RIGA TECHNICAL UNIVERSITY

*Information Technology and Management Science*

ISSN 2255-9094 (online)
ISSN 2255-9086 (print)
December 2018, vol. 21, pp. 29–36
doi: 10.7250/itms-2018-0004
*https://itms-journals.rtu.lv*

# Deep Reinforcement Learning on HVAC Control

Ivars Namatēvs
*Riga Technical University & Turiba University, Riga, Latvia*

***Abstract* – Due to an increase in computing power and innovative approaches of an end-to-end reinforcement learning (RL) that feed data from high-dimensional sensory inputs, it is now plausible to combine RL and deep learning to perform smart building energy control (SBEC) systems. Deep reinforcement learning (DRL) revolutionizes the existing Q-learning algorithm to deep Q-learning (DQL) profited by artificial neural networks. Deep neural network (DNN) is well trained to calculate the Q-function. To create a comprehensive SBEC system, it is crucial to choose an appropriate mathematical background and benchmark the best framework of a model-based predictive control to manage the building heating, ventilation, and air conditioning (HVAC) system. The main contribution of this paper is to explore a state-of-the-art DRL methodology to smart building control.**

***Keywords* – Deep reinforcement learning, deep Q-learning, deep neural network, energy management system.**

## I. Introduction

Deep reinforcement learning (DRL) is power driven by developments in machine learning and refers to the nonlinear methods, including artificial neural networks trained by stochastic gradient descend (SGD) and backpropagation. In recent years, many configurations of DRL have been designed and implemented, which can understand surrounding environment and intelligently control agents through gathering and examining a wide variety of data produced in and around the functional environment.

One such ambient intelligence technology application is the energy efficiency enhancement of apartment and office buildings. There are studies that show that in Europe within buildings the primary energy usage accounts for 41 %, which is further divided into energy consumption by commercial buildings (14 %) and residential buildings (27 %) [1]. However, around 30 % of the energy used in building is consumed by heating, ventilation and air conditioning (HVAC) system. It means that HVAC is the main energy consumer in a building [2]. According to the analysis by Coherent Market Insights, the smart building market is expected to increase at a compound annual growth rate (CAGR) of about 18 % by 2025 [3]. Raising focus on safety and security coupled with increasing efficiency and emission reduction is expected to drive the growth of the smart building market [4]. The efficient process of HVAC systems largely depends on their control system and optimization parameters [5]. Therefore, the energy demand and consumption are directly related to building indoor environment, i.e., temperature setpoints, airflow, humidity level, window and door type, occupancy, etc., as well as the outdoor environment, mainly, by weather conditions [2]. Considering all above-mentioned characteristics of HVAC systems, a challenging task is to develop an accurate and effective control model of an energy management system (EMS) for buildings. Energy management in buildings is the minimisation of the energy required to maintain a desired minimum comfort level for the occupants [6]. Most of the EMSs of buildings are complex nonlinear systems, which are strongly influenced by climate conditions, building operating modes, and occupant time schedules, and should be controlled in a smart way [7].

The eco-efficiency control task of the EMS model is to keep the room temperature, illuminance level as well as $CO_2$ within a predefined comfort range, which can be satisfied with a set of different actuators. The goal of the EMS is to choose the actuator settings depending on building indoor comfort setpoints, numerical climate forecasting and control engineering to achieve necessary comfort requirements and minimise the energy costs [8], [9].

Perception of indoor comfort is related to several environmental factors such as lighting, appropriate temperature, and air quality [6]. To evaluate well-being, there are several comfort measures such as PMV index and adaptive comfort standard [8], [9]. For instance, the PMV index that is used most often is a coded numerical integer [-3,3], which evaluates qualitative thermal sensation for the occupants in such a sequence: cold, cool, slight cool, neutral, slight warm, warm, and hot. To control the comfort level of smart HVAC building systems, it is necessary to know the existing environment, factors that influence it and setpoints that determine a model, which has to be carried out in an intelligible manner.

The author advocates the use of the model-based predictive control (MBPC) approach to smart building energy control (SBEC) with the purpose of efficiently controlling the existing HVAC in commercial buildings [6]. In literature, there are physical and data-driven models for controlling HVAC systems [5]. Physical models are based on mass and energy balance integral-differential equations, but data-driven models should be used for on-line control of HVAC system. For instance, branch-and-bound algorithm [10] is used to find the value $x$ that maximises or minimises the real value function $V(x)$, where $f(x)$ is an objective function that is employed in the implementation of HVAC. In the past decades, the application of SBECs has profited by machine learning, especially by artificial neuron networks (ANN), fuzzy logic and more recently by reinforcement learning (RL) [11]. Another study [12] opens a wide range of methods that have been proposed to solve the building control problems, including linear and dynamic programming, game theory, fuzzy methods, particle swarm optimization (PSW).

Several articles have been published so far on optimization methods [13], control strategies [14] and modelling techniques [15] for building HVAC systems. K. Dalamgkidis et al. [8] have developed an adaptive RL controller based on RL technique

that considers thermal comfort of the occupants of the building, the indoor air quality, and energy consumption.

Autoregressive exogenous (ARX) models are used where their inputs are zone temperatures setpoints of the heating system and cooling system, and outputs are actual zone comfort temperature and energy power measurement or energy performance indicators [16].

Therefore, the formal problem of RL is facing a learning agent interacting with its environment over time to achieve the goal, maximise the value or at least optimise it. An agent must learn its behaviour through trial-and-error exploration and delayed rewarding within a dynamic environment. The problem can be solved by using dynamic programming methods and statistical techniques or by using space of behaviours to find the best performance in the environment [11]. DRL has evolved through intersection of RL and ANN.

