tested on August profile. Figure 5 shows a sample from the price and hot water usage profiles for training and testing.

Figure 5. A sample from price and hot water profiles for training and testing.

4.3. Training Results
All RL agents were trained based on Algorithm 1 using Python with the PyTorch framework [44] on a standard four-core personal computer with parameters as summarized in Table 4. The training of each RL agent was repeated five times. Figure 6 shows the average, maximum, and minimum operation costs of the RL agents per episode. The RL agents reduced the electricity cost per episode as the training advances. Despite all RL agents being able to reduce the electricity cost significantly, the RL agents with more future state variables have achieved more savings in training data. For example, the RL agent #4 was able to reduce the average electricity cost to USD 7.59 while the RL agent #1 could reduce it to USD 10.17. On average, one episode took 3 min to be simulated on a machine with an Intel Xeon CPU @ 3.70 GHz processor and 16.0 GB RAM.

Table 4. Hyperparameters of the RL agent training.

<table>
<thead>
<tr>
<th>Parameter Explanation</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Batch size</td>
<td>32</td>
</tr>
<tr>
<td>Discount rate ((\gamma))</td>
<td>0.99</td>
</tr>
<tr>
<td>(\varepsilon_{0,1,6,\ldots,6})</td>
<td>0.5</td>
</tr>
<tr>
<td>(\varepsilon_{6,0,\ldots,0})</td>
<td>0.03</td>
</tr>
<tr>
<td>(\varepsilon_{0,6,\ldots,6})</td>
<td>140,000</td>
</tr>
<tr>
<td>Target network update frequency ((K))</td>
<td>Every episode</td>
</tr>
</tbody>
</table>
Table 4. Hyperparameters of the RL agent training.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Batch size</td>
<td>32</td>
</tr>
<tr>
<td>Discount rate ($\gamma$)</td>
<td>0.99</td>
</tr>
<tr>
<td>$\epsilon_{\text{start}}$</td>
<td>0.5</td>
</tr>
<tr>
<td>$\epsilon_{\text{end}}$</td>
<td>0.03</td>
</tr>
<tr>
<td>$\epsilon_{\text{decay}}$</td>
<td>140,000</td>
</tr>
<tr>
<td>Target network update frequency ($K$)</td>
<td>Every episode</td>
</tr>
<tr>
<td>Number of episodes ($M$)</td>
<td>125</td>
</tr>
<tr>
<td>Episode length ($T$)</td>
<td>61 days</td>
</tr>
<tr>
<td>Replay memory capacity ($N$)</td>
<td>25,000</td>
</tr>
<tr>
<td>DNN structure</td>
<td>[L, 512, 512, 3] 1</td>
</tr>
<tr>
<td>DNN optimizer</td>
<td>RMSprop</td>
</tr>
<tr>
<td>Optimizer learning rate</td>
<td>0.0001</td>
</tr>
<tr>
<td>DNN loss function</td>
<td>Mean squared error (MSE)</td>
</tr>
<tr>
<td>Control time step ($dt$)</td>
<td>15 min</td>
</tr>
</tbody>
</table>

1. L is the number of state variables.

Figure 6. Training performance of the RL agents.

4.4. Testing Results

Table 5 shows the electricity costs, element usages, and average COPs of the baseline and the RL agents over 30 days. All RL agents were able to achieve significant cost savings over the baseline over this period. The operation cost of the baseline was USD 6.04. Figure 7 shows the operation of the baseline model on a randomly selected day. RL agents #1, #2, #3 and #4 were able to reduce the cost of the water heater operation to USD 5.03, USD 4.37, USD 3.55 and USD 3.44, respectively, over the same period. Additionally, in all cases the temperature of the hot water drawn was never less than the 10 °C below the setpoint, which shows that comfort of end users were maintained.

Table 5. Comparison of baseline and RL.

