applied sciences

MDPI

Article

A Modified Quad Q Network Algorithm for Predicting Resource Management

Yeonggwang Kim [1], Jaehyung Park [1], Jinyoung Kim [1], Junchurl Yoon [2], Sangjoon Lee [3,*] and Jinsul Kim [1,*]

[1] Department of ICT Convergence System Engineering, Chonnam National University, 77, Yongbong-ro, Buk-gu, Gwangju 500-757, Korea; yklovejesus@gmail.com (Y.K.); hyeoung@jnu.ac.kr (J.P.); beyondi@jnu.ac.kr (J.K.)
[2] Team of Energy Platform, Digital Transformation Department, Korea Electric Power Corporation (KEPCO), 55, Jeollyeok-ro, Naju, Jeollanam-do 58322, Korea; hiyoon@kepco.co.kr
[3] Interdisciplinary Program of Digital Future Convergence Service, Chonnam National University, 77, Yongbong-ro, Buk-gu, Gwangju 500-757, Korea
* Correspondence: s-lee@jnu.ac.kr (S.L.); jsworld@jnu.ac.kr (J.K.); Tel.: +82-62-530-1447 (S.L.); +82-62-530-1808 (J.K.)

Abstract: As the resource management systems continues to grow, the resource distribution system is expected to expand steadily. The demand response system enables producers to reduce the consumption costs of an enterprise during fluctuating periods in order balance the supply grid and resell the remaining resources of the product to generate revenue. Q-learning, a reinforcement learning algorithm based on a resource distribution compensation mechanism, is used to make optimal decisions to schedule the operation of smart factory appliances. In this paper, we proposed an effective resource management system for enterprise demand response using a Quad Q Network algorithm. The proposed algorithm is based on a Deep Q Network algorithm that directly integrates supply-demand inputs into control logic and employs fuzzy inference as a reward mechanism. In addition to using uses the Compare Optimizer method to reduce the loss value of the proposed Q Network Algorithm, Quad Q Network also maintains a high accuracy with fewer epochs. The proposed algorithm was applied to market capitalization data obtained from Google and Apple. Also, we verified that the Compare Optimizer used in Quad Q Network derives the minimum loss value through the double operation of Double Q value.

Keywords: demand response; resource management system; smart factory; smart appliances; reinforcement learning; Q-learning; fuzzy logic; Quad Q Network

Citation: Kim, Y.; Park, J.; Kim, J.; Yoon, J.; Lee, S.; Kim, J. A Modified Quad Q Network Algorithm for Predicting Resource Management. *Appl. Sci.* **2021**, *11*, 5154. https://doi.org/10.3390/app11115154

Academic Editors: Sang-Hyun Lee and Seongsoo Cho

Received: 30 April 2021
Accepted: 27 May 2021
Published: 1 June 2021

1. Introduction

Enterprise production systems are attracting serious concern across the economy due to their negative impacts on the industrial environment and market change. The global economy is experiencing an unprecedented demand for resources to seek for new investments with reinforcement and expansion of smart factory infrastructures and large adoption of renewable industrial resources [1]. The global economy is experiencing an unprecedented demand for resource seeking new investments for the reinforcement and expansion of smart factory infrastructures and the large adoption of renewable enterprise resources [2].

As a result, companies around the world are experiencing continuous global restructuring, enacting basic rules and laws for development direction and business expansion. A.I. programs are being introduced in some of the smart factories as resource options for reducing the supply of enterprise during certain periods of time balance supply and demand [3].

However, due to various factors, the number of data type parameters and the accuracy of artificial intelligence models are clearly limited for introducing them into the

field of practice [4]. Lack of quantity of datasets and errors in datasets as representative thresholds presents a serious obstacle to learning artificial intelligence. In this paper, we chose reinforcement learning as a solution to the problem of deficient datasets. The reward system for reinforcement learning varies from one learning system to another by weight [5]. We used an applicable DQN for fuzzy logic in this paper to demonstrate the uncertainty of reward schemes. Although the fuzzy logic was characterized by high levels of uncertainty, complexity, and nonlinearity [6], it was able to solve real-world uncertainty by simulating human experience in the form of rules [7]. Fuzzy logic was introduced by Zadeh. It is a mathematical discipline used to express human reasoning in terms of rigorous mathematical notations. Human beings can make decisions even in the presence of imprecise and incomplete knowledge. Fuzzy logic can facilitate the approximation of human reasoning, beginning with a set of user-supplied human actions. It then builds rules [8]. The fuzzy system can convert these rules into their mathematical equivalents. Additional benefits of the fuzzy logic include its simplicity and its flexibility that can aid in solving problems consisting of imprecise and incomplete data. It can also be used to model nonlinear functions of arbitrary complexities [9].

The potential of fuzzy logic has been explored in numerous fields of artificial intelligence, including signal processing, speech recognition, aerospace, robotics, embedded controllers, networking, business, and marketing. Moreover, fuzzy logic in reinforcement learning has been shown to be a promising technique due to its ability to efficiently facilitate the combination and evaluation of various parameters [10].

However, the reinforcement learning theory was mainly directed at improving the accuracy of the reward value, and there was insufficient evidence to minimize the loss value. The concern with these loss values is often associated with Overfitting issues, which creates further issues. In this paper, we find a solution to the problem when deriving the Loss value from the Double Dueling DQN algorithm. The output of the proposed Quad Q Network algorithm is derived from the Compare Optimizer task, which reduces the loss value. We also demonstrate our performance experimentally using two Sample Data, Total Resource on Google and Apple, and mathematical proofs of the derivation of Double Q value.

The rest of the paper is presented as follows. Section 2 describes previous researches in this field. Section 3 discusses the fundamentals of Dueling Deep Q Network and Double Dueling Deep Q Network as well as the proposed Quad Q Network. Section 4 presents the proposed experiment and results. Section 5 presents a summary of this work and future directions.

2. Literature Review: Related Work

2.1. The Latest Trends in Fuzzy Logic and Reinforcement Learning

In recent years, reinforcement learning has shown the ability to achieve good results in games such as fuzzy fields and robotic manipulators. It has become increasingly popular in the field of motion planning due to its characteristics of not requiring much labeling data. Fuzzy systems have found applications such as control engineering [11], medicine [12], and autonomous agents [13]. In recent years, some researchers have tried to combine reinforcement learning and fuzzy control to address several complex issues in the field of robot control [14]. Fuzzy systems typically have several control parameters that are outputted from the system. These parameters are then transformed into exact numerical values by diffusing to deal with dependencies between outputting different commands, thus providing a solution to improve the accuracy of predictions in Q-learning [15]. Deep reinforcement learning, which employs neural networks to infer state values, has received a great deal of interest.

It is capable of dealing with large complex unknown conditions [16–19], including very large state and activity spaces. Deep reinforcement learning has achieved numerous breakthroughs in complex spectrum access in recent years owing to its inherent advantages. The works in [20] may be regarded as the pioneer work in the investigation of Deep

reinforcement learning. It proposes a dynamic channel access scheme based on the Deep Q Network with experience replay [21]. The user selects one of the M channels to transmit its packet in each slot. Since the user only learns the channel state after selecting it, the related optimization decision problem can be expressed as a partially observable Markov decision method. Deep Q Network can promptly look for the best approach based on experience [22–24], thus it is commonly used in a variety of enterprises. Deep Q Network [25] is being investigated to solve both complex channel access and interference management. The Deep Q Network in [21] continues to pursue the learned policy over time slots and avoids learning a suitable policy.