To solve the RL problem, it is necessary to find the optimal sequence of actions over the prediction horizon [17]. On the other hand, this relates to RL problem that it is necessary to determine the optimal policy, which will collect maximum reward in the long run [10].

One study pointed out that faults or non-optimal control schemes could cause the malfunction of equipment or performance degradation from 15 % to 30 % in commercial buildings [12].

The literature on building energy modelling and forecasting focuses on three categories: long-term load forecasts for system planning, medium-term forecasts for system maintenance, and short-term modelling for daily operation and scheduling [7]. In conclusion, RL techniques are most suitable for the cost minimisation problems, as they are capable of learning optimal behaviour, while the global optimum is not known [16]. At present, there are many studies focusing on improving the accuracy as well as simplifying the building energy control models to make them suitable for on-line control and optimization.

| **Nomenclature Q-learning** | $w$ a weight factor |
|---|---|
| $a$ an action | $v_*(s)$ value of state *s* under the optimal policy |
| $s$ a state | $G_t$ return at time *t* |
| $r$ a reward | $\gamma$ a discount parameter |
| $t$ a discrete time step | $v_\pi$ an evaluation function for policy $\pi$ |
| $a_t$ an action at time $t$ | $v_\pi(s)$ expected return in the state *s* (value of state *s* under policy $\pi$) |
| $s_t$ state at time $t$ | $q_*$ an action value function |
| $r_t$ reward at time $t$ | |
| $\pi$ a policy (decision making rule) | |

| | |
|---|---|
| $\mathcal{S}$ a state space (set of all nonterminal states) | $w$ weight |
| $\mathcal{A}$ an action space (set of actions) | $\mathcal{L}(w)$ a mean square error in Q-values |
| $\mathcal{T}$ a transition function | $V$ a state value |
| $\mathcal{R}$ a reward function (set of all possible rewards) | $V^\pi$ an evaluation function |
| $P$ probability | $S(t)$ a random variable of the state |
| E expectation | $R(t)$ a random variable of the reward |
| | $A(t)$ a random variable of the action |

The rest of the paper is organised as follows. In Section II, the RL framework with emphasis on Markov decision process is given as well as the mathematical ground of Q-learning and deep Q-learning principles has been described. In Section III, the main principles of deep reinforcement learning for HVAC are given. In Section IV, the proposed deep neural network architecture is shaped in the context of deep reinforcement learning. Conclusions and proposals for future research are formulated in Section V.

## II. The Reinforcement Learning Framework

There are two main elements of RL, agent and environment. Agent is the learner and decision maker, but environment – the thing it interacts with. The agent acts in an environment. Each time step the agent receives as the input current state $s_t$ takes action $a_t$ and receives reward $r_t$. After that, the agent receives the next input state $s_{t+1}$ and the next loop starts [17]. The agent chooses the action based on some policy $\pi: a_t = \pi(s_t)$. Figure 1 displays the basic RL scenario.

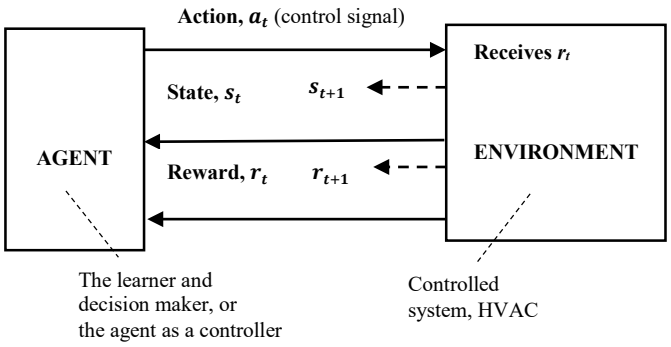


Fig. 1. The basic reinforcement learning scenario.

In HVAC problem, the environment (controlled system) is all those indoor and outdoor factors that influence a room or zone of the building. The agent manages a room or zone actuators of the building, accordingly, to specified setpoints.

In RL, the agent can be a passive learner or an active learner. *A passive learner* simply watches the world going by and tries to learn the utility of being in a various state. *An active learner* must also act using the learned information and can use its problem generator to suggest explorations of unknown portions of the environment [18]. Once the RL components are in place, it is possible to define the learning environment through the observations, actions and a reward function by means of Markov decision process (MDP) [19]. Traditional RL problem can be formulated as MDP.

*A. Markov Decision Process*

MDP or controlled Markov chain consists of four parts:

- *a state space* $\mathcal{S}$, $\forall s \in \mathcal{S}$;
- *an action space* $\mathcal{A}$, $\forall a \in \mathcal{A}$, where actions is $\mathcal{A}(s)$;
- the *transition function* $\mathcal{T}$;
- the *reward function* $\mathcal{R}$.

The state space is the set of all possible states of the system to be controlled. In the case of HVAC system, the state space is the set of *n*-vectors of values of the position of the heating and the cooling actuators, environment disturbances as well as the energy cost minimisation value. In each state, the controller of the system may perform any of a set of possible actions, e.g., heating high, heating low, airflow off, etc. States refer to the available information that is pertinent to the agent's decision making [20].

The actions are dependent on the given state *s* and denoted by $\mathcal{A}(s)$. Actions refer to the decisions [7].

