

Article

Cooperative Multi-Agent Interaction and Evaluation Framework Considering Competitive Networks with Dynamic Topology Changes

Jinbae Kim  and Hyunsoo Lee * 

School of Industrial Engineering, Kumoh National Institute of Technology, Gumi 39177, Korea;
dbk0508@kumoh.ac.kr

* Correspondence: hsl@kumoh.ac.kr; Tel.: +82-(05)-44787661

Received: 31 July 2020; Accepted: 21 August 2020; Published: 23 August 2020



Featured Application: The proposed multi-agent framework can be applied to cooperative tasks between human and machines, such as human–robot interaction, autonomous car driving, and artificial intelligence-based industrial tasks.

Abstract: In recent years, the problem of reinforcement learning has become increasingly complex, and the computational demands with respect to such processes have increased. Accordingly, various methods for effective learning have been proposed. With the help of humans, the learning object can learn more accurately and quickly to maximize the reward. However, the rewards calculated by the system and via human intervention (that make up the learning environment) differ and must be used accordingly. In this paper, we propose a framework for learning the problems of competitive network topologies, wherein the environment dynamically changes agent, by computing the rewards via the system and via human evaluation. The proposed method is adaptively updated with the rewards calculated via human evaluation, making it more stable and reducing the penalty incurred while learning. It also ensures learning accuracy, including rewards generated from complex network topology consisting of multiple agents. The proposed framework contributes to fast training process using multi-agent cooperation. By implementing these methods as software programs, this study performs numerical analysis to demonstrate the effectiveness of the adaptive evaluation framework applied to the competitive network problem depicting the dynamic environmental topology changes proposed herein. As per the numerical experiments, the greater is the human intervention, the better is the learning performance with the proposed framework.

Keywords: adaptive algorithm; competitive network agent; dynamically changes networks; human–machine–agent interaction; reinforcement learning

1. Introduction

Reinforcement learning is concerned with the problem of maximizing the rewards of learning objects that need to be effectively controlled within a defined environment. The more complex the human's behavior and system configuration are, the more difficult the problem is, and the longer it takes for the learning object to learn. These problems can be solved through additional tasks such as pre-learning or preprocessing; however, these tasks are not very effective because preprocessing and pre-learning take a long time to complete and corrupt the learning data. Therefore, to effectively solve complex and difficult reinforcement learning problems, methods for solving problems through intuitive and professional human intervention have been proposed. The benefit of learning with the help of humans is that as the learning objects are intended to resemble humans, their learning goals can be clearly defined, and learning can be completed quickly without pre-learning or preprocessing.

However, existing studies on reinforcement learning focusing on learning problems with human help were concerned with the single model problem of a simple environment, static simple network and single agent.

As such, this study focuses on the two more competitive network topologies and it consists of multi-agent (e.g., combat or soccer) in a complex manner. To learn effectively, learners can model human evaluations in the form of quantitative real numbers to successfully achieve their goals and apply adaptive policies to humans. Herein, we propose a framework whereby the test results are precisely and strategically calculated.

The following section provides a review of existing research and literature related to human-machine evaluations and reinforcement learning. Section 3 proposes the cooperative human-machine evaluation framework and its algorithm. The implementation of the proposed framework and the performance analysis of the experimental example are presented in Section 4. Finally, the concluding remarks and future scope of work are presented in Section 5.

2. Background and Literature Review

This study applies reinforcement learning to the effective integration of human-machine evaluations toward a goal and compared previous studies on reinforcement learning on the basis of several categories. The learning object may be a robot, a system, or a game, and reinforcement learning is being investigated in various fields that require complex learning toward goals. There are a number of control strategies and relevant applications. Roman, et al. [1] proposed an adaptive control using fuzzy components to control a town crane. Zhang, et al. [2] applied a semi-global state synchronization method to actuator control under unknown nonuniform input delay.

When the learning object is a robot, these robots are generally classified as robot arms, humanoid robots, and industrial robots [3–5]. Robots with a high degree of freedom usually require complicated and difficult calculations because they require a goal that can perfectly mimic human behavior [6]. Additionally, robots learn how to behave similarly to humans so that they can also learn to expend a collaborative effort to help with human tasks or purposes [7]. The results from these studies can be applied to large-scale industrial applications that begin with learning and imitating simple movements, achieving goals through collaboration with humans, and applying them to real industries [8,9].

Reinforcement learning is also often used in areas such as games, where the goal of learning is clearly stated. This is because the algorithm of reinforcement learning is rewarded and updated according to the actions taken by the learning object. For example, when the object performs a mission in the game, the reward it obtains depends on its actions, which affects the final result and the target value [10].

Studies have also been conducted to learn systems, besides robots and games. System network and communication are complex and computationally heavy because of the need for a system that can optimize performance and goals. Furthermore, the computational load of networks that optimize paths such as an escape is heavy because it is necessary to derive the learning results by computing problems in real time. Therefore, learning methods that effectively deal with the real-time computation of a problem have been investigated [11–13].

Moreover, various methods have been studied regarding the problem of reinforcement learning to deal with values of measuring devices such as sensors. There are methods for learning via effectively estimating, interpolating, and approximating values before preprocessing, such as in the case of sensor values [14,15]. In addition, some studies have designed adaptive algorithms to ensure that learning objects are updated more reliably and effectively as they learn [16–18].

Table 1 summarizes the algorithms, contexts, and ideas proposed in previous studies and identifies the learning methods used in each study. These studies classified whether the learning object learned via the designed system itself or with human cooperation and also classified the learning objects into robots (robot arms and humanoids), games, or systems.

Table 1. Reinforcement Learning-Based Studies and Human–Robot Interaction.

Research Studies	Application	Learning Method	Human-Robot Interaction	System
Sheng [19]	Table-lifting task performed jointly by a human and robot	RL ^a	O ^c	Humanoid Robot
Lin [20]	The biped walking and balance control	RL	X ^d	Humanoid Robot
Tsurumine [21]	Learning of cloth manipulation by dual-arm	DRL ^b	X	Robot Arm
Breyer [22]	Object picking with a robotic manipulator	DRL	X	Robot Arm
Wang [23]	Collaborative robot to learn from human demonstrations about assembly tasks	RL	O	Robot Arm
Gu [24]	Door opening with a robotic manipulator	DRL	X	Robot Arm
Knox [25]	Training an agent manually evaluative reinforcement learning	RL	O	Game
Celemin [26]	Learning continuous actions from corrective advice communicated by humans	RL	O	Game
Kim [27]	Adaptive human–machine real number evaluation in dynamic competing network	RL	O	Game
Lee [28]	Human crowd evacuation problem using look-ahead-based reinforcement learning	RL	X	System
Le [29]	Complex agent-based model of evacuation	RL	X	System
Qiu [30]	Block chain-empowered mobile edge computing	DRL	X	System
Chen [31]	Mobile edge computing IoT networks	DRL	X	System

^a Reinforcement Learning (RL), ^b Deep Reinforcement Learning (DRL), ^c The manuscript handles human-robot interactions, ^d Human-robot interaction is not handled in the manuscript.