Actual Reinforcement Environments Learning in the channel research is complex, and requires retraining of the Deep Q Network. To fix this problem, an adaptive Deep Q Network scheme [26] is proposed, which evaluates the cumulative incentives of the current policies over time. When the incentive falls below a certain level, the Deep Q Network is retrained to find new good policies. The simulation results show that when the channel state change, the scheme can detect it and begin relearning to achieve high returns. A Deep Reinforcement Learning system based on actor-critics is suggested, which can improve performance in a large number of channels [27]. Most of the methods listed above are limited to instances of only one user rather than multi-user or multiple channel setting, where collisions are common. Taking this into consideration, a fully distributed model [28] is used for users to perform dynamic multi-user spectrum access. For Dynamic Spectrum Access, a deep multi-user reinforcement learning method [29] is developed. This method can effectively represent the overall complex environment while avoiding the high computational costs incurred by the broad state space and inefficient observability. The spectrum access scheme proposed in [30] is based on Q learning and can assign idle spectrum holes without interaction with users. proposes multi-agent reinforcement learning based on Q learning, which can effectively avoid interference and achieve good sensing performance. Each radio in this model considers the actions of other radios to be a part of the environment [31]. Each radio tries to avoid transmissions from other wideband autonomous cognitive radios while also avoiding a jammer signal that sweeps through the entire spectrum band of interest. However, it is more difficult to achieve a stable policy in a volatile and complicated climate. To achieve stable policies in complex environments, research to reduce the loss value is needed, research in this area is growing [32].

2.2. Introduction to the Algorithms of Deep Q Learning

Deep learning network consists of numerous float values, weights, and biases [33]. These values determine the deep learning output, and the initialization process is required because these values are not defined before the learning begins. Although the most commonly used method is to initialize with random values, Glorot Initialization allows one to create an initial weight value with more stable results. This method is also called Xavier Initialization because it was proposed by Xavier Glorot. In Deep Q Network (DQN), the idea of target network is applied. If one looks again at the expression that updates the Q value in Q-Learning, one can see formula $Q(s,a) = Q(s,a) + \alpha(R + \gamma maxQ(s',a') - Q(s,a))$. The DQN has an idea called target network [34]. In the formula that updates Q value in Q-Learning, the α represents the learning rate. In the update formula, Q(s,a) is modified by multiplying $R + \gamma maxQ(s',a') - Q(s,a)$ by α. If α is 0, then $Q(s,a)$ will not change, and if the α is 1, then $Q(s,a)$ will be $R + \gamma maxQ(s',a')$. Because α is usually a value between 0.0 and 1.0, $Q(s,a)$ tends to $R + \gamma(maxQ(s',a))$ as the learning continues. Here, the problem of using network weights with the same goal as the current value arises if weights and biases of the Q Network are expressed as θ. The weight of the Q Network changes each time one learns. Goals change as well, thus, one might have difficulty in converging to a certain value. Therefore, we need to create a separate network with the same structure as the Q Network called target network, and mark it with a symbol θ^-, as in the formula $Q(s,a;\theta) \rightarrow R + \gamma maxQ(s',a';\theta^-)$.

The target network retains a fixed value during the original Q Network learning for a few times and periodically resets it to the original Q Network value [35]. This can be an effective way of learning because Q Network can be closer to a fixed target network. DeepMind's David Silver cited three major improvements after on the DQN, which he has presented to Nature in his fourth talk, "Deep Reinforcement Learning". They are Double DQN, Prioritized replay, and Dueling DQN. This paper introduces about Double DQN and Dueling DQN with direct calculation change. Double DQN was originally an algorithm using two Q Networks to improve the problem of Q-Learning in 2010, before DQN came out. Double Q-Learning shows that Q-learning is sometimes poor because Q values tend to be overvalued. If one looks closely at the $max\ Q(s', a')$ in the original Q-Learning expression, one can see that this value, given the status s', selects a', which has the highest Q value in the Q Network, and multiplies the Q value by gamma to create a target value for $Q(s, a)$.

However, the problem is that one has to use the same Q Network when selecting and importing Q values [36]. The selected Q value is usually large value. Once it is imported, the Q value goes in an increasing direction. However, if the Q value becomes larger than necessary, the performance of the Q Network will be greatly reduced. To prevent this, the first method proposed by Hadoopan Haselt is to use two Q Networks [37]. The key to Double DQN is that these two Q Networks can help prevent unnecessary growth of Q values. As we can see above, DQN already had the idea of target network.

The DQN of formula $Q(s, a; \theta) \rightarrow R + \gamma maxQ(s', a'; \theta^-)$ is replaced by the Double DQN of formula $Q(s, a; \theta) \rightarrow R + \gamma maxQ(s', argmaxQ(s', a'; \theta); \theta^-)$. Once one action is selected, using the existing Q network target network, Q value is obtained. Dueling DQN is the result of the network before one prompts for A value Q. A and V are then divided by the idea back together. The important thing is that one can calculate the V here. For each state by calculating the V, Q should be added as $Q(s, a) = V(s) + A(s, a) - \frac{1}{|A|} \sum_a A(s, a)$.

If you obtain a Q value, the Q value of the state with a higher value becomes a higher value, and the Q value of the state with a lower value becomes a lower value. Therefore, the agent moves to a state led by a high value and avoids the behavior of giving a low value. The idea of separating information about the value and the behavior has been similarly applied to the Quad Q Network algorithm presented in this paper. Unlike the Double DQN above, Dueling DQN changes the network structure. It is the same as combining into an output layer, which selects the behavior at the end. V and A are calculated separately in the middle. Finally the Q value is calculated to select the behavior.

3. Description of the Quad Q Network Algorithms and Functionalities

In the Deep Q Network, the idea of target network is the key. If one looks at the expression that updates the Q value in Q-Learning, it's the same as $Q(s, a) = Q(s, a) + \alpha(R + \gamma maxQ(s', a') - Q(s, a))$. In the Deep Q Network, the target is $(R + \gamma maxQ(s', a') - Q(s, a))$. The idea of Dueling DQN is to divide the result value of the network by V and A before obtaining the Q value and then merge it again. Figure 1 shows the combination of V (value function) and A (advantage) at the bottom.

The concept of Dueling DQN starts with splitting Q value into two. The existing Q value refers to the value when an action is taken at a given state, which is divided into two terms. One may wonder why we first divide then merge later. In order to see where the agent is interested in obtaining future rewards from a value stream perspective, to maximize future rewards, the agent ends up on the way to score and resource forecasting and resource forecasting. Based on the advantage stream, if no action is taken right away, one's reward is affected. Thus it is necessary to pay attention to the preceding value. If you dualize Q-learning into two streams, one can establish which state is valuable without having to learn the results by trying to act on each state. In a normal DQN, the agent must calculate the Q value for each action in a state when the value function is bad. Calculation of the Q value is done even in situations where the state is different or not best to get a reward. Despite performing all actions possible in the states, a bad results can be obtained.

The reason for dueling is because normal DQN calculates Q-learning with state and action pairs, If either of these values is bad, then calculating the Q value with them will only prevent us from learning the optimal policy, thus, the priorities are re-arranged priorities. This will reduce unnecessary computational time and allow learning with slightly higher quality samples. Figure 2 below shows a schematic diagram of the Dueling DQN algorithm written as a control in this study.