<table>
<thead>
<tr>
<th>Operation Strategy</th>
<th>Look Ahead</th>
<th>Electricity Cost</th>
<th>Element Usage</th>
<th>Average COP$_{HP}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>N/A</td>
<td>USD 6.04</td>
<td>121 min</td>
<td>4.79</td>
</tr>
<tr>
<td>RL agent #1</td>
<td>30 min *</td>
<td>USD 5.03</td>
<td>226 min</td>
<td>5.30</td>
</tr>
<tr>
<td>RL agent #2</td>
<td>30 min</td>
<td>USD 4.37</td>
<td>106 min</td>
<td>5.39</td>
</tr>
<tr>
<td>RL agent #3</td>
<td>1 h</td>
<td>USD 3.55</td>
<td>73 min</td>
<td>5.47</td>
</tr>
<tr>
<td>RL agent #4</td>
<td>2 h</td>
<td>USD 3.44</td>
<td>71 min</td>
<td>5.58</td>
</tr>
</tbody>
</table>

* Future price information only.
The RL agents were able to reduce the electricity cost of the water heater for two main reasons. First, all RL agents could successfully avoid peak price periods and run the heat pump and/or an element during off price periods whenever it is possible. For example, the RL agent #4 turned on the heat pump for 8945 min and only 305 min of these were during peak price. However, the baseline turned on the heat pump for 12,366 min and 1765 min of these were during peak price. For RL agent #4, only 3.41% of the heat pump operation coincided with peak price periods, whereas 14.27% for the baseline. Second, RL agents #2, #3 and #4 could maintain a good balance between the use of heat pump and relatively inefficient electric elements. As a result, these RL agents achieved higher COP$_{HP}$ values and less element usages. For example, the RL agent #2 used elements for 106 min and had the average COP$_{HP}$ of 5.39, whereas the baseline used elements for 121 min and had the average COP$_{HP}$ of 4.79. RL agent #1, however, was not able to reduce the element usage because it could not do any preheating prior to large hot water draws since it does not have any information about future hot water draws. Figure 8 shows the operation of RL agent #1 on a sample day. As shown in the figure, the agent only sent normal and load up commands when the price was low and was able to avoid higher price periods. For example, the agent heated up the water by sending load up commands before and after the peak price around 15:00. Consequently, the agent could avoid peak price periods. In addition, the agent sent shed commands most of the time to operate in lower temperatures for higher COP values. However, unlike RL agent #1, did not do any heating before or after peak price periods and therefore could not avoid the peak price at 17:00.

Table 6 summarizes the performance of the RL agents, the rule-based, MPC-based, and optimization-based controllers over five days. The rule-based controller achieved 19.1% savings. RL agents #2, #3 and #4 achieved 19.1%, 34.6% and 35.3% savings, respectively, whereas their counterparts, MPC #2, MPC #3 and MPC #4, achieved 3.7%, 23.5% and 30.9% savings, respectively. The optimization-based controller, as the golden standard, achieved a 40.4% saving.
for higher COP values. However, unlike RL agent #1, did not do any heating before or after peak price periods and therefore could not avoid the peak price at 17:00.

Table 5. Comparison of baseline and RL.

<table>
<thead>
<tr>
<th>Operation Strategy</th>
<th>Look Ahead</th>
<th>Electricity Cost</th>
<th>Element Usage</th>
<th>Average COP&lt;sub&gt;HP&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>N/A</td>
<td>USD 6.04</td>
<td>121 min</td>
<td>4.79</td>
</tr>
<tr>
<td>RL agent #1</td>
<td>30 min</td>
<td>USD 5.03</td>
<td>226 min</td>
<td>5.30</td>
</tr>
<tr>
<td>RL agent #2</td>
<td>30 min</td>
<td>USD 4.37</td>
<td>106 min</td>
<td>5.39</td>
</tr>
<tr>
<td>RL agent #3</td>
<td>1 h</td>
<td>USD 3.55</td>
<td>73 min</td>
<td>5.47</td>
</tr>
<tr>
<td>RL agent #4</td>
<td>2 h</td>
<td>USD 3.44</td>
<td>71 min</td>
<td>5.58</td>
</tr>
</tbody>
</table>

* future price information only.