In relation to the existing studies that have explored mimicking of human behavior, research related to the task of table lifting performed by humans and humanoid robots has been conducted. Humanoid robots move by predicting human movements through prediction-based algorithms, and the reliability of the prediction is arrived at by the motion predictor [19]. A study was conducted on dynamically walking and balancing robots that use reinforcement learning to learn dynamic gait without prior learning. It aimed to solve complex control problems with respect to the robot's motion control by mapping the motion space from discrete to continuous areas. The balanced learning method, which used the movement of the robot arms and legs to move the zero-moment point in the robot sole, can keep the biped robot in a static stable state. It showed that the robot can learn how to improve motion in terms of the walking speed in the proposed way [20].

There have also been studies exploring flipping of handkerchiefs and folding of t-shirts. Emphasis was placed on the learning of the robotic arms, and deep reinforcement learning was investigated to learn complex policies through high-level observations such as typing. Because deep reinforcement learning requires a large number of training samples, a method that improves the sample efficiency and learning stability with fewer samples by combining the characteristics of smooth policy updates with automatic feature extraction of deep neural networks was proposed [21]. Similarly, there has been a study involving learning robots that pick up and classify objects. To be able to interact with dynamic objects in an unstructured environment, robots need manipulation capabilities to handle the confusion, change, and object variability. The robots learn a closed-loop policy that

maps depth camera inputs to motion commands and compare different approaches to make the problem easier to deal with, including the reward formation, curriculum learning, and the use of pre-trained policies with reduced work to pre-start tasks. Training the robots with heuristics helped achieve the desired behavior [22]. Collaborative robots are widely used in hybrid assembly tasks involving intelligent manufacturing. Research related to teaching-learning-collaboration models, where collaborative robots can learn through human demonstrations and support human partners in the working environment, has been put forth. This approach allows humans to control the robot using natural language instructions according to their personal work preferences. The robot then learns the assembly demonstrations from human using the maximum entropy inverse reinforcement learning algorithm and updates the task-based knowledge with the optimal assembly strategy. In the collaborative process, the robot can leverage the learned knowledge to actively support people in collaborative assembly work. As such, in the case of humanoid robot learning, an object is trained by imitating or pre-learning human behavior. Hence, pre-learning is essential at all stages of simulation or learning [23]. Robotic applications of reinforcement learning often undermine the autonomy of the learning process to achieve practical training time in real physical systems. To overcome this problem, recently developed deep reinforcement learning algorithms based on off-policy training for deep Q-functions can be extended to complex 3D manipulation tasks, and efficiently implement deep neural network policies to train the actual physical robot [24].

In contrast to the case involving interaction between the self-learning agent and the environment, it is recommended to train an agent manually using an evaluative reinforcement framework to update the rewards of human trainer feedback on the current state. Based on the evaluation of the agent's recent performance, the trainer can offer rewards in any form of representation that can be mapped to scalar values. The agent's goal is to act in consideration of the current state by receiving feedback from the human to choose the action that will receive the most rewards. For this purpose, the agent incorporates the reward function obtained from humans and selects the behavior that is expected to receive the highest reward. By learning a reward model for human feedback, agents can act on their goals even when there is no human feedback and choose a task that is expected to maximize their rewards when human feedback is provided. The agent attempts to maximize the immediate reward, assuming that the human trainer had already considered the impact of each behavior, when receiving feedback. This problem is consistent with supervised learning. Assuming each action is a training sample, for the selected action at time t , the state s_t at the current time t and the state s_{t+1} at the next time $t + 1$ are considered attributes of the sample, and the reward of the human trainer for that action is considered a label [25]. Delivered by the corrective advice communicated by human framework, a reinforcement learning method models new human feedback based on manually training agents through evaluative reinforcement framework. The author used a binary signal in the action domain of the agent. Further, the reward value was updated by appropriately utilizing past human feedback [26].

Existing studies to effectively learn through human evaluation mainly deal with simple, uncomplicated problems of a single network. In a recent study dealing with the situation where two or more network topologies compete and coexist, an effective algorithm is proposed in which human evaluation is adaptively updated [27]. In this paper, propose an algorithm that strategically updates rewards until a stable situation is reached, dealing with an environment where two or more dynamically changing networks compete with each other and adaptively update human evaluation.

Human evacuation frameworks used in emergencies have focused on modeling and simulating emergencies. Recently, reinforcement learning has been investigated for real-time shortest path calculation methods that can be used in emergency situations [28]. This method has both advantages and disadvantages in terms of two approaches: agent-based modeling and equation-based modeling. Because agent-based models are slow but accurate, studies are being conducted to produce fast linear models based on reinforcement learning. In addition, the formula-based modeling method is fast, but the range of erroneous measurements is rather large [29].

In-depth reinforcement learning methods have been studied for mining and processing large amounts of data in a dynamically changing environment [30]. Additionally, studies dealing with the resource allocation problems of a large amount of resources and devices have been conducted using deep neural networks to learn the environment and make decisions regarding the allocation problems according to network conditions, such as service latency and requirements [31].

Previous studies related to effective object learning methods for tasks that require human expertise remain useful to overcome the limitations of preprocessing learning, which requires large amounts of computation and numerous samples. Reinforcement learning has evolved in recent years in terms of its applicability to complex sequential decision-making tasks which are generally modeled using the Markov decision process (MDP). Reinforcement learning is a methodology for deciding the next action after inferring the reward based on the achievement of the goal, which is in turn based on the agent's current state and the interactions between the systems exhibiting the current state [32].

The typical learning method among traditional reinforcement learning algorithms is the Q-learning method which infers the maximized reward value by calculating the Q-function, which is a behavioral value function as shown in Equation (1), at each time period [33].

$$Q_{t+1}(s_t, a_t) = (1 - \alpha)Q_t(s_t, a_t) + \alpha(r_{t+1} + \gamma \max_a Q_t(s_{t+1}, a)) \quad (1)$$

In (1), s_t is the state at time t , a_t is the behavior of time t , r_{t+1} is the reward of time $t + 1$, and α is the learning rate in the range of (0,1). The closer the α is to 1, the more likely it is to induce learning to place a greater emphasis on the current situation and to adjust the share of rewards for future predicted behavior through a discount rate, γ [34].

3. Methodology of Cooperative Human–Robot Evaluation

This study is concerned with the problem of artificial intelligence soccer game shown in Figure 1. This is an example of a special case that differs from those previously investigated. In this paper, to deal with the environment in which two or more dynamically changing multi-agents in networks compete and coexist with each other, designed an artificial intelligence soccer game similar to the artificial intelligence basketball game covered in [27].