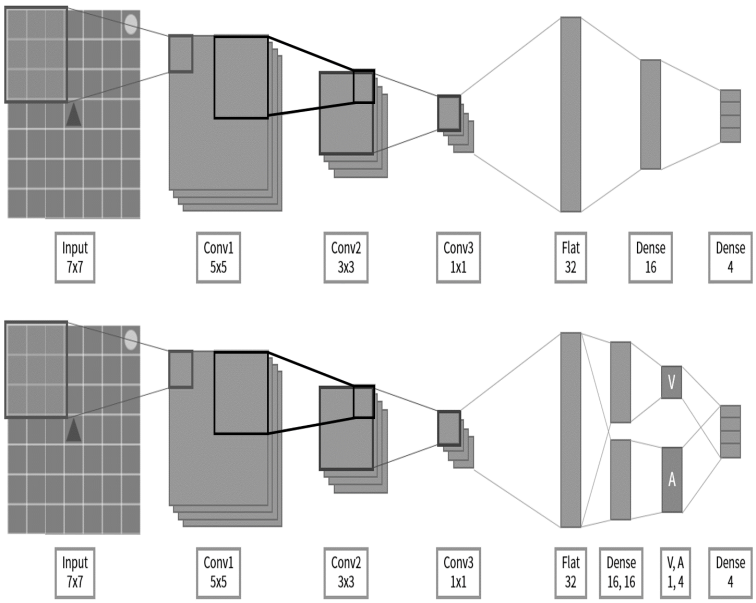

Figure 1. A figure showing the DQN network structure in Grid World. Below shows Dueling DQN.

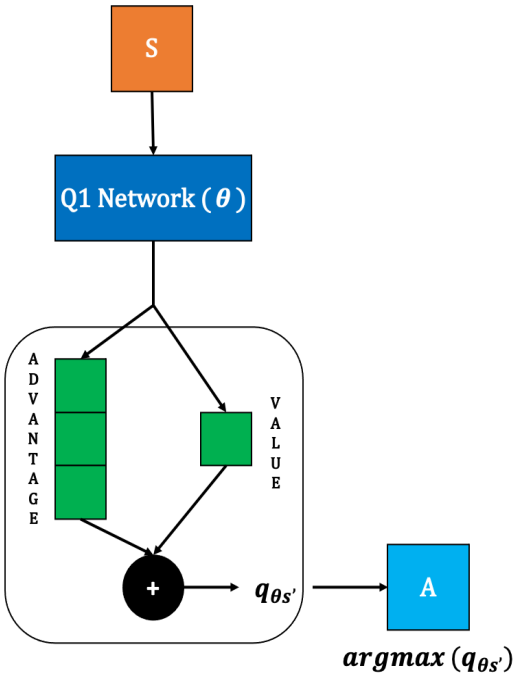

Figure 2. A figure showing Dueling DQN into two terms from the point of view of a Value stream and an Advantage stream.

In the Figure 2, $q_{\theta s'}$ assumes value as the following array (1), calculated from Value stream and Advantage stream.

$$q_{\theta s'} = \begin{bmatrix} 4.1 & 12.4 & 3.1 & 2.1 \\ 3.7 & 3.2 & 14.2 & 1.0 \\ 10.2 & 2.1 & 1.2 & 7.2 \end{bmatrix} \tag{1}$$

In addition, in Figure 2, the $argmax(q_{\theta s'})$ value can be presented as in the following array (2) values in terms of the weight for each row.

$$argmax(q_{\theta s'}) = \begin{bmatrix} 1 & 2 & 0 \end{bmatrix} \tag{2}$$

These Algorithms can fully play a role in increasing the accuracy of the existing Deep Q Network process. In addition, existing algorithms have the advantage of being easy to apply to Double Deep Q Network algorithms. The process is shown in Figure 3 and $q_{\theta' s'}$ was assumed to be array (3).

$$q_{\theta' s'} = \begin{bmatrix} 4.1 & 2.4 & 5.1 & 2.1 \\ 3.7 & 4.2 & 1.2 & 1.0 \\ 1.2 & 2.1 & 1.2 & 8.2 \end{bmatrix} \tag{3}$$

$argmax(q_{\theta s'})$ through $q_{\theta s'}$ is applied to $q_{\theta' s'}$, and $Double\ q_{s'}$ is derived using Formula (4).

$$Double\ q_{s'} = q_{\theta' s'}[range(batch_{size}), argmax(q_{\theta s'})] \tag{4}$$

The above $Double\ q_{s'}$ calculated using Formula (4) gives the following array (5) when it is calculated based on the example.

$$Double\ q_{s'} = \begin{bmatrix} 2.4 & 1.2 & 1.2 \end{bmatrix} \tag{5}$$

The calculated $Double\ q_{s'}$ is substituted in Formula (6), and the calculated Formula (6) is substituted in Formula (7) in order to calculate the final $loss_{s'}$.

$$Target\ q_{s'} = R + \gamma * Double\ q_{s'} \tag{6}$$

$$Loss_{s'} = [Target\ q_{s'} - q_{\theta s} * onehot(A)]^2 \tag{7}$$

The following calculated loss value can be clearly reduced compared to the one obtained before the verification process using the Q2 Network. Naturally, the performance varies widely depending on the parameter value, datasets feature, and the environment. However, the above algorithm clearly has a limit to simply determine the $argmax(q_{\theta s'})$ value through $q_{\theta s'}$ alone. Therefore, Quad Q Network suggested in this paper was created by duplicating Q1 and Q2, as shown in Formula $q_{\theta s'}$ and $q_{\theta' s'}$ were then computed separately before deriving a loss value using the Formula process shown above. The detailed Quad Q Network algorithm's drawings are shown as in Figure 4.

The Quad Q Network proposed in this paper does not reverse the structure of the existing Dueling Deep Q Network algorithm, thus, a similar precondition was established for the same Dataset, and the $Loss_{s'}$ value was calculated by substituting the idea of Double Deep Q Network derived only from one precondition, $q_{\theta s'}$. However, the comparison of two loss values to derive the loss value through the calculation of $q_{\theta' s'}$ can amplify the performance by selecting the side with less loss value. In addition, in Figure 4, the $argmax(q_{\theta' s'})$ value is presented by array (8) values as the weight for each row.

$$argmax(q_{\theta' s'}) = \begin{bmatrix} 2 & 1 & 3 \end{bmatrix} \tag{8}$$

$argmax(q_{\theta' s'})$ value through $q_{\theta' s'}$ is applied to $q_{\theta' s'}$, and $Double\ q_{\theta'}$ is derived from Formula (9).

$$Double\ q_{\theta'} = q_{\theta' s'}[range(batch_{size}), argmax(q_{\theta' s'})] \tag{9}$$

The above $Double\ q_{\theta'}$ calculated by Formula (9) gives array (10) when calculation is based on the example.

$$Double\ q_{\theta'} = \begin{bmatrix} 3.1 & 3.2 & 7.2 \end{bmatrix} \tag{10}$$

The calculated $Double\ q_{\theta'}$ is substituted in Formula (11), and the calculated Formula (11) is substituted in Formula (12) to calculate the final $loss_{\theta'}$.

$$Target\ q_{\theta'} = R + \gamma * Double\ q_{\theta'} \tag{11}$$

$$Loss_{\theta'} = [Target\ q_{\theta'} - q_{\theta s} * onehot(A)]^2 \tag{12}$$

The Quad Q Network, proposed by comparing initially computed $Loss_{s'}$ and $Loss_{\theta'}$, proceeds with the learning to derive performance improvements through lowering of the loss value.