Figure 7. Baseline operation.

Figure 8. Operation based on RL agent #1.

Table 6. Performance of RL agents, and MPC-based and optimization-based controllers.

<table>
<thead>
<tr>
<th>Operation Strategy</th>
<th>Look Ahead</th>
<th>Electricity Cost</th>
<th>Element Usage</th>
<th>Average COP&lt;sub&gt;HP&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>N/A</td>
<td>USD 1.36</td>
<td>75 min</td>
<td>4.88</td>
</tr>
<tr>
<td>Rule-based</td>
<td>N/A</td>
<td>USD 1.10</td>
<td>58 min</td>
<td>4.63</td>
</tr>
<tr>
<td>RL agent #2</td>
<td>30 min</td>
<td>USD 1.10</td>
<td>68 min</td>
<td>5.35</td>
</tr>
<tr>
<td>RL agent #3</td>
<td>1 h</td>
<td>USD 0.89</td>
<td>58 min</td>
<td>5.27</td>
</tr>
<tr>
<td>RL agent #4</td>
<td>2 h</td>
<td>USD 0.88</td>
<td>54 min</td>
<td>5.47</td>
</tr>
<tr>
<td>MPC #2</td>
<td>30 min</td>
<td>USD 1.31</td>
<td>157 min</td>
<td>5.76</td>
</tr>
<tr>
<td>MPC #3</td>
<td>1 h</td>
<td>USD 1.04</td>
<td>104 min</td>
<td>5.70</td>
</tr>
<tr>
<td>MPC #4</td>
<td>2 h</td>
<td>USD 0.94</td>
<td>84 min</td>
<td>5.61</td>
</tr>
<tr>
<td>Optimization</td>
<td>5 days</td>
<td>USD 0.81</td>
<td>47 min</td>
<td>5.48</td>
</tr>
</tbody>
</table>

Overall, all controllers achieved significant savings compared to baseline. The rule-based controller, despite its very simple structure, was able to perform better than MPC #2. In terms of element usage, it performed very closely to RL agent #4 and the optimization-based controller, but could not maintain a good balance between element usage and COP<sub>HP</sub>, and therefore got the lowest COP<sub>HP</sub> values. The rule-based controller simply operated the water heater in high temperatures to minimize the use of elements at the expense of low heat pump efficiency. The RL agents outperformed their corresponding MPC controllers for all look ahead periods including 30 min, 1 h and 2 h. For example, the RL agent with 2 h of future hot water usage volumes and electricity prices information (RL agent #4) achieved USD 0.88, while MPC #4 achieved USD 0.94 using the same information. Figures 9 and 10 show the operations based on RL agent #4 and MPC #4, respectively. Overall, the RL agents maintained a good balance between element usage and COP<sub>HP</sub>. The MPC-based controllers achieved the highest COP<sub>HP</sub> values, which however increased their element use. For example, RL agent #4 had an average COP<sub>HP</sub> of 5.47, while MPC #4, the equivalent of RL agent #4, had an average COP<sub>HP</sub> of 5.61. RL agent #4 used elements for 54 min only while MPC #4 used elements for 84 min. As a result, despite very close performance, RL agent #4 cost 6¢ less than MPC #4 in five days due to less element usage. The cost of RL agent #4 was also very close to the cost of the optimization-based controller. These results, therefore, show that RL is able to reduce electricity cost without any prior knowledge about the water heater and its power usage. RL can also adapt to unseen price signal and water usage profiles.
In practice, this research can empower multiple stakeholders, including utilities and building occupants. Because the proposed approach is easily scalable and uses the standard DR commands only, utilities can easily adapt this RL approach to their DR programs to control participating water heaters. For example, utilities can use such approaches to minimize electricity consumption during peak periods or to maximize the consumption of renewable energy. Given that the use of the CTA-2045-compatible water heaters is expected to increase due to the incentives provided to the manufacturers, there is a tremendous potential for such approaches to have an application area in utility side. Additionally, these results showed that RL can reduce the cost of the water heater operation more than 30%. Therefore, this research has an application area for the development of BEMS to save building occupants electricity cost. In the context of BEMS, similar RL approaches can also be used for other devises such as HVAC and energy storage units.
5. Conclusions and Limitations