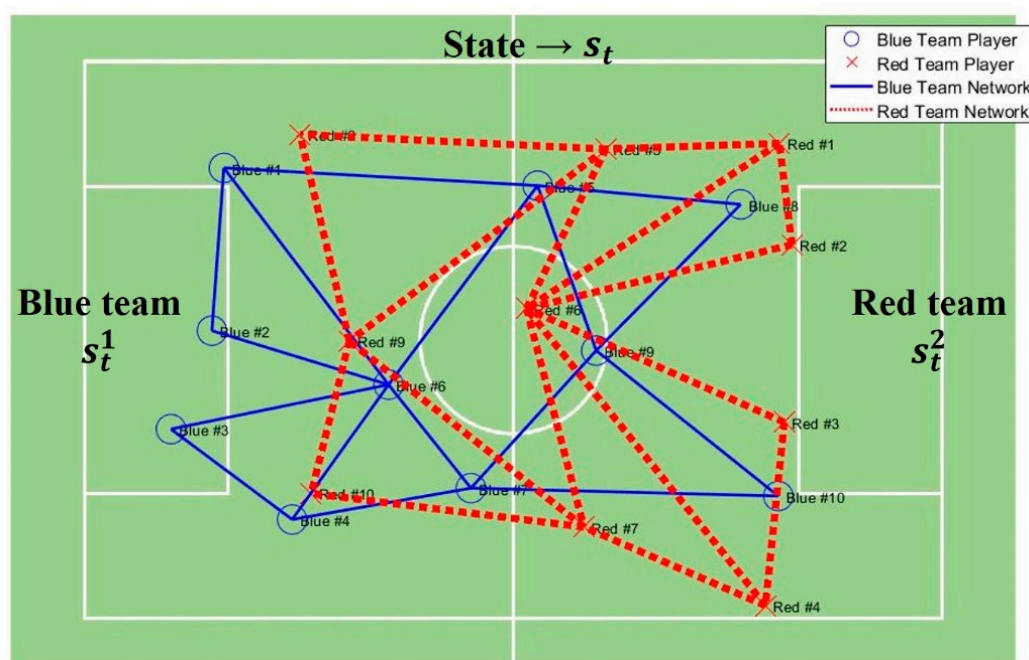


Figure 1. Dynamically Changes Multi-Agent in Competitive Network mapped with a Soccer Game.

The learning goal of the soccer game in this study is to pass the ball by avoiding the red team soccer players who are obstructed by the blue team soccer players. In real-life football games, players move dynamically on the field and try to pass the ball successfully to a player from the same team.

As such, players in football games described as multi-agent form a network topology as shown in Figure 1, and these network topologies show the two teams competing against each other. Each team is given a network topology at time t , and there is a state as a reference when it is expressed as a reinforcement learning problem considering the situation and environment, and there is another state that follows and competes similarly.

To deal with this reinforcement learning problem in this special case, this study defines the Q-Function as Equation (2) and calculates the maximum human evaluation reward value by calculating every time cycle period, t .

$$Q_{t+1}(s_t^1, s_t^2, a_t) = (1 - \alpha)Q_t(s_t^1, s_t^2, a_t) + \alpha(h_{t+1}^* + \gamma \max Q_t(s_{t+1}^1, s_{t+1}^2, a_t)) \quad (2)$$

State s_t^2 that coexists with state s_t^1 is not an independent state, but it is affected by the same action in the same environment. Therefore, it can be defined as Equation (3).

$$a_{t+1}(\vec{s}_t^2) = s_{t+1}^2 \quad (3)$$

Traditional reinforcement learning problems often involved a single network topology with the learning objects. However, this study covers two or more network topologies such as a dynamically changing competitive network. In a single network topology, the reward policy could be learned through a system reward update calculated from the learning object. Conversely, in this study, to establish a reward policy for a complex network in which two networks are learned in the same environment, the procedure for obtaining rewards through human evaluation and the reward of the system calculated from the learning object are properly applied. Here, a procedure is involved that updates rewards through human intervention, while existing studies [25,26] have mapped human evaluation through a binary process; moreover, this study advances evaluation feedback in a more quantitative and accurate form. The human rating is entered as a real value between 0 and 1, and the reward policy is updated in a much more accurate form.

The human evaluation reward, h_t , obtained in the process of learning a number of times, can be modelled by a Gaussian distribution as shown in Equation (4),

$$h_t \sim N(\mu, \sigma^2) \quad (4)$$

where μ is the evaluation mean, and σ is the distribution of standard deviation. In general, the mean of the Gaussian distribution is estimated \bar{h}_t using Equation (5),

$$\bar{h}_t = \frac{\sum_{i=1}^t h_i}{n} \quad (5)$$

where n is the sample size and is the number of quantitative rewards from human evaluation that has been learned for a number of times t . The standard deviation of the Gaussian distribution is estimated using Equation (6).

$$\hat{\sigma} = \sqrt{\frac{n \sum_{i=1}^t h_i^2 - \left(\sum_{i=1}^t h_i\right)^2}{t(t-1)}} \quad (6)$$

In repeated learning, when a correction value evaluated by a human is normalized through the distribution, as described above in Equation (6), \bar{h}_t is calculated and h_t is estimated. Subsequently,

the reward value obtained through the estimated human evaluation via repeated learning is subjected to a procedure for updating it adaptively, as shown in Equations (7) and (8).

$$G_{t+1} = G_t + \left(\eta \Delta F(\overline{h_{t+1}}) \right)^2 \quad (7)$$

$$\overline{h}_{t+1} = \overline{h}_t - \frac{\eta}{\sqrt{G_t + \epsilon}} \cdot \eta \Delta F(\overline{h}_t) \quad (8)$$

Here, Equation (7) can be used to minimize the loss function $F(\overline{h_{t+1}})$ that defines the difference between the estimate $\overline{h_{t+1}}$ during time $t + 1$ and estimate $\overline{h_t}$ the time t . When $\overline{h_{t+1}}$ is updated, the step size η should be set differently for each estimate iteration; hence, η is increased when the variation in the estimated human evaluation reward value is small, and η is reduced when the variation in the reward value through the estimated human evaluation is large. G_t stores the sum of squared values of $\overline{h_t}$ updated through estimation in time t . In the case of updating $\overline{h_t}$, the process proceeds to a size inversely proportional to the root of G_t in the existing step size η . The reason for this is that if the estimated human evaluation value $\overline{h_t}$ has changed significantly, it will move less. Conversely, if the estimated human evaluation value $\overline{h_t}$ has changed less, it will shift more. At this time, ϵ is a small value of between 10^{-4} and 10^{-8} , meant for preventing division by zero. As per Equation (8), the reward value, $\overline{h_t}$, adaptively updated by the human evaluation is compared with the human evaluation reward, h_t , obtained from learning in the present iteration. As shown in Equation (9), h_t is updated again via adopting the greater of the two values.

$$h'_t = \max(\bar{h}_t, h_t) \quad (9)$$

In Equation (9), the correction value h'_t updated adaptively by human evaluation should be appropriately calculated with the reward value of the system derived from the update of the learning object to determine the final reward value, h_t^* .

$$h_t^* = (1 - \delta)r_t + \delta h_t' \quad (10)$$

In Equation (10), h_t^* is calculated as the reward, r_t , of the system derived from the updating of the learning object, and the correction value, h_t' , adaptively updated via human evaluation, and it is calculated using the appropriate policy rate, δ , by the human intervene policy rate. On application of the quantitative evaluation of human being proposed in this study adaptively, the framework that established the system reward value and the appropriately calculated policy through the update of learning object is shown in Figure 2.