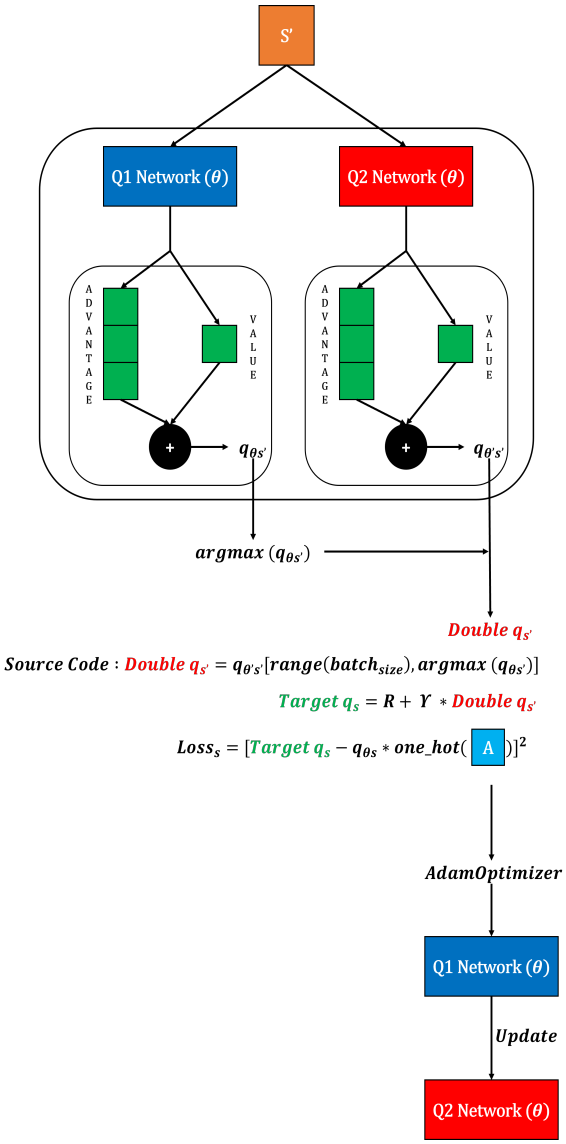

Figure 3. The process of extracting Target Q Value by adding Q1, Q2 Network, the idea of the Double DQN algorithm, to a Dueling DQN algorithm.

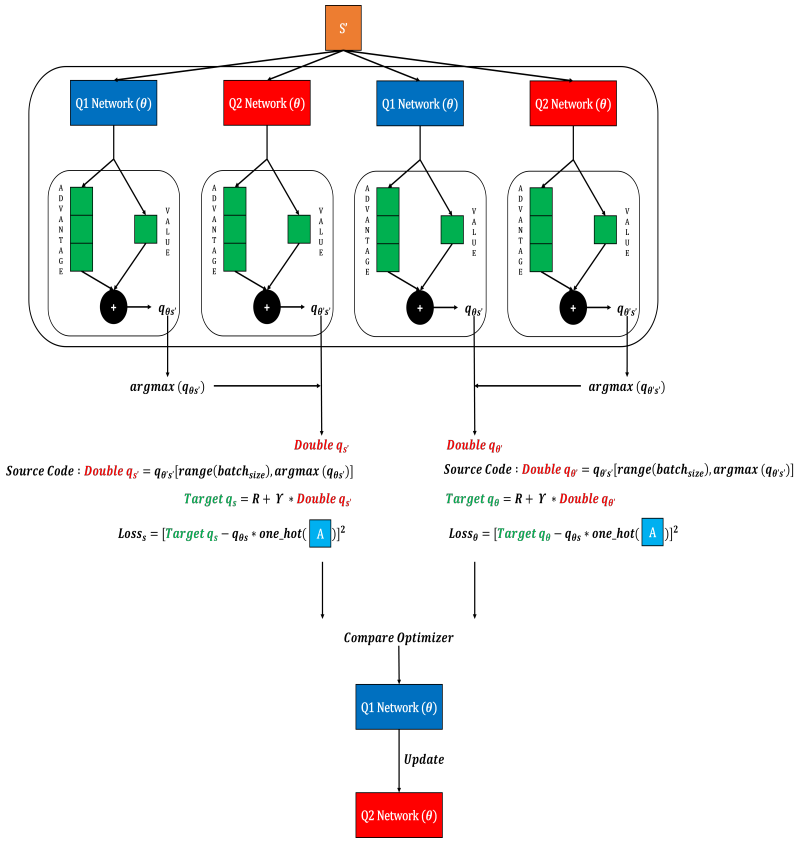

Figure 4. Quad Q Network plot created by duplicating Q1 and Q2, followed by two *argmax* processes and comparison of Loss values.

4. Experiment and Results

4.1. Datasets of Industrial Google Resource Value

In this paper, we conducted a study using Total Resource on Google from 28 March 2014 to 10 November 2017. The configuration of the datasets consisted of Date, Price, and Volume as shown in Table 1.

Table 1. Total Resource on Google from 28 March 2014 to 10 November 2017.

Date	Price	Volume
28 March 2014	559.99	41,003
31 March 2014	556.97	10,772
1 April 2014	567.16	7932
2 April 2014	567.00	146,697
3 April 2014	569.74	5,087,500
4 April 2014	543.14	6,377,600
7 April 2014	538.15	4,368,717
8 April 2014	554.90	3,148,563
9 April 2014	564.14	3,323,579
⋮	⋮	⋮
3 November 2017	1032.48	1,076,350
6 November 2017	1025.90	1,124,765
7 November 2017	1033.33	1,112,146
8 November 2017	1039.85	1,088,395
9 November 2017	1031.05	1,244,886
10 November 2017	1028.07	720,674

In order to visualize this dataset, it is presented as in Figure 5 below in order to visualize the overall value of Total Resource on Google.

Figure 5. A graph for visualizing the overall value of Total Resource on Google.

4.2. DQN, Dueling DQN, Double Dueling DQN, and QQN Learning Results on Google's Total Resource

Based on Table 1, we conducted a study in this paper. Datasets of Total Resource were obtained from Google, and a learning process was conducted by measuring the reward based on these datasets.

Figure 6 presents learning results by substituting the DQN algorithm based on the proposed Total Resource on Google. After the learning process, the minimum reward was −118, the maximum reward was 8, the maximum loss value was approximately 92 K, and the minimum loss value was 19.

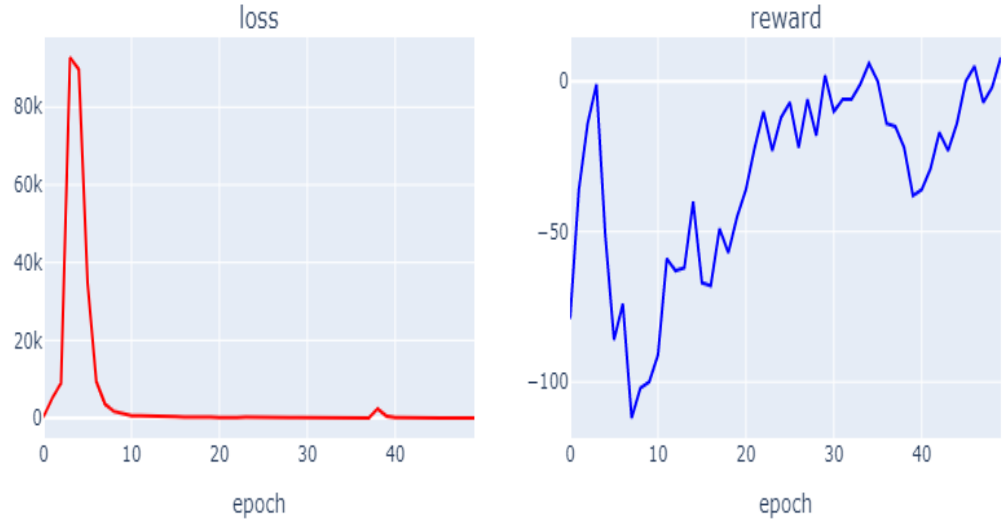

Figure 6. Learning results by substituting the DQN algorithm based on the proposed Total Resource on Google.

Figure 7 presents learning results by substituting the Dueling DQN algorithm based on the proposed Total Resource on Google. After the learning process, the minimum reward was −50, the maximum reward was 21, the maximum loss value was approximately 805, and the minimum loss value was 4.5.