The random variable denoting the state at time *t* is $\mathrm{S}(t)$, and the actual state at time *t* is $s_t$, whereas the following actual action is denoted as $a_t$. The state at time *t*+1, $s'$ depends upon the state at time *t*, $\mathrm{s}_t$, and upon the action $\mathrm{a}_t$ performed at time *t*.

This dependence is described by the transition function $\mathcal{T}$, so that $\mathcal{T}(s_t, a_t) = s'$, which is the new state at time *t*+1. Transition might be probabilistic, so that $\mathcal{T}(s, a)$ may return to a state sampled from a probability distribution over $\mathcal{S}$ [21], i.e., $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0,1]$. Since there is only a restricted number of states, we may define the probability from one state, let us say initial state *s* to next state $s'$, where $P_{s,s'}(a)$ is the probability that performing action *a* in state *s* will transform *s* into new state $s'$. This is:

$$\mathrm{P}_{s,s'}(a) = P\big(\mathcal{T}(s,a)\big). \tag{1}$$

Accordingly, the transition function can characterise the model of the agent-environment system, which is the following:

$$\mathcal{T}(s,a,s') \sim Pr(s'|s,a)\,. \tag{2}$$

To start with HVAC system modelling, the agent needs to use some parametric *model* of the controlled system (environment). At each time step, the agent receives the representation of the controlled system state $S_t \in \mathcal{S}$ and on that basis selects action $A_t \in A(t)$.

Finally, at each episode an agent (controller) receives a reward that depends upon the state *s* and the action *a* performed. The random variable denoting the reward at time *t* is $R(t)$, and the actual reward at time $t$ is $r_t$, i.e., reward received is a function of state at time *t* and action at time *t*. $R(t) = R(s_t, a_t)$ or $\mathcal{R}(s) \rightarrow \mathcal{R}(s,a) \rightarrow \mathcal{R}(s,a,s')$. Consequently, the reward function describes the expected reward being in a certain state or choosing a certain action while being in a specific state. The reward function looks only one step ahead [7]. It means that the reward is instant. Typically, instead of considering the reward function itself, the expectations are to be considered, which are written for fixed *s* and *a*:

$$R(s,a) = \mathrm{E}[R(s,a)]. \tag{3}$$

In other words, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$, where $\mathcal{R}_a(s,s')$ is the immediate reward received by the agent after it performs the transition to *s*' from state *s*.

As both transitions and rewards may be probabilistic, they depend upon the current state and the current actions, and there is no further dependence on previous state, actions, or rewards. Markov property is important for RL; it states that the environment dynamics must depend only on the current state and choose actions, thus enabling one to predict the next state and its expected reward only using currently available information and not entire history up to the current situation. This settles the definition of MDP.

In the case of HVAC problem, it is to minimise the total energy cost while maintaining the temperature, air conditions of each zone within the desired range and without fluctuation [22].

Referring to RL problem, the last important component is the policy, which is denoted as $\pi(s) \rightarrow a$. Policy defines the way RL agent behaves [7]. Policies can be:

- *Deterministic*, specifying which action should be taken under each state;
- *Stochastic*, i.e., probabilities of several actions are given.

In HVAC case, the agent aims at optimising a stochastic policy [25], i.e., $\pi : \mathcal{S} \times \mathcal{A} \times \mathcal{R} \rightarrow \mathbb{R}_+$

In general, there are three methods of policy assessment, i.e., value iteration, policy iteration, and policy search. Value iteration starts with a random *value function* and updates to an improved value function in an iterative process until reaching an optimal value function [17]. Among them Q-iteration (model-based), Q-learning (model free) are most widely used. Dynamic programming and temporal difference methods rely heavily on the notion of the value functions for solving RL problems [18].

*Policy iteration* evaluates policies by constructing their value functions and uses these value functions to find the improved policies [12]. Policy evaluation for Q-functions (model-based) and SARSA (model free) are mainly used. In the policy iteration-based scheme, an agent first computes the value function under the current policy (assuming a fixed or stationary policy). It means, first, to evaluate policy (a critic role); secondly, after policy evaluation the policy can be improved (an actor role). Methods for policy evaluation can be classified as follows: temporal difference methods, e.g., TD($\lambda$), SARSA, etc., and Monte Carlo policy evaluation.

A policy that specifies the same action each time a state is visited is termed *stationary* policy [17]. A policy that specifies that an action be independently chosen from the same probability distribution over the possible actions each time a state is visited is termed a *stochastic* policy [18].

Finally, the *policy search* uses optimisation techniques to directly search for an optimal policy, where policy gradient (model-based) and greedy policy (model free) are mainly used. Policy search methods do not use value functions at all; instead, they use optimisation techniques [19], e.g., gradient methods or evolutionary methods [11]. Among the most successful methods to policy search is neuroevolution [23], which uses *evolutionary computation* to optimize a population of neural networks.

*B. Q-learning Algorithm*

There are two learning approaches with the solution of optimal control problems (using on-line measurements): *indirect learning* and *direct learning*. The latter includes such schemes as *value-function based learning*, e.g., Q-learning, *policy space learning*, e.g., genetic algorithms, policy gradient. With the class of value-function based schemes, two separate major classes are *policy iteration* or actor-critic learning, and *value iteration* [17].

Value iteration schemes are based on some on-line version of the value iteration recursion, e.g., Q-learning, deep Q-learning, deterministic Q-learning, double Q-learning, fitted Q-learning. To evaluate the appropriate Q-learning algorithm for HVAC problem, it is necessary to formulate the key concepts. These are values and value functions, actions, states, rewards and reward functions, policy, return, and discounting [17], [18].