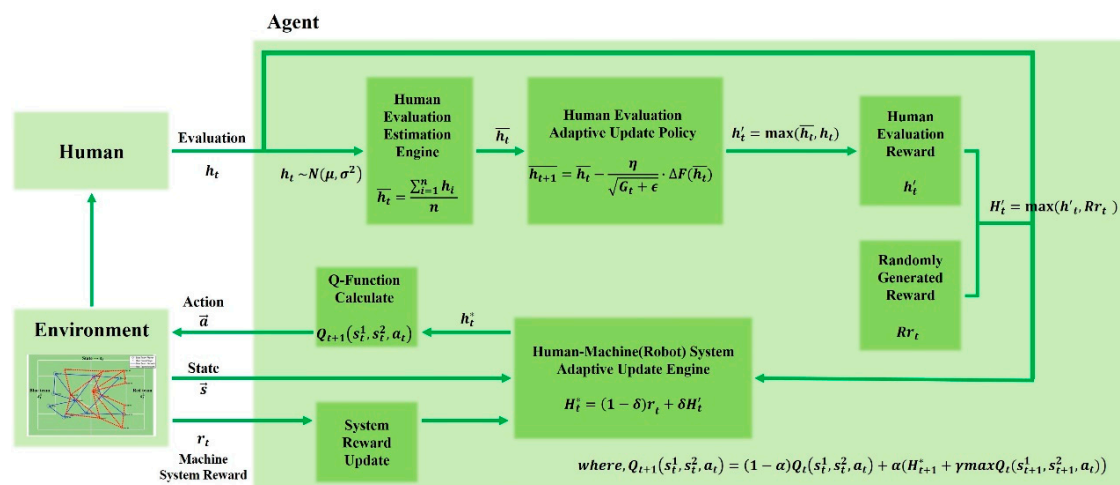


Figure 2. Reinforcement Learning Framework using Cooperative Human–Machine Evaluation.

Table 2 shows the complete algorithm involved in the proposed cooperative human–machine evaluation framework. First, the human intervene rate, δ , the discount rate, γ , and learning rate, α , are defined (lines 1–3). The agent observes the new state (line 6). Then, receive the exact quantitative evaluation h from the human (line 10), the update of the adaptive human evaluation reward (lines 11–15). Finally, when the reward for the Q-function is updated and determined, the Q-function is calculated (lines 17 and 18). This process is repeated for a given time, T , to proceed with the learning process (lines 4).

Table 2. Algorithm for Cooperative Human–Robot Evaluation Framework.

1:	$\delta \leftarrow \text{constant//human intervene rate}$
2:	$\gamma \leftarrow \text{constant//discount rate}$
3:	$\alpha \leftarrow \text{constant//learning rate}$
4:	for $1 \leq t \leq T$
5:	while true do
6:	$\vec{s} \leftarrow \text{getState}()$
7:	$\text{TakeAction}(\vec{a})$
8:	$\text{TakeSystemReward}(r_t)$
9:	wait for next time step
10:	$h \leftarrow \text{gethumanQuantitativePreciseSignal}()$
11:	$Rh_t \leftarrow \text{getRandomlyGeneratedReward}()$
12:	if $h \neq 0$
13:	Estimation for $h \sim N(\mu, \sigma^2)$
14:	$\bar{h}_t = \frac{\sum_{i=1}^n h_i}{n}$
15:	$h_{t+1} = h_t - \frac{\eta}{\sqrt{G_t + \epsilon}} \cdot \Delta F(h_t)$
16:	$h'_t = \max(\bar{h}_t, h_t)$
17:	$H'_t = \max(h'_t, Rh_t)$
18:	$\text{TakeFinalReward}(H'_t)$
19:	$H_t^* = (1 - \delta)r_t + \delta H'_t$
20:	$Q_{t+1}(s_t^1, s_t^2, a_t) = (1 - \alpha)Q_t(s_t^1, s_t^2, a_t) + \alpha(H_{t+1}^* + \gamma \max Q_t(s_{t+1}^1, s_{t+1}^2, a_t))$
21:	end if
22:	end while
24:	end for

4. System Implementation and Experimental Results

This section explains in detail the cooperative human–machine evaluation framework introduced in Section 3. The implementation of the proposed framework and the numerical analyses using the proposed software program are presented in this section. The program software is developed using MATLAB© (made by MathWorks, Natick, MA, USA) and C++ language. The implemented software program includes several panels and two graph windows.

The panels that make up the program deal with the network topology as shown in Figure 3. They form a network in two dimensions as shown in Figure 3a,b and in three dimensions as shown in Figure 3c and are expressed in a distribution to maximize visibility.

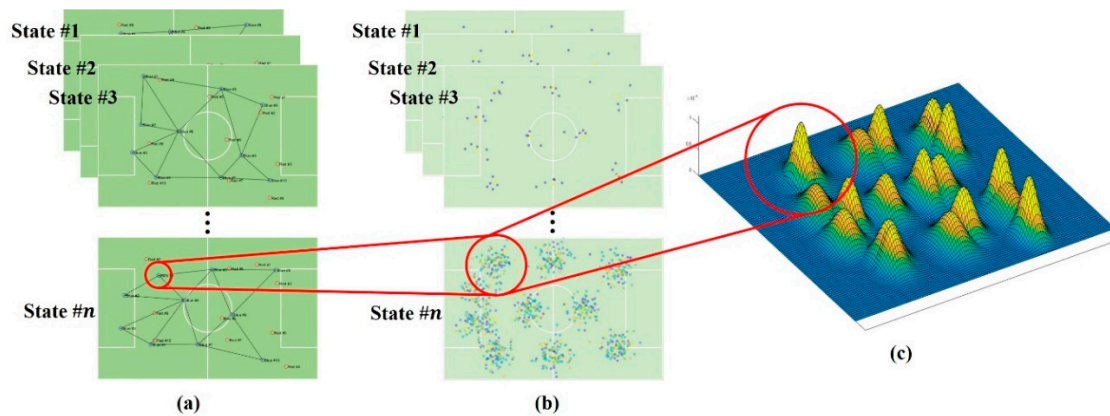


Figure 3. Players' (Agent) Network Topology in Soccer Game Problem, (a) Soccer Game Network Topology. (2-dimension) at time t , (b) Possible Actions and the Relevant Statutes (2-dimension), and (c) Distribution for Possible Actions and the Following Status.

The graph window and the windows that show the calculated values are shown in Figure 4. The windows have several functions. In a soccer game depicted in dynamically changing competitive network topologies as shown in Figure 4a, the players' status changes and the system's reward is calculated and shown.

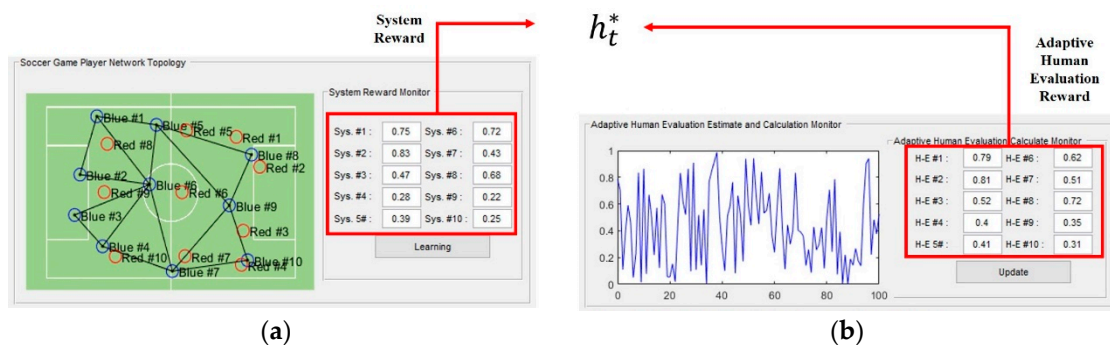


Figure 4. Final Reward of the Competitive Network with Dynamic Change Network Topology in Soccer. Game, (a) Automatically Calculated System's Reward and (b) Human's Feedback for Winning a Game.

