Article

Decision Making for Self-Driving Vehicles in Unexpected Environments Using Efficient Reinforcement Learning Methods

Min-Seong Kim [1], Gyuho Eoh [2] and Tae-Hyoung Park [3,*]

[1] Control Robotics Department, Chungbuk National University, Cheongju 28644, Korea; kim_min_sung@naver.com
[2] Department of Mechatronics Engineering, Tech University of Korea, Siheung 15073, Korea; gyuho.eoh@tukorea.ac.kr
[3] Department of Intelligent Robotics, Chungbuk National University, Cheongju 28644, Korea
* Correspondence: taehpark@cbnu.ac.kr; Tel.: +82-043-261-3240

Abstract: Deep reinforcement learning (DRL) enables autonomous vehicles to perform complex decision making using neural networks. However, previous DRL networks only output decisions, so there is no way to determine whether the decision is proper. Reinforcement learning agents may continue to produce wrong decisions in unexpected environments not encountered during the learning process. In particular, one wrong decision can lead to an accident in autonomous driving. Therefore, it is necessary to indicate whether the action is a reasonable decision. As one such method, uncertainty can inform whether the agent's decision is appropriate for practical application where safety must be guaranteed. Therefore, this paper provides uncertainty in the decision by proposing DeepSet-Q with Gaussian mixture (DwGM-Q), which converges the existing DeepSet-Q and mixture density network (MDN). Calculating uncertainty with the Gaussian mixture model (GMM) produced from MDN made it possible to calculate faster than the existing ensemble method. Moreover, it verified how the agent responds to the unlearned situation through the Simulation of Urban MObility (SUMO) simulator and compared the uncertainty of the decision between the learned and nontrained situation.

Keywords: reinforcement learning; mixture density model; decision making

check for updates

Citation: Kim, M.-S.; Eoh, G.; Park, T.-H. Decision Making for Self-Driving Vehicles in Unexpected Environments Using Efficient Reinforcement Learning Methods. *Electronics* **2022**, *11*, 1685. https://doi.org/10.3390/electronics11111685

Academic Editors: Ahmad Taher Azar, Anis Koubaa, Alaa Khamis, Ibrahim A. Hameed and Gabriella Casalino

Received: 30 April 2022
Accepted: 24 May 2022
Published: 25 May 2022

1. Introduction

Autonomous driving on highways is divided mainly into longitudinal and lateral decisions [1]. The longitudinal decision sets the vehicle's longitudinal target speed, while the lateral decision determines whether to change the lane. Furthermore, the appropriate target speed is necessary to contribute to the driver's safety by maintaining a certain distance from the vehicle in front. The lane-changing decision is a crucial action that allows autonomous vehicles to reach their destination quickly and is, also, an action for overtaking, obstacle avoidance, cut-in/cut-out, etc. Therefore, the autonomous vehicle can be efficiently operated by safely driving and arriving at the destination in the shortest time when the longitudinal and lateral decisions are harmonized.

Decision-making problems in autonomous driving applied rule-based methods such as Intelligent Driver Model (IDM) [2] and Minimizing Overall Braking Induced by Lane Changes (MOBIL) [3]. In addition, these methods have been successfully used in the Defense Advanced Research Projects Agency (DARPA) Urban Challenge (DUC) [4–6]. However, rule-based methods have disadvantages in that they are designed for a specific environment and are difficult to extend to actual autonomous driving, where various situations can occur.

Conversely, deep reinforcement learning (DRL) has the potential for better scalability and generalization than non-learning methods for autonomous driving decision-making problems. For this reason, DRLs are widely studied in the fields of alarm systems [7], education [8], medicine [9], and sensors [10]. Huegle et al. [11,12] proposed DeepSet-Q and DeepScene-Q, which considered multiple vehicles in a region of interest (ROI) and achieved state-of-the-art quality in 2019 and 2020. However, most existing DRLs are black-box models, so the output process is unknown, and confidence in a decision is unevaluated. In particular, the agent's decision may cause an accident in an unexpected environment that the agent has never experienced during the learning process. McAllister et al. [13] mentioned that quantifying confidence is necessary to improve stability.

A common way to evaluate confidence in decisions is to quantify uncertainty, divided into epistemic and aleatoric uncertainty [14]. Epistemic uncertainty refers to the uncertainty of the model and can be reduced as it learns from additional data, while aleatoric uncertainty indicates randomness that arises from the nature of data and cannot be reduced through further learning. Both uncertainties are necessary to make risk-aware decisions, specifically epistemic uncertainty that can distinguish between learned and unlearned situations.

The problem of reaching the destination safely and quickly can be replaced by the difficulty of maintaining the target speed without accident. This paper used the Simulation of Urban MObility (SUMO) simulator to verify whether an autonomous vehicle could safely and continuously maintain its target speed in a highway environment. In addition, it was checked whether the agent could detect an inappropriate decision from the estimated uncertainty for an unexpected environment that was not encountered during the learning process. This study's contributions are as follows:

(1) This paper proposed DeepSet-Q with Gaussian mixture (DwGM-Q), which considered the unexpected environments that the DeepSet-Q agent cannot.
(2) A method different from the existing process was presented, and the estimation speed was increased compared to the uncertainty inference method using the current ensemble.
(3) The uncertainty estimation result was shown through the SUMO simulator.