Figure 8 presents learning results by substituting the Double Dueling DQN algorithm based on the proposed Total Resource on Google. After the learning process, the minimum reward was −44, the maximum reward was 28, the maximum loss value was approximately 8322, and the minimum loss value was 3.6.

Figure 9 presents learning results by substituting the Quad Q Network algorithm based on the proposed Total Resource on Google. After the learning process, the minimum

reward was −6, the maximum reward was 29, the maximum loss value was approximately 56, and the minimum loss value was 1.4.

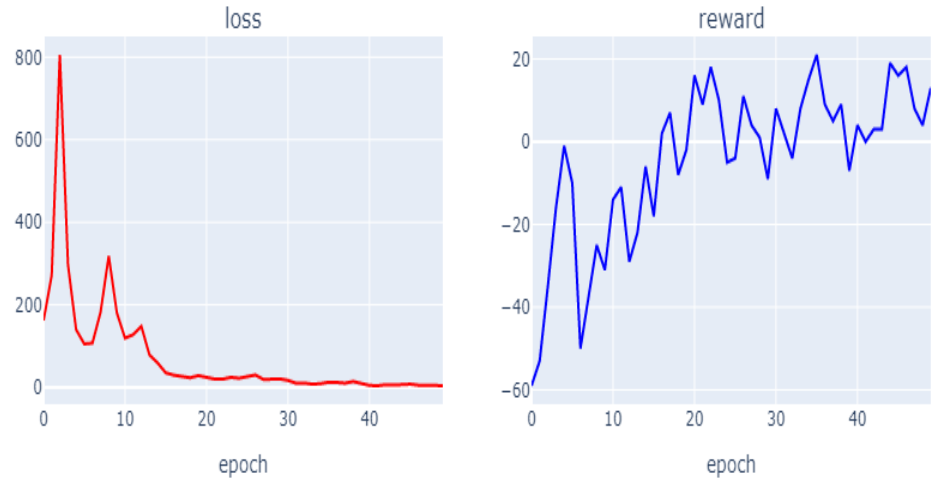

Figure 7. Learning results by substituting the Dueling DQN algorithm based on the proposed Total Resource on Google.

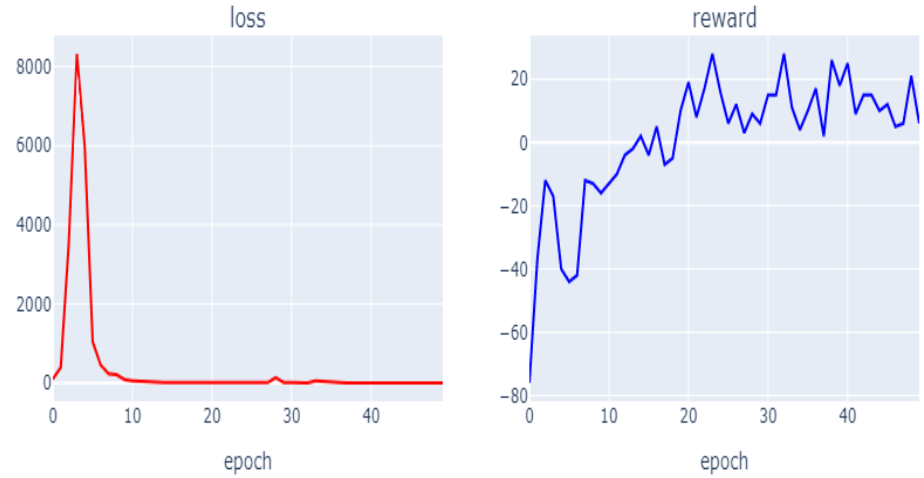

Figure 8. Learning results by substituting the Double Dueling DQN algorithm based on the proposed Total Resource on Google.

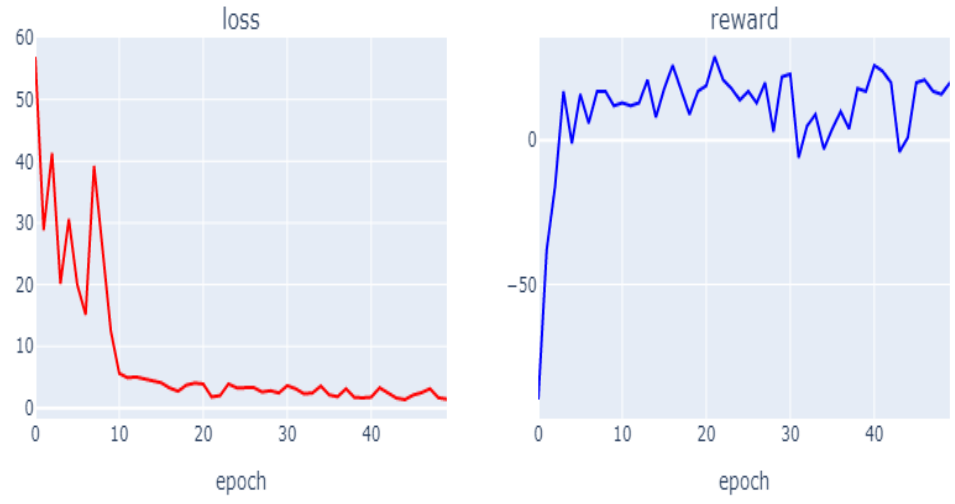

Figure 9. Learning results by substituting the Quad Q Network algorithm based on the proposed Total Resource on Google.

Figure 10 shows a comparative analysis of the proposed Q Network reward of Total Resource on Google. We can see that the reward values have stabilized quickly in the order of DQN, Dueling DQN, Double Dueling DQN, and Quad Q Network. The proposed Quad Q Network algorithm presents an enhanced performance compared to the number of epochs based on the reduction of Loss value.

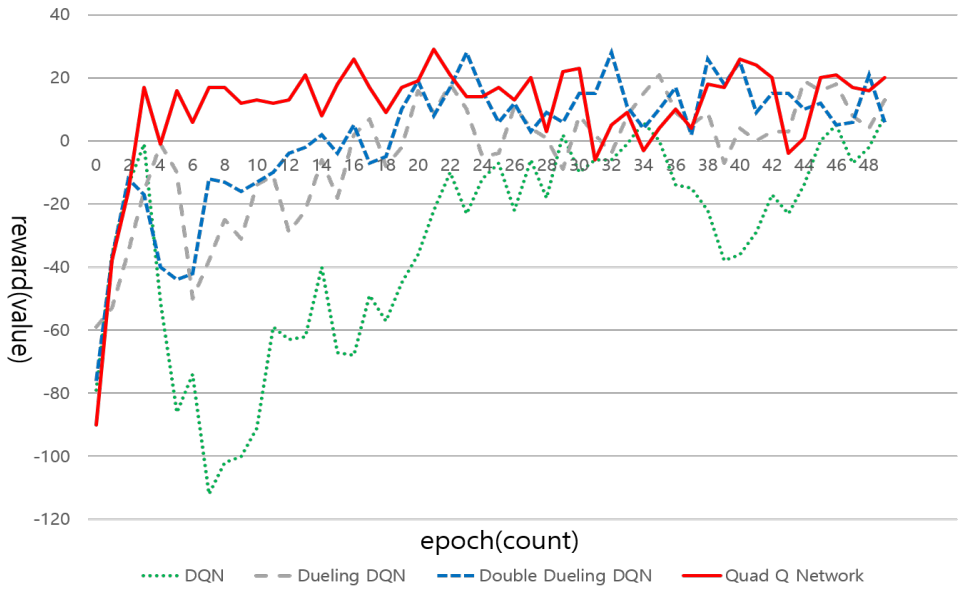

Figure 10. Comparative analysis of the proposed Q Network reward for the value of Total Resource on Google.