At the same time, the user monitors two competitive network topologies depicted as a football game and inputs feedback on the behavior of the current state with a real number between 0 and 1.

The evaluation feedback from the human is calculated by estimation and adaptive reward evaluation calculation, as shown in Table 2.

Afterward, the reward from the system in Figure 4a and the human evaluation in Figure 4b are calculated according to the given level of human intervention, and the reward value according to the behavior of the learning object is updated.

Figure 5 shows a program that implements the proposed cooperative human-machine evaluation framework for the reinforcement learning of competitive network topologies. The panels and windows that make up the program are described in detail in Table 3. For instance, in a soccer game, there exist both network topologies: one and its enemy, coexisting in the same environment.

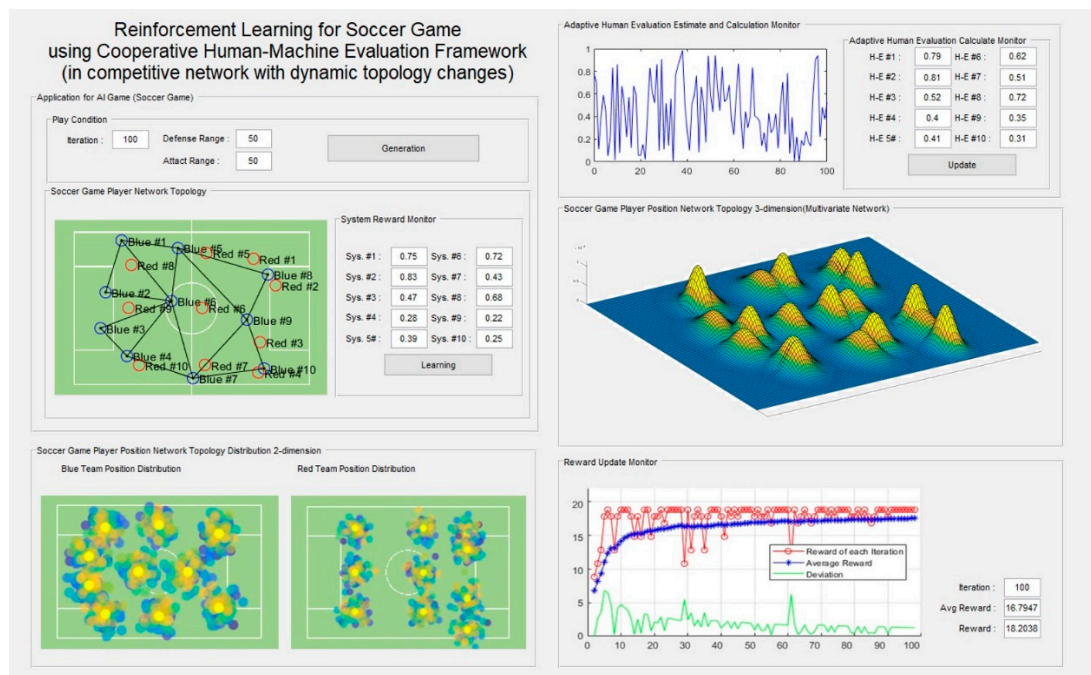


Figure 5. Implemented Soccer Game using the Proposed Cooperative Human–Machine Evaluation Framework Considering Competitive Network with Dynamic Topology Changes.

In this study, a soccer game was supposed, and an experiment was carried out through the example of learning to deliver the ball to allies while avoiding the enemy network that obstructed the flow of the ball. The players of both teams do not stay in a fixed position but have a distribution over a defined range and move dynamically; thereby, changing the network topology. The human evaluator visually confirms the degree of obstruction of the path for the ball to be delivered by the enemy and enters a real number between 0 to 1 accordingly. When a game agent passes a ball without any intervene, an expert (human) evaluates the pass closed to 1 using the developed system. The human’s evaluations are subjective and influenced by the locations of the opponents and other reasons. The evaluation score modifies the reward values using Equations (9) and (10). Then, the game agent changes the passing action to obtain higher reward using the provided algorithm shown in Table 2. In the iterative learning, the human evaluation reward is calculated and updated by the provided calculations of the reward value of the system itself and the human evaluation reward value by the cooperative human–machine evaluation framework.

Figure 6 shows the results of learning a soccer game represented by two networks that coexist using the proposed cooperative human–machine evaluation framework. There are two algorithms prepared for comparisons with the proposed cooperative human–machine evaluation framework.

Table 3. Implemented Cooperative Human–Machine Evaluation Software Program.

Type	Function	Detailed Function	Configurations
Panel	Application of competitive network (e.g., Soccer Game).	<ul style="list-style-type: none"> - Iteration/parameters' setting. - Defense/attack range. - Button generation. 	<ul style="list-style-type: none"> - Define an iteration - By defining the defense range and attack range, specify the range of the following actions.
Panel	Network topology.	<ul style="list-style-type: none"> - Monitoring players' network topology. - Monitoring system's reward. 	<ul style="list-style-type: none"> - Representation of a coexisting network topology only with current statuses of both players' group.
Panel	Players' statue network topology distribution (2-dimension).	<ul style="list-style-type: none"> - Monitoring both players' (e.g., Blue and Red team) network topology. 	<ul style="list-style-type: none"> - Representation of a coexisting network topology with current statuses and the following future statuses.
Panel	Players' state network topology distribution (3-dimension).	<ul style="list-style-type: none"> - Representing 3-dimensional network topology. 	<ul style="list-style-type: none"> - Representation of probability distributions depicting the following statuses.
Window	Adaptive human evaluation.	<ul style="list-style-type: none"> - Input window for human reward evaluation. - Calculation of adaptive human evaluation. 	<ul style="list-style-type: none"> - The network topology learned and evaluated by humans in each iteration - It is updated by the adaptive human evaluation strategy between the estimated values and the current evaluated values in the iteration.
Window	Reward calculation and updates.	<ul style="list-style-type: none"> - Analyzing integrated reward. - Measurement of system performance. 	<ul style="list-style-type: none"> - The rewards obtained for each learning are calculated - The average value is updated, and system performance is visualized.

The graph of Figure 6a represented by the symbol “o” signifies the reward of cooperative human–machine evaluation framework in each iteration as calculated by the algorithm of Table 2 and the graph of Figure 6a represented by the symbol “*” signify the value of the average value of the reward of a cooperative human–machine evaluation framework. Figure 6b is an algorithm in which the human evaluation is evaluated in binary (of 0 and 1). Moreover, the graphs of Figure 6b represented by the symbol “o” signify the reward of the simple evaluation in binary in each iteration, and the graph of Figure 6b represented by the symbol “*” signifies the average reward of evaluated in binary strategy. Figure 6c is an algorithm in which the traditional MDP method without human intervention. The graphs of Figure 6c represented by the symbol “o” means the reward of the MDP method in each iteration and the graph of Figure 6c represented by the symbol “*” signify the average reward of evaluated in the MDP method. As seen in Figure 6, the proposed cooperative human–machine evaluation achieved the fastest convergence.