*Value functions* determine the optimal policy of the system. Specifically, when an exact model of the environment is determined and available, the agent can control which action will result in the best successor state. The best successor state is defined as the one with the largest value.

Alternatively, in problems where a precise model of the environment is not available, the state-action value is used instead since it provides the means to selecting the actions, as it is in the case of HVAC problem. If the process is in state *s* and the policy $\pi$ is followed, the expected action will be *a* with the largest value reward.

*A return* is an actual reward received by an agent while following a certain policy $\pi$. A return is a random variable and is the discounted sum of rewards or cumulative reward in one whole episode:

$$G_t = \sum_{t=0}^{\infty} \gamma^t r_t \,, \tag{4}$$

where $G_t$ is a return following time *t*; $\gamma^t$ is a discount parameter at time *t*.

The aim of the agent is to maximise the rewards it receives [20] and finally come over to the highest reward score. The return refers to the total reward received or to the reward received after a small amount of time. The return can be used to update the value function. In order to overcome the problem when the return may reach infinity *discounting* is used [17].

There are three main methods of assessing future rewards that have been studied, i.e., total reward, average reward, and total discounted reward [18]. Total discounted reward is the simplest case, which will be used in HVAC problem [21]. The total discounted reward from time *t* is defined to be:

$$r_t + \gamma r_{t+1} + \gamma^2 r_{t+2}, \dots, \gamma^n r_{t+n}, \tag{5}$$

where $r_t$ is the reward received at time *t*; $\gamma$ is a discount parameter, a number between 0 and 1.

Discounting assigns greater weights to the immediate rewards and less to very distant ones [10]. The smaller the discount factor, the less we take care about long-term rewards.

If the process is in state *s* and the policy $\pi$ is followed, the *expected return* will be written $v_\pi(s)$, i.e.:

$$v_\pi(s) = \mathrm{E}[R(s,\pi,t) + \gamma R(s,\pi,t+1) + , \dots, \gamma^n R(s,\pi,t+n) + , \dots], \tag{6}$$

where $v_\pi$ *is the evaluation function* for a policy $\pi$.

As the value discounting is exponential, $v_\pi$ also satisfies the following equations (7) and (8) [21] for all *s*:

$$v_\pi(s) = R(s,\pi) + \gamma\, \mathrm{E}\big[v_\pi\big(\mathrm{S}(s,\pi,t+1)\big)\big]. \tag{7}$$

$$v_\pi(s) = R(s,\pi) + \sum_{s'} Pr_{s,s'}(\pi)\, v_\pi(s'). \tag{8}$$

Thus, in a finite-state problem, if R and P are known, the evaluation function $v_\pi$ can be calculated by solving a set of linear equations, one for each state. However, as the set is large, this is a time and resource consuming task, as it constitutes for high computational power. This problem can be declined to make the calculations more convenient for usage in the following means.

Having into account that the value function for actual state at time *t* is expected to be a discounted sum of rewards from a certain or any state over all possible episodes (iterations) it can be written as follows:

$$v(s_t) = \mathbb{E}_{a_{t+1\sim\pi} s_{t+1\sim P}} \sum_{l=0}^{\infty} \gamma^l r_{t+l}. \tag{9}$$

Thus, the value function is an indicator of immediate as well as future rewards. The expected return or *the value function under policy* $v_\pi(s)$ can be rewritten and is the prediction of the return value for any state:

$$v_\pi(s) = \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) || s_0 = s, a_0 = a, a_t = \pi(s_t). \tag{10}$$

Therefore, the value of state *s* under the optimal policy is the following:

$$v_*(s) = max_\pi v_\pi(s). \tag{11}$$

The Q-learning algorithm instead of the value function for optimal policy employs the *Q-function* or *state-action function*, whose argument is not only a state, as can be found so far, but also an action. This allows optimising not only the policy, but also the control policy. The expression for the *Q-learning function* looks like this:

$$v(s_t,\ a_t) \leftarrow r_t\ + \gamma \cdot v(s_{t+1}), \tag{12}$$

where $a_t$ is an action chosen at time $t$ out of the set of all possible actions $\mathcal{A}$.

Since the purpose of the system is to maximise the total sum of the reward, $v(s_{t+1})$ is replaced by $max_{a\in A}Q(s_t, a)$ and as a result, the following expression is obtained:

$$Q(s_t,\ a_t) = r_t\ + \gamma\ \cdot max_{a\in\mathcal{A}}\, Q(s_{t+1}, a). \tag{13}$$

The expected return from starting at $s_t$ the following policy $\pi$ for one step, i.e. taking action $a_t$ and then following policy $\pi$ is:

$$Q_\pi\,(s_t, a_t) = R(s, a) + \sum_s P_{s,s'}(a) v_\pi\,(s'). \tag{14}$$

This is much simpler to calculate than $v_{\pi*}$, for $Q_\pi(s, \pi(s))$ it is only necessary to look one step ahead from state $s_t$, rather to calculating the whole evaluation function of $\pi^*$. For model-free algorithms, the explicit model, e.g., $p(s', r|s, a)$ is required, whereas this is not needed for model-based algorithms

$$Q_\pi(s, a) = r(s, a) + \gamma\ max_{a\in\mathcal{A}} Q(f(s, a), a'). \tag{15}$$