4.3. DQN, Dueling DQN, Double Dueling DQN, and QQN Learning Results on Apple's Total Resource

Since the model was customized for specific datasets, the datasets type was changed and Total Resource on Apple was used. The learning process was achieved by measuring the reward based on this datasets.

Figure 11 presents learning results by substituting the DQN algorithm based on the proposed Total Resource on Apple. After the learning process, the minimum reward was −57, the maximum reward was 53, the maximum loss value was approximately 4257, and the minimum loss value was 9.6.

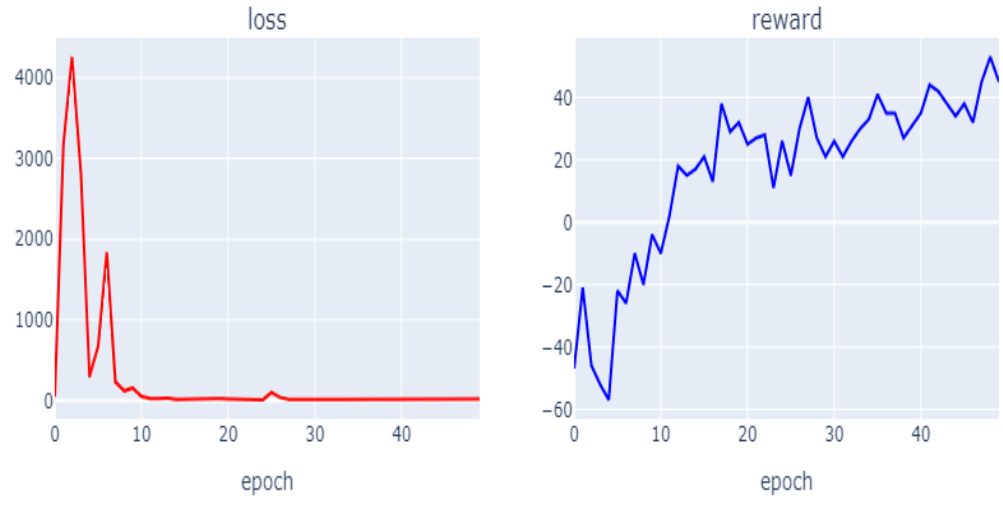

Figure 11. Learning results by substituting the DQN algorithm based on the proposed Total Resource on Apple.

Figure 12 presents learning results by substituting the Dueling DQN algorithm based on the proposed Total Resource on Apple. After the learning process, the minimum reward was −228, the maximum reward was 7, the maximum loss value was approximately 176 K, and the minimum loss value was 43.

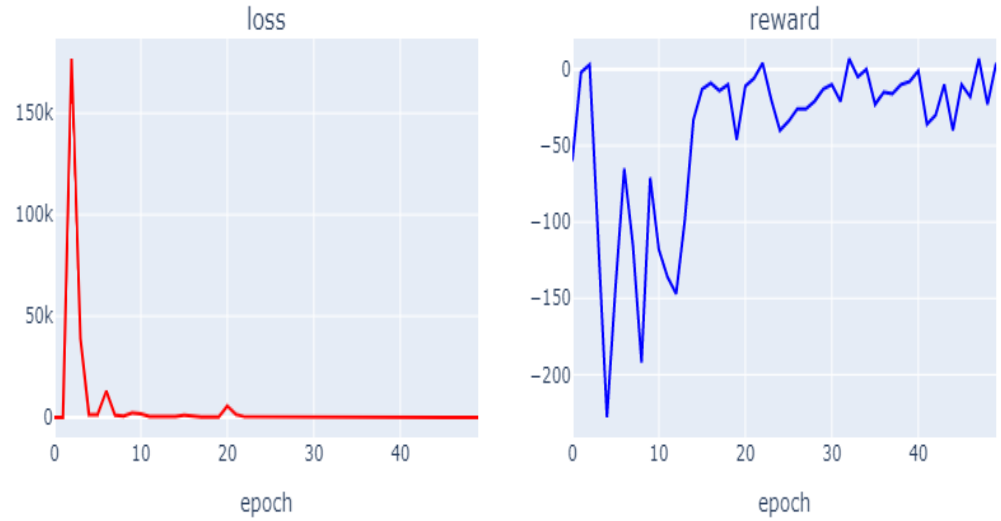

Figure 12. Learning results by substituting the Dueling DQN algorithm based on the proposed Total Resource on Apple.

Figure 13 presents learning results by substituting the Double Dueling DQN algorithm based on the proposed Total Resource on Apple. After the learning process, the minimum reward was −39, the maximum reward was 58, the maximum loss value was approximately 2814, and the minimum loss value was 7.

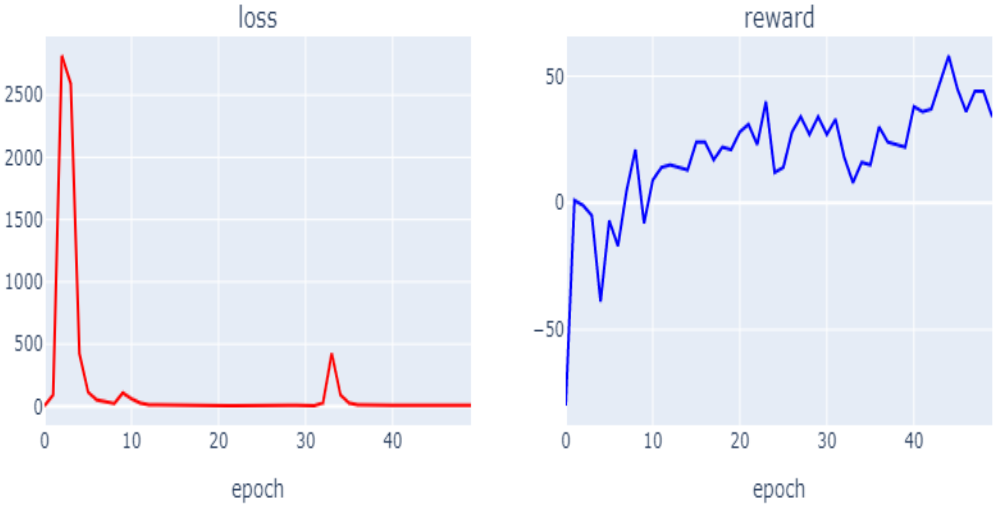

Figure 13. Learning results by substituting the Double Dueling DQN algorithm based on the proposed Total Resource on Apple.

Figure 14 presents learning results by substituting the Quad Q Network algorithm based on the proposed Total Resource on Apple. After the learning process, the minimum reward was 0, the maximum reward was 42, the maximum loss value was approximately 118, and the minimum loss value was 0.6.

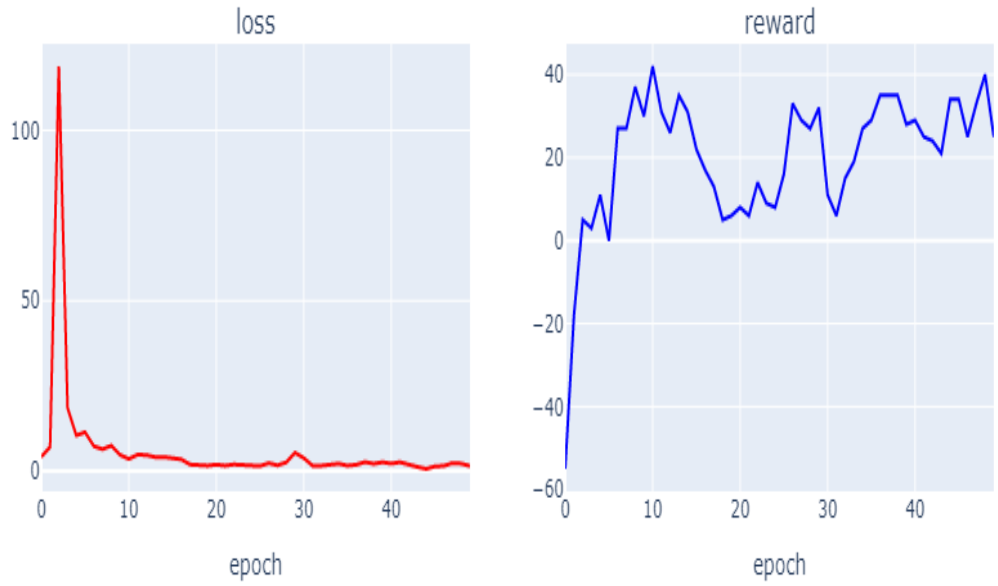

Figure 14. Learning results by substituting the Quad Q Network algorithm based on the proposed Total Resource on Apple.