This result is considered to be the result of human intervention that made quick decisions with specialized knowledge, unlike the way of updating only the reward of the system composed of existing learning objects. This study proposed a strategic method that aims to adaptively and quickly update the human evaluation scale and converge quickly to the maximum value.

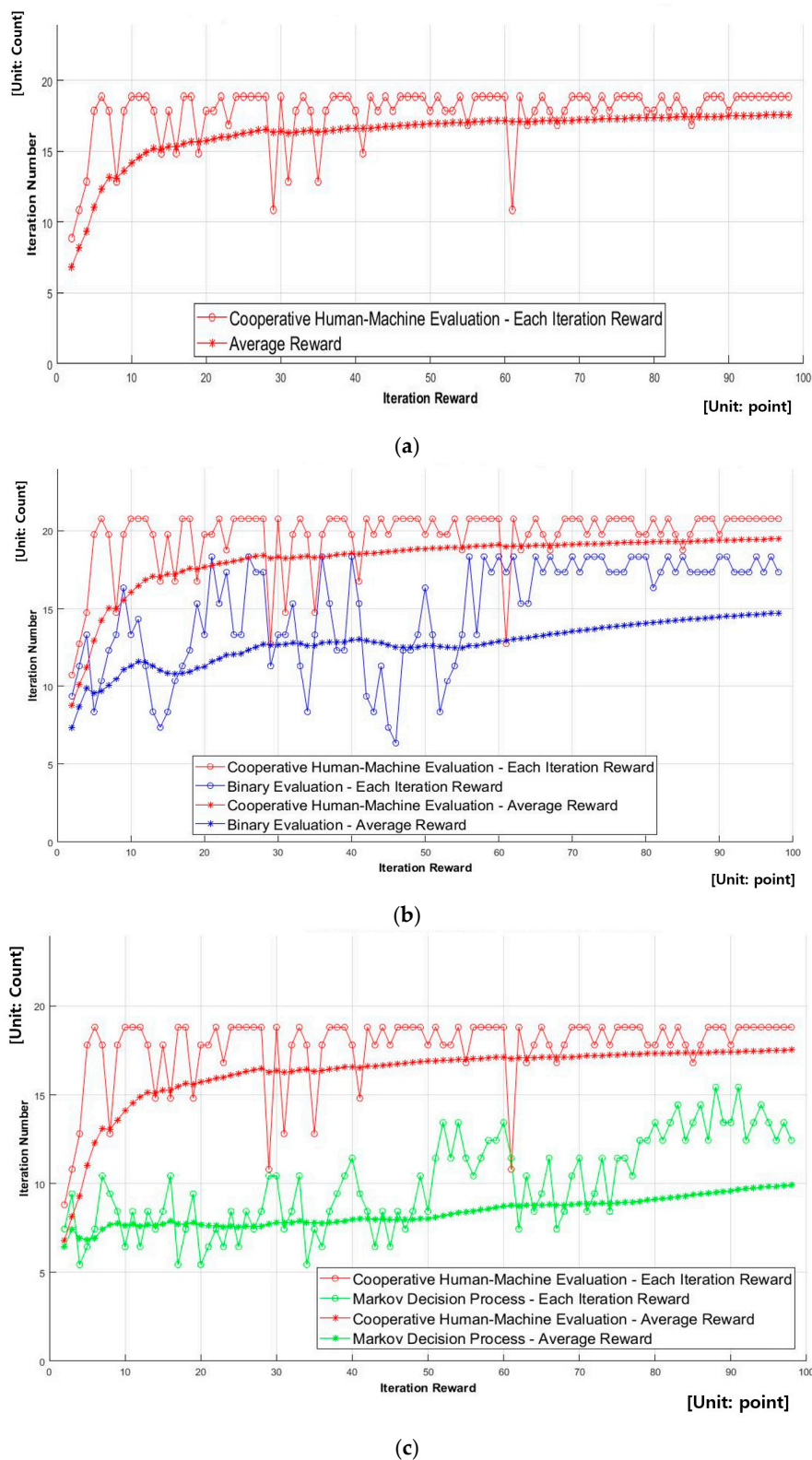
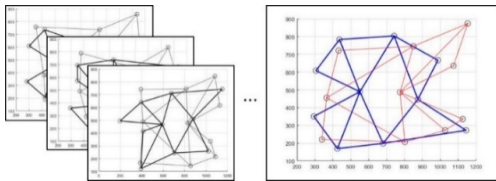


Figure 6. Comparisons of three evaluation methods, (a) the proposed cooperative human–machine. Evaluation framework, (b) human–machine evaluation with binary human evaluation, and (c) traditional Markov decision process with human evaluation.

Table 4 shows how the proposed cooperative human–machine evaluation framework differs in learning performance according to the degree of human intervention. The results are summarized

according to the parameter of the degree of human intervention rate δ . The problem addressed in this experiment is that two networks each have 10 nodes and coexist in the learning environment simultaneously. The problem is solved in three cases by applying the proposed cooperative human–machine evaluation framework. The first case solved the problem only with the system reward of the network topology existing inside the environment without any human intervention. As the training iterations were repeated, the reward value increased, and the system learned to converge to the maximum; however, the performance was poor and needed enough iterations. On the other hand, in the second case, if the policy for updating reward is updated by applying the reward calculated through the system’s internal reward and the proposed cooperative human–machine evaluation framework at a 50% rate, convergence tended to occur quickly to the maximum reward value. In the last case, where a 90% rate was applied, the highest Q-function value was obtained at the same number of learning while converging to the maximum reward value as the target. This is shown by comparison in Figure 7.

Table 4. Comparisons of three different experiment scenarios using the proposed framework.

 Number of agents in the same team on Network: 10 (Totally, 20 Nodes in each Network)			
	Case 1	Case 2	Case 3
Cooperation Type (δ = Human Intervention Rate)	No Cooperation (Independent System)	Cooperation ($\delta = 0.5$)	Cooperation ($\delta = 0.9$)
Average Reward	9.3045	21.3144	23.4633
Q-Function Value	12.3784	26.1066	27.1156

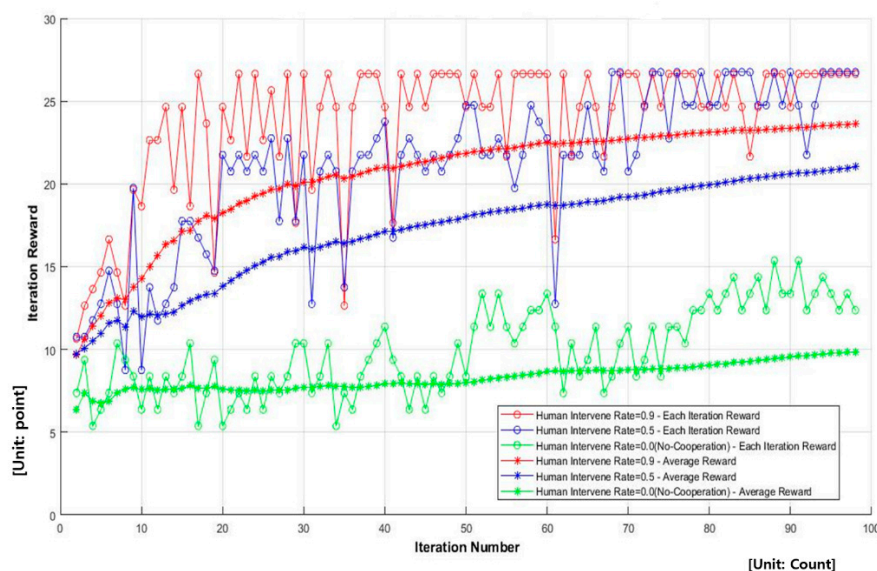


Figure 7. Comparisons of the proposed framework with three different human intervention parameters.