The target *Q*-value:

$$Q_*(s, a) = max_\pi\,(s, a). \tag{16}$$

Thus, the optimal control policy $\pi$ is to choose an action *a* by:

$$a = argmax_{a\in\mathcal{A}} Q_*(s, a). \tag{17}$$

Thus, the basic idea behind many RL algorithms is to estimate the Q-function or action-value function using the Bellman equation as an iterative update.

$$Q(s_t,\ a_t) \rightarrow Q_* \text{ as } i\ \rightarrow\ \infty. \tag{18}$$

Then, the policy can be updated by:

$$Q''_*(s, a) =\ Q'_*(s, a) + \alpha\big[r_{t+1}\ + \gamma max_a Q_{(s',a)} - Q'_*(s, a)\big], \tag{19}$$

where $Q'_*(s, a)$ – an old value;
$\alpha$ – a learning rate;
$r_{t+1}$ – a reward;
$\gamma$ – a discount factor;
$max_a Q_{(s',a)}$ – an estimate of optimal future value.

*Q*-value is usually stored in a 2D table whose inputs are a state and an action. Provided that the representation of Q-function is tabular and the environment is Markovian, there is proof that the Q-learning algorithm converges [24]. In practice, this approach is impractical because the action-value function is estimated separately for each action, without any generalization [17]. It is common to use a function approximators to estimate the action-value function as a linear or non-linear approximators, e.g., neural networks, especially, when the state and action spaces are large and continuous.

### *A. Deep Q-learning*

In the continuous or high-dimensional state space, the discretization matrix, which is used in Q-learning as the transporter of the action value function, $q_*$, will inevitably lead to long iteration time and difficult convergence [24]. To avoid this problem, the deep Q-learning (DQN) and a deep neural network (DNN) approach is used to approximate the state-action value function *Q* with weights *w*:

$$Q_*(s, a) \approx Q(s, a, w). \tag{20}$$

This approximation is used to define the objective function by mean-square error in *Q*-values:

$$\mathcal{L}(w) = \mathbb{E}[(r + \gamma max_{a'} Q(s', a', w) - Q(s, a, w))^2]. \tag{21}$$

The following *Q-learning gradient* is calculated as follows:

$$\frac{\partial \mathcal{L}(w)}{\partial w} = \mathbb{E}[(r + \gamma max_{a_{t+1}} Q(s_{t+1,} a_{t+1,} w) - -Q(s_{t,} a_{t,} w) \frac{\partial Q(s_t, a_t, w)}{\partial w}]. \tag{22}$$

If the standard Q-learning algorithm is used in harmony with neural networks, it oscillates or diverges due to the fact that data are sequential [25].

## III. The Deep Reinforcement Learning for HVAC System

Three categories of HVAC or energy forecasting methods have been reported in the literature [5]. There are *physics-based* (white-box) models or mathematical ones. A lot of mature physic-based tools exist: EnergyPlus, ESP-r, and TRNSYS, Modelica.

*Data-driven* (black-box models or empirical). These models are known as purely data-driven models. Ma et al. [25] combined multiple linear regression and self-regression methods to predict the building monthly energy consumption. ARX is implemented to predict 1 h ahead building load. ANN is another popular method in building energy prediction for building operation and control.

*Combination of physics-based and data-driven* (gray-box or hybrid) modelling approaches. These models can be linear, non-linear, static, dynamic, explicit or implicate, discrete or continuous, deterministic or probabilistic, deductive, inductive or floating [26].

Real world control tasks like smart building automation are seldom deterministic due to the stochastic nature of the

environment, e.g., people, weather, and often involved continuous actions, such as temperature setpoints, supply air flow rate etc.

*A. Problem Formulation*

In the context of HVAC problem, the aim is to reduce the cost of energy as well as to optimise the energy performance indicators, e.g., heating, air flow, PMV index, performance efficiency factor [16].

HVAC problems should be considered as follows:

- Thermal comfort control problem (TCCP). The structure for studying direct TCCP is set up by specifying the error between predictive heating amount mean setpoint and feedback heating amount, predictive airflow amount mean setpoint and feedback airflow amount.
- Energy cost minimisation problem.

Building consists of *n* zones. $\mathbb{Z}$ denotes the set of zones, such that $Z_i \in \mathbb{Z}$, $\forall_i = \mathbb{F}$ representing the zone performance efficiency factor of zones to be analysed.

Let $E_i$ denote the total zone energy consumption, which can be calculated as follows:

$$E_i = P^+ - (P_f^- + P_a^-), \tag{20}$$

where $P^+$ is power generation;
$P_f^-$ is fixed power consumption;
$P_a^-$ is adjustable consumption.

For modelling and control purposes, the data should be collected every 15 minutes. The thermal dynamics of the HVAC model of each zone is the following:

$$T_j\langle t+1\rangle = f_j\left(T_{n_1}\langle t\rangle, \dots, T_{n_q}, F_j\langle t\rangle, R_j\langle t\rangle, T_o\langle t\rangle, T_s\langle t\rangle\right) + O + X, \tag{21}$$

where
$f_j$ is some unknown nonlinear function;
$T_{n_1}\langle t\rangle$ is a temperature in the *j*-th zone of the building at time *t*;
$T_o\langle t\rangle$ is the outdoor temperature;
$T_s\langle t\rangle$ is the indoor temperature;
*O* is the variable related to the occupancy;
*X* is the variable related to other factors;
$\{n_1, \dots, n_q\}$ is the set of zones related adjacent to *j*.