Figure 15 shows comparative analysis of the proposed Q Network reward of Total Resource on Apple. We observe that reward values stabilize quickly in the order of DQN, Dueling DQN, Double Dueling DQN, and Quad Q Network. Depending on Datasets. It is absolutely difficult to judge the performance of DQN and Dueling DQN. However, in the case of the proposed Quad Q Network, experiments using previous datasets also confirmed that fast performance derivation could result from loss value reduction, further resulting in fast stabilized reward values.

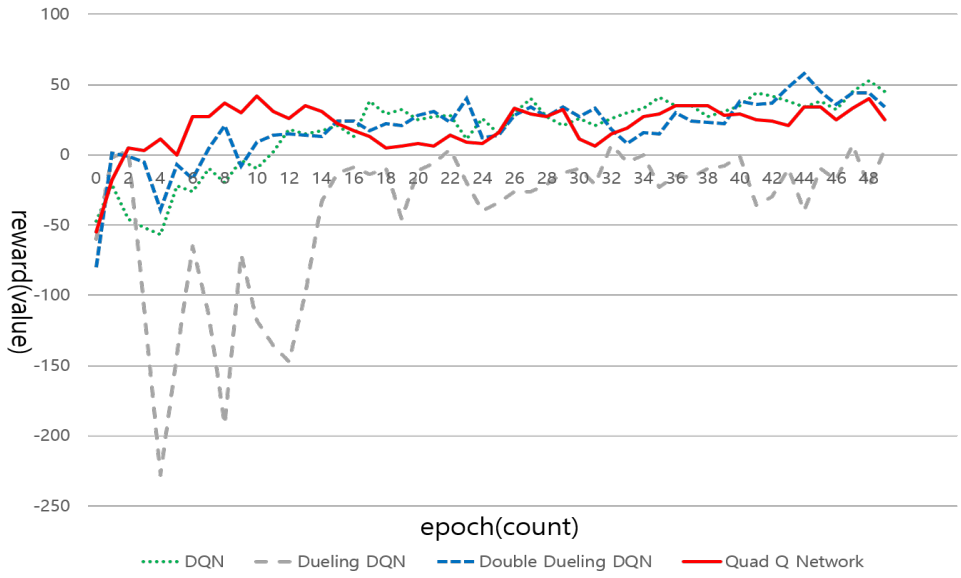

Figure 15. Comparative analysis of the proposed Q Network reward for the value of Total Resource on Apple.

5. Conclusions

Fuzzy reasoning mimics human reasoning and tests random actions that agents might take as reward functions in reinforcement learning. However, there has been no works on additional tasks particularly in loss value calibration tasks in conventional reinforcement

learning. In this paper, loss value correction with Quad Q Network has are shown to achieve superior performance. The presented simulation scenarios demonstrate that the proposed Quad Q Network algorithm could induce smooth supply and demand of enterprise resources and prevent use of surplus resources and resource missing events. This paper proposes a supply-demand response algorithm to minimise enterprise resource and resource price by shifting load supply-demand, in response to enterprise resource price signal and consumer preferences, from peak periods when the resource price is high to off-peak demand when the resource price is low. In this study, an effective household resource management tool was developed using Q-learning to deal with dynamic resource prices and different resource consumption patterns without compromising the users' lifestyle or preferences. The proposed quad Q Network performed better than other techniques through renewal of *Loss* value using *Target Q* value as a task for minimizing the loss value to predict enterprise resources. Quad Q Network algorithm was proven to have good performances of loss reduction through other datasets. These algorithms could be used for predicting resources in other fields such as predicting future electricity demand, predicting computer resources, and predicting network communication resources in the future.

Author Contributions: Conceptualization, Y.K. and J.K. (Jinsul Kim); methodology, Y.K. and J.P.; software, Y.K. and J.P.; validation, J.P. and S.L.; formal analysis, J.P.; investigation, J.Y. and S.L.; resources, J.Y. and J.K. (Jinyoung Kim); data curation, J.P. and S.L.; writing—original draft preparation, Y.K.; writing—review and editing, J.K. (Jinyoung Kim) and S.L.; visualization, J.Y.; supervision, J.K. (Jinsul Kim) and S.L.; project administration, J.K. (Jinsul Kim); funding acquisition, J.K. (Jinsul Kim). All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2020-2016-0-00314) supervised by the IITP (Institute for Information & communications Technology Planning & Evaluation) also the results of a study on the supported by the BK21 FOUR Program(Fostering Outstanding Universities for Research, 5199991714138) funded by the Ministry of Education(MOE, Korea) and National Research Foundation of Korea(NRF) and supported by the IITP grant funded by the MSIT(No. 2019-0-01343).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data sharing not applicable.

Conflicts of Interest: The authors declare that there are no conflicts of interest.

References

1. Alfaverh, F.; Denai, M.; Sun, Y. Demand Response Strategy Based on Reinforcement Learning and Fuzzy Reasoning for Home Energy Management. *IEEE Access* **2020**, *8*, 39310–39321. [CrossRef]
2. Wu, Y.; Dai, H.N.; Wang, H. Convergence of Blockchain and Edge Computing for Secure and Scalable IIoT Critical Infrastructures in Industry 4.0. *IEEE Internet Things J.* **2020**, *8*, 2300–2317. [CrossRef]
3. Vaio, A.D.; Boccia, F.; Landriani, L.; Palladino, R. Artificial Intelligence in the Agri-Food System: Rethinking Sustainable Business Models in the COVID-19 Scenario. *Sustainability* **2020**, *12*, 4851. [CrossRef]
4. Agbehadji, I.E.; Awuzie, B.O.; Ngowi, A.B.; Millham, R. Review of Big Data Analytics, Artificial Intelligence and Nature-Inspired Computing Models towards Accurate Detection of COVID-19 Pandemic Cases and Contact Tracing. *Int. J. Environ. Res. Public Health* **2020**, *17*, 5330. [CrossRef]
5. Lee, S.; Choi, D.-H. Energy management of smart home with appliances, energy storage system and electric vehicle: A hierarchical deep reinforcement learning approach. *Sensors* **2020**, *20*, 2157. [CrossRef]
6. Athanassopoulos, E.; Voskoglou, M.G. A philosophical treatise on the connection of scientific reasoning with fuzzy logic. *Mathematics* **2020**, *8*, 875. [CrossRef]
7. Chen, C.-H.; Jeng, S.-Y.; Lin, C.-J. Mobile Robot Wall-Following Control Using Fuzzy Logic Controller with Improved Differential Search and Reinforcement Learning. *Mathematics* **2020**, *8*, 1254. [CrossRef]
8. Rustum, R.; Kurichiyanil, A.M.J.; Forrest, S.; Sommariva, C.; Adeloye, A.J.; Zounemat-Kermani, M.; Scholz, M. Sustainability Ranking of Desalination Plants Using Mamdani Fuzzy Logic Inference Systems. *Sustainability* **2020**, *12*, 631. [CrossRef]
9. Gowida, A.; Elkatatny, S.; Al-Afnan, S.; Abdulraheem, A. New computational artificial intelligence models for generating synthetic formation bulk density logs while drilling. *Sustainability* **2020**, *12*, 686. [CrossRef]