These results may vary depending on the assumed conditions and parameters. However, the proposed cooperative human–machine evaluation framework is effective in the complex network topologies in the context of human evaluation intervention using reinforcement learning.

In addition, a quantitative empirical study was conducted from two perspectives to show that the method proposed in this study is effective. As shown in Figure 8, when learning in three different ways during the same time, the time when the pass success rate of the Artificial Intelligence (AI) soccer game reaches more than 95% was compared. The result of learning the pass of players in the AI soccer game using the method of strategically updating the reward in the form of real number through human evaluation, the method proposed in this study, shows a pass success rate of up to 96.6%. The binary number reward method achieved a pass success rate of 86.6% within the same time, and when learning using the Markov Decision Process method, a pass success rate of 62.6% was achieved during the same time. Using the method proposed in this study, strategically updating the reward in the form of a real number through human evaluation, the time required for the pass success rate of players in the AI soccer game to reach 95% was $t = 29$. In contrast, the binary number reward method took $t = 73$ to reach the 95% success rate of the pass. When learning using the Markov Decision Process method, the time it takes for the pass success rate to reach 95% is $t = 93$.

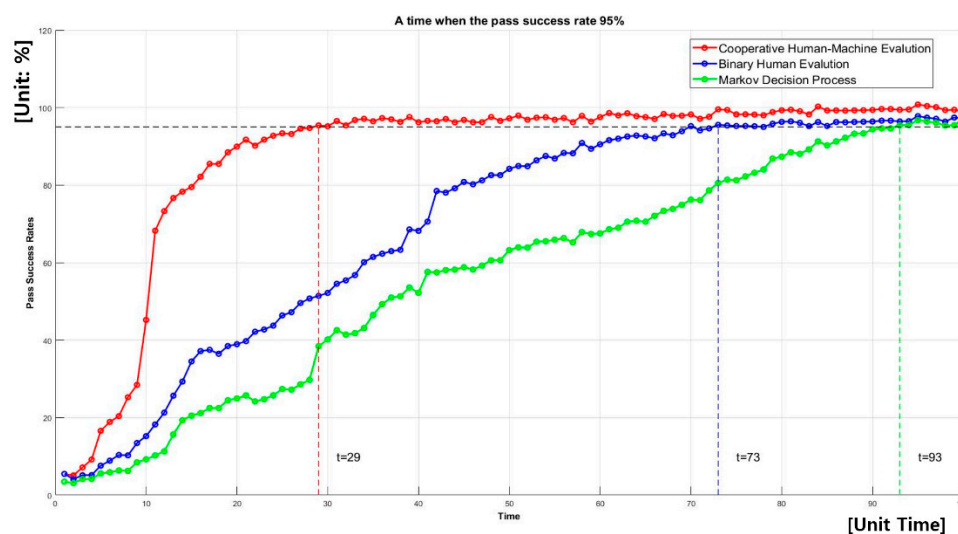


Figure 8. Comparisons among the proposed framework and the other existing methods.

5. Conclusions

As reinforcement learning is applied to several goal-oriented systems, the difficulty of problems and the complexity of the calculations increase, and various methods have been proposed to solve them. In the process of researching algorithms that solve these problems and lead to good performance, methods of learning are studied by adding accurate and quantitative evaluation through human intervention with expert knowledge and experience in reinforcement learning process. This trend has led to the need for pre-processing or pre-learning to make learning objects resemble human behavior or appearance in reinforcement learning. Existing studies involving human intervention at the same time as learning are relatively scarce and instead relied on prior learning.

This study proposed a new method of updating the rewards of the system obtained in the process of learning by learning objects and the rewards derived from the evaluation of humans with expertise on the problem. This framework also proposed an adaptive strategy to update rewards resulting from a stable and effective human evaluation. In addition to that, if the learning object was dealing with a simple and independent form, this study deals with the problem of complex network topology. This type of problem suggests the need for a cooperative human–machine evaluation proposed in this study. The proposed framework was implemented as a software program that supports the cooperative human–machine evaluation framework to demonstrate efficiency. Therefore, the effectiveness is demonstrated by comparing the results in various scenarios.

The proposed multi-agent framework can be applied to cooperative tasks between human and machines, such as human–robot interaction, autonomous car driving, and artificial intelligence-based industrial tasks. In particular, it contributes to fast learning process using multi-agents' evaluations. However, the provided framework is limited from the fact that human's evaluation interface and timings are crucial for faster training processes. In addition, various industrial tasks require different human evaluation methodologies. For this reason, the provided framework and its implementation have to be modified with the objectives of applications.

Further studies can consider the problem of determining more effectively the degree of human intervention required to update the combination of rewards via human evaluation and rewards from system learning. In addition, the provided framework can be applied to several real-life applications and scenarios. In a problem similar to the dynamic change competitive network topology discussed in this study, human intervention can be learned more quickly and efficiently.