Each control module (CM) controls the heating amount $H_j$ and air flow rate $F_j$ to modulate the thermal comfort of the *j*-th zone. Typically for HVAC systems the PID (a proportional-integrated-derivative) controller is often used. The model of control of heating amount is:

$$H_j\langle t\rangle = \begin{cases} t_{min}, & \text{if } e_j\langle t\rangle < 20^\circ C \\ t_{max} - t_{min}, & \text{if } 20^\circ\text{C} \le e_j\langle t\rangle \le 24^\circ C \\ t_{max}, & \text{if } e_j\langle t\rangle > 24^\circ C \end{cases} \tag{22}$$

The model of air flow amount is:

$$F_j\langle t\rangle = \begin{cases} \alpha_j, & \text{if } e_j\langle t\rangle < 20^\circ C \\ (\varpi_j - \alpha_j), & \text{if } 20^\circ\text{C} \le e_j\langle t\rangle \le 24^\circ\text{C} \\ \varpi_j, & \text{if } e_j\langle t\rangle > 24^\circ C \end{cases} \tag{23}$$

To minimise the cost function subject to the thermal dynamics system requires optimisation over and only on the set of possible actions based on a given state where data are mapped by sensors.

Regarding the multi-objective optimisation problems, i.e., where environment can be demanded by certain temperature setpoints and airflow rate setpoint as well as energy cost minimisation, the reward is derived as a simple multiple-task joint reward with three components. The joint reward components could easily generalize the necessary action.

**Component 1:** Temperature reward and air flow amount.

$$r_{a_1} = \begin{cases} -\delta_{a_1^-}, & \text{if } \delta_{a_1^-} > 24^\circ\text{C}, \text{and if } \delta_{a_1^-} < 20^\circ\text{C} \\ \delta_{a_1^+}, & \text{if } \delta_{a_1^+} \in [20^\circ\text{C}, 24^\circ\text{C}]. \end{cases} \tag{24}$$

**Component 2**: Controlling the total energy consumption, defined in Equation 20, is done as follows:

$$r = \begin{cases} -3\zeta_2 + 4[\max(P_f^-) - \max(\tilde{P}_f^-)], & \text{if } \max(\tilde{P}_f^-) < \max(P_f^-) \\ -3\zeta_1 - 1, & \text{otherwise,} \end{cases} \tag{25}$$

where $\zeta_1$ and $\zeta_2$ are coefficients based on trial and error procedure and according to [24] empirically are $\zeta_1 = 40$ and $\zeta_2 = -50$.

The building HVAC system is operated to maintain a desired comfort temperature within each zone (room), based on current temperature and outside weather disturbances, i.e., temperature, solar radiation, relative humidity, and wind. The zone temperature at the next time step is only determined by the current system state and environment disturbances, and the conditioned air input from the HVAC system. It is independent of the previous state of the building. Therefore, the HVAC control operation can be treated as a Markovian decision process (MDP).

*B. Deep Q-Learning for HVAC*

To control HVAC system, a range of parameters can be tuned, e.g., hot water temperature in radiators, chilling, etc. The HVAC system consists of *n* zones. Each zone provides heating and air flow rate that can be chosen from multiple discrete levels.

Control actions are as follows:

- Heating power (radiator), positive values = heating [W/m$^2$].
- Cooling power (chilling), positive values = cooling [W/m$^2$].

Control variables or control signals referring to action:

- Heating with 3 settings (off, low, medium, high).

- Air flow with 3 settings (off, low, medium, high).
- Window control with 2 settings (open, close).
- Door control with 2 settings (open, close).

It means 36 possible actions. To simply HVAC system, the heating and air flow can be only used as actions. Thus, there are **9 actions.**

System states (state space): The optimal control action is determined based on the observation of the current system state. State contains information to decide on control actions, e.g., environment disturbances (room temperature, PMV index, occupancy, time of the day, season, weather conditions, energy consumption, number of computers, etc.). Control variables referring to the current system state have 7 states:

- Heating off.
- Heating low.
- Heating medium.
- Heating high.
- Air flow (cool off).
- Air flow (cool low).
- Air flow (cool medium).
- Air flow (cool high).

Reward function: The control goals can be expressed by signing the reward of each state and action pairs. The values are estimated by deep Q-learning method.

Building dynamics: The thermal environment evolution inside a building is a physical process and can be generally encoded in the form of transition probabilities, e.g., the probability that the room temperature increases 1 degree Celsius given a supply air flow rate.

Value function: The combination of all possible values of each feature in the state vector forms a large state space. For approximation of the Q-value the DNN is used. Value matrix can be used to map all possible combinations of state and actions.

The batch mode can be used. **Batch mode** means that the entire data set for learning is available from the start, as opposed to the on-line mode of the algorithms in which data are acquired sequentially while the learning algorithm executes [17].

## IV. The Proposed Deep Neural Network for HVAC System

The topology of the DQN is based on the multilayer neural network with input layer, three hidden layers, and output layer. The proposed neural network is to approximate Q-values. Gradient descend is used as an optimizer to learn a policy for an agent. The architecture of DRL is shown in Fig. 2.