10. Yang, Q.; Jang, S.; Yoo, S.J. Q-Learning-Based Fuzzy Logic for Multi-objective Routing Algorithm in Flying Ad Hoc Networks. *Wirel. Pers. Commun.* **2020**, *113*, 115–138. [CrossRef]
11. Chen, S.; Lin, T.; Jheng, K.; Wu, C. Application of Fuzzy Theory and Optimum Computing to the Obstacle Avoidance Control of Unmanned Underwater Vehicles. *Appl. Sci.* **2020**, *10*, 6105. [CrossRef]
12. Gao, H.; Ran, L.G.; Wei, G.W.; Wei, C.; Wu, J. VIKOR Method for MAGDM Based on Q-Rung Interval-Valued Orthopair Fuzzy Information and Its Application to Supplier Selection of Medical Consumption Products. *Int. J. Environ. Res. Public Health* **2020**, *17*, 525. [CrossRef]
13. Bylykbashi, K.; Qafzezi, E.; Ampririt, P.; Ikeda, M.; Matsuo, K.; Barolli, L. Performance Evaluation of an Integrated Fuzzy-Based Driving-Support System for Real-Time Risk Management in VANETs. *Sensors* **2020**, *20*, 6537. [CrossRef]
14. Ahmad, M. Alshorman, Omar Alshorman. Fuzzy-Based Fault-Tolerant Control for Omnidirectional Mobile Robot. *Machines* **2020**, *8*, 55. [CrossRef]
15. Chen, L.; Hu, X.; Tang, B.; Cheng, Y. Conditional DQN-based motion planning with fuzzy logic for autonomous driving. *IEEE Trans. Intell. Transp. Syst.* **2020**, 1–12. [CrossRef]
16. Li, Y.; Zhang, W.; Wang, C.X.; Sun, J.; Liu, Y. Deep reinforcement learning for dynamic spectrum sensing and aggregation in multi-channel wireless networks. *IEEE Trans. Cogn. Commun. Netw.* **2020**, *6*, 464–475. [CrossRef]
17. Xu, Y.; Yu, J.; Buehrer, R.M. The Application of deep reinforcement learning to distributed spectrum access in dynamic heterogeneous environments with partial observations. *IEEE Trans. Wirel. Commun.* **2020**, *19*, 4494–4506. [CrossRef]
18. Raj, V.; Dias, I.; Tholeti, T.; Kalyani, S. Spectrum access in cognitive radio using a two-stage reinforcement learning approach. *IEEE J. Sel. Top. Signal Process.* **2018**, *12*, 20–34. [CrossRef]
19. Luong, N.C.; Hoang, D.T.; Gong, S.; Niyato, D.;Wang, P.; Liang, Y.C.; Kim, D.I. Applications of deep reinforcement learning in communications and networking: A Survey. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 3133–3174. [CrossRef]
20. Wang, S.; Liu, H.; Gomes, P.H.; Krishnamachari, B. Deep reinforcement learning for dynamic multichannel access. In Proceedings of the International Conference on Computing, Networking and Communications (ICNC), Silicon Valley, CA, USA, 26–29 January 2017; pp. 599–603.
21. Schaul, T.; Quan, J.; Antonoglou, I.; Silver, D. Prioritized experienc replay. *arXiv* **2016**, arXiv:1511.05952.
22. Ye, H.; Li, G.Y. Deep reinforcement learning for resource allocation in V2V communications. In Proceedings of the 2018 IEEE International Conference on Communications (ICC), Kansas City, MO, USA, 20–24 May 2018; pp. 1–6.
23. Liu, S.; Hu, X.; Wang, W. Deep reinforcement learning based dynamic channel allocation algorithm in multibeam satellite systems. *IEEE Access* **2018**, *6*, 15733–15742. [CrossRef]
24. Shi, Z.; Xie, X.; Lu, H.; Yang, H.; Kadoch, M.; Cheriet, M. Deep reinforcement learning based spectrum resource management for industrial internet of things. *IEEE Internet Things J.* **2020**, *8*, 3476–3489. [CrossRef]
25. Zhu, J.; Song, Y.; Jiang, D.; Song, H. A new deep Q-learningbased transmission scheduling mechanism for the cognitive Internet of Things. *IEEE Internet Things J.* **2017**, *5*, 2375–2385. [CrossRef]
26. Wang, S.; Liu, H.; Gomes, P.H.; Krishnamachari, B. Deep reinforcement learning for dynamic multichannel access in wireless networks. *IEEE Trans. Cogn. Commun. Netw.* **2018**, *4*, 257–265. [CrossRef]
27. Zhong, C.; Lu, Z.; Gursoy, M.C.; Velipasalar, S. Actor-Critic deep reinforcement learning for dynamic multichannel access. In Proceedings of the 2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP), Anaheim, CA, USA, 26–29 November 2018; pp. 599–603.
28. Chang, H.; Song, H.; Yi, Y.; Zhang, J.; He, H.; Liu, L. Distributive dynamic spectrum access through deep reinforcement learning: A reservoir computing-based approach. *IEEE Internet Things J.* **2019**, *6*, 1938–1948. [CrossRef]
29. Naparstek, O.; Kobi, C. Deep multi-user reinforcement learning for dynamic spectrum access in multichannel wireless networks. In Proceedings of the Globecom 2017—2017 IEEE Global Communications Conference, Singapore, 4–8 December 2017; pp. 1–7.
30. Huang, X.L.; Li, Y.X.; Gao, Y.; Tang, X.W. Q-learning based spectrum access for multimedia transmission over cognitive radio networks. *IEEE Trans. Cogn. Commun. Netw.* **2020**, *7*, 110–119. [CrossRef]
31. Aref, M.A.; Jayaweera, S.K.; Machuzak, S. Multi-agent reinforcement learning based cognitive anti-jamming. In Proceedings of the 2017 IEEEWireless Communications and Networking Conference (WCNC), San Francisco, CA, USA, 19–22 March 2017; pp. 1–6.
32. Zhang, Y.; Cai, P.; Pan, C.; Zhang, S. Multi-agent deep reinforcement learning-based cooperative spectrum sensing with upper confidence bound exploration. *IEEE Access* **2019**, *7*, 118898–118906. [CrossRef]
33. Nabipour, M.; Nayyeri, P.; Jabani, H.; Mosavi, A.; Salwana, E.; Shahab, S. Deep Learning for Stock Market Prediction. *Entropy* **2020**, *22*, 840. [CrossRef] [PubMed]
34. Gu, J.; Fang, Y.; Sheng, Z.; Wen, P. Double Deep Q-Network with a Dual-Agent for Traffic Signal Control. *Appl. Sci.* **2020**, *10*, 1622. [CrossRef]
35. Polvara, R.; Patacchiola, M.; Hanheide, M.; Neumann, G. Sim-to-Real quadrotor landing via sequential deep Q-Networks and domain randomization. *Robotics* **2020**, *9*, 8. [CrossRef]
36. Sun, Y.; Ran, X.; Zhang, G.; Xu, H.; Wang, X. AUV 3D Path Planning Based on the Improved Hierarchical Deep Q Network. *J. Mar. Sci. Eng.* **2020**, *8*, 145. [CrossRef]
37. Li, D.; Xu, S.; Li, P. Deep reinforcement learning-empowered resource allocation for mobile edge computing in cellular v2x networks. *Sensors* **2021**, *21*, 372. [CrossRef]