Author Contributions: Methodology, J.K. and H.L.; software, J.K.; validation, J.K. and H.L.; formal analysis, J.K.; investigation, J.K.; writing—original draft preparation, J.K.; writing—review and editing, J.K. and H.L.; visualization, J.K.; and supervision, H.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Priority Research Centers Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science, and Technology (2018R1A6A1A03024003).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Roman, R.C.; Precup, R.E.; Bojan-Dragos, C.A.; Szedlak-Stinean, A.I. Combined model-free adaptive control with fuzzy component by virtual reference feedback tuning for tower crane systems. *Procedia Comput. Sci.* **2019**, *162*, 267–274. [\[CrossRef\]](#)
2. Zhang, M.; Saberi, A.; Stoorvogel, A.A. Semi-global state synchronization for discrete-time multi-agent systems subject to actuator saturation and unknown nonuniform input delay. *Eur. J. Control* **2020**, *54*, 12–21. [\[CrossRef\]](#)
3. Lasi, H.; Fettke, P.; Kemper, H.G.; Feld, T.; Hoffmann, M. Industry 4.0. *Bus. Inf. Syst. Eng.* **2014**, *6*, 239–242. [\[CrossRef\]](#)
4. Pech, M.; Vrchota, J. Classification of small- and medium-sized enterprises based on the level of industry 4.0 implementation. *Appl. Sci.* **2020**, *10*, 5150. [\[CrossRef\]](#)
5. Saif, S.; Rahmadani, F.; Lee, H. Implementation and simulation of cyber physical system for robotic arm control in smart factory. *J. Korean Inst. Intell. Syst.* **2019**, *29*, 308–315. [\[CrossRef\]](#)
6. Kartoun, U.; Helman, S.; Yael, E. A human-robot collaborative reinforcement learning algorithm. *J. Intell. Robot. Syst.* **2012**, *60*, 217–239. [\[CrossRef\]](#)
7. Kober, J.; Bagnell, J.A.; Peters, J. Reinforcement learning in robotics: A survey. *Int. J. Robot. Res.* **2013**, *32*, 1238–1274. [\[CrossRef\]](#)
8. Doltsinis, S.; Ferreira, P.; Lohse, N. A symbiotic human–machine learning approach for production ramp-up. *IEEE Trans. Hum. Mach. Syst.* **2017**, *48*, 229–240. [\[CrossRef\]](#)
9. Huang, B.Q.; Cao, G.Y.; Guo, M. Reinforcement learning neural network to the problem of autonomous mobile robot obstacle avoidance. In Proceedings of the 2005 International Conference on Machine Learning and Cybernetics, Guangzhou, China, 18–21 August 2005. [\[CrossRef\]](#)
10. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Petersen, S. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533. [\[CrossRef\]](#)
11. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Wierstra, D. Continuous control with deep reinforcement learning. *arXiv* **2015**, arXiv:1509.02971.
12. Stanley, K.O.; Miikkulainen, R. Efficient reinforcement learning through evolving neural network topologies. In Proceedings of the 4th Annual Conference on Genetic and Evolutionary Computation 2002, San Francisco, CA, USA, 9–13 July 2002; pp. 569–577.
13. Oh, E.; Lee, H. Development of convolution-based multi-directional and parallel ant colony algorithm considering network with dynamic topology changes. *Appl. Sci.* **2019**, *9*, 3646. [\[CrossRef\]](#)

14. Berz, E.L.; Tesch, D.A.; Hessel, F.P. Machine-learning-based system for multi-sensor 3D localisation of stationary objects. *IET Cyber Phys. Syst. Theory Appl.* **2018**, *3*, 81–88. [\[CrossRef\]](#)
15. Van Hoof, H.; Neumann, G.; Peters, J. Non-parametric policy search with limited information loss. *J. Mach. Learn. Res.* **2017**, *18*, 2472–2517.
16. Kussy, M.; Zajdel, R. Application of reinforcement learning algorithms for the adaptive computation of the smoothing parameter for probabilistic neural network. *IEEE Trans. Neural Netw. Learn. Syst.* **2014**, *26*, 2163–2175. [\[CrossRef\]](#)
17. Valasek, J.; Doebbler, J.; Tandale, M.D.; Meade, A.J. Improved adaptive–reinforcement learning control for morphing unmanned air vehicles. *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **2018**, *38*, 1014–1020. [\[CrossRef\]](#)
18. Kim, J.; Kim, S.; Lee, S. Pattern recognition and classifier design of bio-signals based interface in human-artificial intelligence interaction (HAI) framework for real time evaluation of emotions. *J. Korean Inst. Intell. Syst.* **2019**, *29*, 242–249. [\[CrossRef\]](#)
19. Sheng, W.; Thobbi, A.; Gu, Y. An integrated framework for human–robot collaborative manipulation. *IEEE Trans. Cybern.* **2014**, *45*, 2030–2041. [\[CrossRef\]](#)
20. Lin, J.L.; Hwang, K.S.; Jiang, W.C.; Chen, Y.J. Gait balance and acceleration of a biped robot based on Q-learning. *IEEE Access* **2016**, *4*, 2439–2449. [\[CrossRef\]](#)
21. Tsurumine, Y.; Cui, Y.; Uchibe, E.; Matsubara, T. Deep reinforcement learning with smooth policy update: Application to robotic cloth manipulation. *Robot. Auton. Syst.* **2019**, *112*, 72–83. [\[CrossRef\]](#)
22. Breyer, M.; Furrer, F.; Novkovic, T.; Siegwart, R.; Nieto, J. Comparing task simplifications to learn closed-loop object picking using deep reinforcement learning. *IEEE Robot. Autom. Lett.* **2019**, *4*, 1549–1556. [\[CrossRef\]](#)
23. Wang, W.; Li, R.; Chen, Y.; Diekel, Z.M.; Jia, Y. Facilitating human–robot collaborative tasks by Teaching-Learning-Collaboration from human demonstrations. *IEEE Trans. Autom. Sci. Eng.* **2018**, *16*, 640–653. [\[CrossRef\]](#)
24. Gu, S.; Holly, E.; Lillicrap, T.; Levine, S. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017. [\[CrossRef\]](#)
25. Knox, W.B.; Stone, P. Tamer: Training an agent manually via evaluative reinforcement. In Proceedings of the 2008 7th IEEE International Conference on Development and Learning, Monterey, CA, USA, 9–12 August 2008. [\[CrossRef\]](#)
26. Celemin, C.; Ruiz-del-Solar, J. COACH: Learning continuous actions from corrective advice communicated by humans. In Proceedings of the 2015 International Conference on Advanced Robotics (ICAR), Istanbul, Turkey, 27–31 July 2015. [\[CrossRef\]](#)
27. Kim, J.; Lee, H. Adaptive Human–Machine Evaluation Framework Using Stochastic Gradient Descent-Based Reinforcement Learning for Dynamic Competing Network. *Appl. Sci.* **2020**, *10*, 2558. [\[CrossRef\]](#)
28. Lee, H. Human crowd evacuation framework and analysis using look-ahead-based reinforcement learning algorithm. *Int. J. Digit. Hum.* **2016**, *1*, 248–262. [\[CrossRef\]](#)
29. Le, V.M.; Vinh, H.T.; Zucker, J.D. Reinforcement learning approach for adapting complex agent-based model of evacuation to fast linear model. In Proceedings of the 2017 Seventh International Conference on Information Science and Technology (ICIST), Da Nang, Vietnam, 16–19 April 2017. [\[CrossRef\]](#)
30. Qiu, X.; Liu, L.; Chen, W.; Hong, Z.; Zheng, Z. Online deep reinforcement learning for computation offloading in blockchain-empowered mobile edge computing. *IEEE Trans. Veh. Technol.* **2019**, *68*, 8050–8062. [\[CrossRef\]](#)
31. Chen, J.; Chen, S.; Wang, Q.; Cao, B.; Feng, G.; Hu, J. iRAF: A deep reinforcement learning approach for collaborative mobile edge computing IoT networks. *IEEE Internet Things J.* **2019**, *6*, 7011–7024. [\[CrossRef\]](#)
32. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*; MIT Press: Cambridge, MA, USA, 1998.
33. Greenwald, A.; Hall, K.; Serrano, R. Correlated Q-learning. In Proceedings of the 20th International Conference on Machine Learning (ICML), Washington, DC, USA, 21–24 August 2003; pp. 242–249.
34. Watkins, C.J.; Dayan, P. Q-learning. *Mach. Learn.* **1992**, *8*, 279–292. [\[CrossRef\]](#)