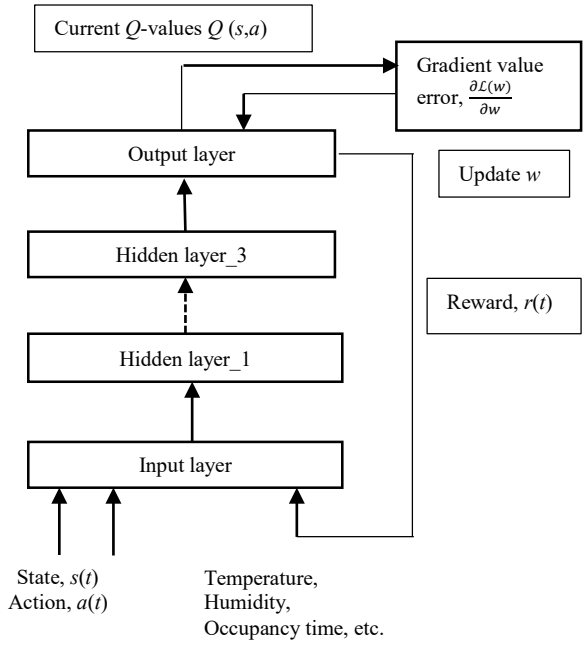


Fig. 2. The architecture of the proposed DRL.

The input state vector set layer consists of various types of features, i.e., indoor air temperature, outdoor air temperature, solar radiance, humidity, occupancy time schedule, etc. These features can characterise the influencers into and around the building and are given by a time-window of two following time steps. The state values are scaled to comparable range before feeding the input state to the neural network. For controlling energy consumption one extra neuron is added. Each hidden layer has 150 neurons with Rectified Linear Units (ReLUs) as activation functions. ReLUs apply the function $y=max(x, 0)$ and increase the nonlinear properties of the decision function and speed up the training of neural network. Simultaneously, they keep the gradient more or less constant [24].

The output layer represents the Q-value of the combined actions. Each combined action is possible combination of the actions, i.e., heating medium, air flow low, or heating off, air flow, cooling high etc. The hyper-parameters initially are time, $t$ = 15 min, the learning rate, $\alpha$ = 0,005, the discount factor, $\gamma$, number of episodes = 5000, range of input state vectors [0,1].

Finally, DQN is adapted to approximate state-action function. The neural network is constructed such as that it takes as input the states. The output consists of number of all possible actions. Each output is going to train to return Q-value for exact action. The system chooses the action for which Q-value is maximal and after action is completed the new state is given. The agent can learn and update policies directly from sensory inputs.

## V. Conclusion

The theory of RL provides an insight into the way the agents may optimise their control of an environment. This paper describes how RL is applied in optimal control of building energy consumption through management of zone actuators. Deep reinforcement learning with optimization method deep Q-learning has been proposed. Data-driven model based on neural network is proposed to be used for the on-line control of HVAC. Deep neural network is proposed to approximate the state-action value function $Q$ with weights $w$. The feasibility and robustness will be demonstrated experimentally on real data provided by a building management system of one of the leading companies in the field of microclimate of buildings in Latvia. Future work will focus on the improvement of the control system by making it aware of the room occupancy schedule, and on estimating more accurately the potential energy savings.

## References

[1] R. S. Smith, "Model Predictive Control of Energy Flow and Thermal Comfort in Buildings", *MPC Seminar, EPFL*, Lausanne, Switzerland, May 23, 2013.

[2] H. W. Lin, and T. Hong, "On Variations of Space-Heating Energy Use in Office Buildings", *Applied Energy*, vol. 111, pp. 515–528, 2013. https://doi.org/10.1016/j.apenergy.2013.05.040

[3] Coherent Market Insights. [Online]. Available: https://www.coherentmarketinsights.com/market-insight/advanced-energy-storage-market-746 [Accessed: Sept.2, 2018].

[4] N. Parks, "Energy efficiency and the smart grid," *Environmental Science & Technology*, vol. 43, no. 9, pp. 2999–3000, May 2009. https://doi.org/10.1021/es900771j

[5] Efficiency and the Smart Grid, *Environmental Science & Technology*, pp. 2999–3000, May 1, 2009. [Online]. Available: https://pubs.acs.org/doi/pdf/10.1021/es900771j [Accessed: Sept.2, 2018].

[6] Z. Afroz, G M. Shafiullah, T. Urmee, and G. Higgins, "Modelling Techniques used in HVAC Control Systems: A Review", *Renewable and Sustainable Energy Reviews*, vol. 83, pp. 64–84, 2018. https://doi.org/10.1016/j.rser.2017.10.044

[7] P. M. Ferreira, A. E. Ruano, S. Silva, and E.Z.E. Conceico, "Neural Networks Based Predictive Control for Thermal Comfort and Energy Savings in Public Buildings", *Energy and Buildings*, pp. 238–251, 2012. https://doi.org/10.1016/j.enbuild.2012.08.002

[8] X. Li, and J. Wen, "Review of Building Energy Modelling for Control and Operation", *Renewable and Sustainable Energy Reviews*, vol. 37, pp. 517–537, 2014. https://doi.org/10.1016/j.rser.2014.05.056

[9] K. Dalmagkidis, D. Kolokotse, K. Kalaitzakis, and G. S. Stavrakakis, "Reinforcement Learning for Energy Conservation and Comfort in Buildings", *Building and Environment* vol. 42, no. 7, pp. 2686–2698, 2007. https://doi.org/10.1016/j.buildenv.2006.07.010

[10] B. W. Olsen, and K. C. Parson, "Thermal Comfort Standards and to the Proposal New Version of EN ISO 7730", *Energy and Buildings*, vol. 34, no. 6, pp. 537–548, 2002. https://doi.org/10.1016/S0378-7788(02)00004-X

[11] F. Oldewurtel, A. Parisio, C. N. Jones, M. Morari, D. Gyalistras, M. Gwerder, V. Stauch, B. Lehmann, and K. Wirth," Energy Efficient Building Climate Control using Stochastic Model Predictive Control and Weather Predictions", Paper presented at the 2010 American Control Conference (ACC2010), 30 June – 2 July 2010, Baltimore, Maryland, USA. https://doi.org/10.1109/ACC.2010.5530680