This paper presented an RL-based water heater control approach that aims to minimize the electricity cost of a water heater under a TOU electricity pricing policy. A set of RL agents were trained using the DQN algorithm and their performance was tested on an unseen pair of price and hot water usage profiles for 30 days. In addition, the performance of the developed RL agents was further validated through comparison to a set of rule-based, MPC-based and optimization-based controllers for five days. The comparison between RL agents and other controllers was conducted for five days because the MPC-based controllers require an optimization in each control time step. Thus, longer simulation periods result in longer computational times. Additionally, because the optimization-based controller optimizes the operation for the entire simulation period, longer simulation periods would result in more variables, which makes the optimization more challenging.

The results showed that the proposed approach can help save electricity cost significantly without any prior knowledge about the device. The RL agents were able to outperform the rule-based and MPC-based controllers and saved electricity cost in the range of 19% to 35% compared to the baseline operation. Additionally, the RL agent with 2 h of future hot water usage volumes and electricity prices information (RL agent #4) achieved a very close performance to the golden standard (i.e., the optimization-based controller).

Two main limitations are acknowledged. First, the proposed approach was tested on a water heater simulator. Although the simulator used in this study was validated using an actual water heater, the results and outcomes of this study are still based on a simulator. Additional field studies, therefore, need to be conducted in future work to see if/how the experimental results and findings—in terms of cost saving performance and comparison between RL and MPC—will change for actual water heaters. Second, for all MPC-based and optimization-based controllers, and RL agents #2, #3 and #4, it was assumed that future hot water usage volumes are known. However, future hot water usage volumes may not be known. Future research efforts could study the forecasting of future hot water usage volumes. In addition, additional experiments can be conducted to explore whether hot water usage patterns can be represented by a set of proxy variables (e.g., time of day).


Funding: This work was funded by the U.S. Department of Energy, Energy Efficiency and Renewable Energy, Building Technology Office under contract number DE-AC05-00OR22725.

Acknowledgments: This manuscript has been authored by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a nonexclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

Conflicts of Interest: The authors declare no conflict of interest.
Nomenclature
Abbreviations
BEMS Building Energy Management System
COP Coefficient of Performance
DHW Domestic Hot Water
DNN Deep Neural Networks
DOE Department of Energy
DP Dynamic Programming
DQN Deep Q-networks
DR Demand Response
FQI Fitted Q-iteration
GA Genetic Algorithms
GPM Gallons Per Minute
HPWH Heat Pump Water Heater
HVAC Heating, Ventilation, and Air Conditioning
MDP Markov Decision Process
MPC Model Predictive Control
RL Reinforcement Learning
RMSE Root Mean Squared Error
RTP Real-Time Pricing
TOU Time-of-Use
Symbols
\( \dot{Q} \) Heat added
\( m \) Mass flow rate
\( \Delta T \) Temperature change
\( m \) Water mass
\( s1 \) Heat pump cycle
\( s2 \) Lower element cycle
\( s3 \) Upper element cycle
\( V \) Hot water usage volume
\( C \) Heat fraction
\( N \) Node
\( T \) Temperature
\( UA \) Standby heat loss coefficient
\( \lambda \) Electricity price
Subscripts
\( amb \) Ambient
\( E \) Element
\( HP \) Heat pump
\( set \) Setpoint
\( w \) Water

References


18. Tarragona, J.; Fernández, C.; de Gracia, A. Model predictive control applied to a heating system with PV panels and thermal energy storage. Energy 2020, 197, 117229. [CrossRef]


40. Brandi, S.; Piscitelli, M.S.; Martellacci, M.; Capozzoli, A. Deep reinforcement learning to optimise indoor temperature control and heating energy consumption in buildings. *Energy Build.* 2020, 224, 110225. [CrossRef]