[12] S. Whiteson, "Adaptive Representation for Reinforcement Learning", *Springer*, p. 133, 2010. https://doi.org/10.1007/978-3-642-13932-1

[13] M. Han, X. Zhang, L. Xu, R. May, S. Pan, and J. Wu, "A Review of Reinforcement Learning Methodologies on Control Systems for Building Energy", Working papers in transport, tourism, information technology and microdata analysis, Dalarna University, Nr. 2018:02, 2018. ISSN 1650-5581

[14] H. Belink, and A. H. Costa, "Batch Reinforcement Learning for Smart Home Energy Management", *Proceedings of the 24th International Joint Conference on Artificial Intelligence,* pp. 2561–2567, 2015.

[15] V. Mansur, P. Carreira, and A. Arsenio, "A Learning Approaches for Energy Efficiency Optimization by Occupancy Detection", *Internet of Things. User-Centric IoT*, pp. 9–15, 2015. https://doi.org/10.1007/978-3-319-19656-5_2

[16] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement Learning: A Survey", *Journal of Artificial Intelligence Research*, vol. 4, pp. 237-285, 1996. https://doi.org/10.1613/jair.301

[17] J. Ma, J. Qin, T. Salsbery, and P. Xu, "Demand Reduction in Building Energy Systems Based on Economical Model Predictive Control", *Chemical Engineering Science*, vol. 67, no. 1, pp. 92–100, 2012. https://doi.org/10.1016/j.ces.2011.07.052

[18] R. S. Sutton, and A. G. Barto, "*Reinforcement Learning: An Introduction*", 2nd ed., Cambridge, Massachusetts, London: A Bradford Book MIT Press, 2018.

[19] C. Szepesvari, "Algorithms for Reinforcement Learning", *Synthesis Lectures on Artificial Intelligence and Machine Learning series*, Morgan & Claypool Publishers, 2009.

[20] A. Juliani, V.-P. Berges, E. Vckay, Y. Gao, H. Henry, M. Motta, and D. Lange, "Unity: A General Platform for Intelligence Agents", 2018. arXiv:1809.02627v1.7Sep2018.

[21] A. Cahill, "Catastrophic Forgetting in Reinforcement-Learning Environments" M.S. thesis, University of Otago, New Zealand, 2010.

[22] C. J. C. H. Watkins, "Learning from Delayed Rewards", Ph.D. thesis, King`s Colege, London, 1989.

[23] T. Wei, Y. Wang, and Q. Znu, "Deep Reinforcement Learning for Building HVAC Control", *DAC`17*, June 18–22, 2017, Austin, TX, USA. https://doi.org/10.1145/3061639.3062224

[24] V. Heirdich-Meisner, C. Igel, "Neuroevolution Strategies for Episodic Reinforcement Learning", *Journal of Algorithms* vol. 64, no. 4, pp. 152– 168, 2009. https://doi.org/10.1016/j.jalgor.2009.04.002

[25] E. Mocanu, D. C. Mocanu, P. H. Nguyen, A. Liotta, M. E. Webber, M. Gibescu, and J. G. Slootweg, "Energy Optimization using Deep Reinforcement Learning", 2017. arXiv:1707.05878v1.

[26] J. Ma, J. Qin, T. Salsbury, and P. Xu, "Demand Reduction in Building Energy Systems Based on economic model predictive control", *Chemical Engineering Science*, vol. 67, no. 1, pp. 92–100, 2012. https://doi.org/10.1016/j.ces.2011.07.052

[27] A. Afram, and F. Janabi-Sharif, "Theory and Applications of HVAC Control System – A Review of Model Predictive Control (MPC)", *Building and Environment"* vol. 72, pp. 343–355, 2014. https://doi.org/10.1016/j.buildenv.2013.11.016

[28] A. Krizhevsky, I. Sutskever, and G. E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks", in *NIPS'12 Proceedings of the 25th International Conference on Neural Information Processing Systems,* vol. 1, 2012, pp. 1097–1105.

**Ivars Namatēvs** holds a *Mg. sc. ing.* from Riga Technical University and a MBA degree from Riga Business School. His research interests include deep artificial intelligence, especially reinforcement learning in machine learning, data mining methods and their application, genetic algorithms as well as foresight for artificial intelligence. The most important publications: I. Namatēvs. "Concept Analysis of Complex Adaptive Systems", *International Scientific Forum: Proceedings of XVI International Scientific Conference: Towards Smart, Sustainable and Inclusive Europe: Challenges for Future Development*. Riga, Latvia, 28–30 May 2015; Namatēvs, I., Aleksejeva, L., Poļaka, I. "Neural Network Modelling for Sports Performance Classification as a Complex Socio-Technical System", *Information Technology and Management Science,* vol. 19. pp. 45–52. 2016. Available from doi:10.1515/itms-2016-0010.; Namatēvs, I. "Exploring Model-Driven Domain Analysis for Software Engineering. *Proceedings of XVI International Scientific Conference.* Turība University, Riga, Latvia, 18 May 2017; Namatēvs, I., Aleksejeva, L. "Decision Algorithm for Heuristic Donor-Recipient Matching". *Mendel Soft Computing Journal*, vol.23, No.1, pp. 33–40, June 2017. ISSN:1803-3814.
E-mail: ivars@turiba.lv