2. Related Work

The classic DRL algorithm has recently shown successful results in lane-change decisions [15–17]. However, the classical DRL outputs the expected action value as a scalar, so reliability is not guaranteed. Bayesian approaches were first used to represent reliability in the field of deep learning [18]. It represents the produced value as the probability distribution rather than a scalar. The uncertainty quantification can be derived from variance in probability distributions, and dropout was used to estimate uncertainty [19]. This method is costly because it requires several samplings to obtain one uncertainty. However, the expensive uncertainty estimation through the ensemble method can be alleviated through parallelization [20]. After producing a Gaussian mixture model (GMM) using a mixture density network (MDN), a study was published quantifying the uncertainty through GMM [21].

The Bayesian approach has also been extended and used in DRL. Bayesian reinforcement learning (BRL) represents the expected action value as a probability distribution rather than a scalar. An estimated uncertainty through MC (Monte-Carlo) dropout [22] and quantified uncertainty from multiple networks through the ensemble technique have been reported in some studies [23,24]. Nevertheless, all of these methods require calculating the network of the same structure multiple times to quantify the uncertainty once.

This paper provides confidence by estimating uncertainty in the existing DeepSet-Q and calculating uncertainty as one network through a mixture density model (MDM). Quantifying uncertainty using MDN eliminated the need for sampling in the existing Bayesian method and excluded the need to calculate the same network multiple times as in the ensemble method. Consequently, it is possible to reduce the computing power required for calculation, which can be used in an embedded environment at a lower price. Research on computing offload has recently been conducted in various fields, which is an essential procedure for practical application [25,26].

3. Method

3.1. Reinforcement Learning

Reinforcement learning (RL) is a subfield of machine learning in which agents interact with some environment to learn policies that maximize expected future returns. This policy determines what action to take on an arbitrary state quantity. The environment transitions to the next state when the work is done, and the agent is rewarded. Reinforcement learning problems can be modeled as Markov decision-making processes (MDPs). Expressed as a tuple, it is equivalent to (S, A, P, R, γ), where S is the state space, A is the action space, P is the state transition probability, R is the immediate reward, and γ is the discount factor.

In Bayesian reinforcement learning, the reward can be obtained through a probabilistic basis approach. Therefore, the return $Z^{\pi} = \sum_{k=t}^{\infty} \gamma^t R_t$ is also a random variable for the fixed policy π. Q-learning aims to learn the optimal action-value function $Q^*(s_t, a_t)$, defined in Equation (1) [27].

$$Q^{\pi}(s_t, a_t) = E(Z^{\pi}(s, a) = \max_{\pi} E[(\textstyle\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)|s_t = s,\ a_t = a,\ \pi) \tag{1}$$

This value function is called the expectation of the value distribution. Through a neural network, the optimal behavioral value function is approximated as $Q(s, a;\ \theta) \approx Q^*(s,\ a)$ [28].

$$\mathrm{T}^{\pi}\mathrm{Z}(\mathrm{x},\ \mathrm{a}) = \mathrm{R}(\mathrm{x},\ \mathrm{a}) + \gamma \mathrm{P}^{\pi}\mathrm{Z}(\mathrm{X}',\ \mathrm{A}') \tag{2}$$

In Equation (2), $X' \sim P(\cdot|x, a)$, $A' \sim \pi(\cdot|X')$, the target value is estimated through three random variables. The value distribution in RL is expressed, as follows, through the distributional Bellman operator T^{π} for a fixed policy π and shown to be a contraction mapping for the maximal form of the Wasserstein metric [29].

3.2. Mixture Density Network (MDN)

MDN was first proposed in the research conducted by Bishop [30], focusing on mixture density networks and outputs π_j, μ_j, σ_j constituting the GMM. The mixture model refers to a distribution linearly combined with basic distributions, and j.jGMM pertains to several Gaussian distributions that are linearly combined.

$$p(x) = \sum_{k=1}^{K} \pi_j(x) N(x \mid \mu_j, \sigma_j) \tag{3}$$

In the above equation, π_j is the weight of the j-th Gaussian distribution, μ_j is the average of the Gaussian distribution, and σ_j is the standard deviation of the Gaussian distribution. Moreover, $\pi_j \in [0, 1]$ and $\sum_{j=1}^{K} \pi_j = 1$ are satisfied for K Gaussian distributions.

3.3. Uncertainty Estimation

The equations for estimating aleatoric uncertainty caused by inherent noise contained from GMM data, and epistemic uncertainty caused by the untrained model were proven in the study conducted by Choi et al. [21]. When K is defined as the number of Gaussian distributions, the total uncertainty for the input vector x follows the following equation:

$$\sigma_t = \sum_{j=1}^{K} \pi_j \sigma_j(s) + \sum_{j=1}^{K} \pi_j(x) \|\mu_j(x) - \sum_{k=1}^{K} \pi_k(x) \mu_k(x)\|^2 \tag{4}$$

In Equation (4), σ_t denotes the sum of the uncertainties, π_j denotes the weight of the i-th Gaussian model, and $\mu_i(x)$ denotes the average of the i-th Gaussian model.

$$\sigma_e = \sum_{k=1}^{K} \pi_k(x) \|\mu_k(x) - \sum_{j=1}^{K} \pi_j(x) \mu_j(x)\| \tag{5}$$

$$\sigma_a = \sum_{k=1}^{K} \pi_k(x)\sigma_k(x) \tag{6}$$

In Equations (5) and (6), σ_e corresponds to epistemic uncertainty, and σ_a corresponds to aleatoric uncertainty. σ_e for each decision can be calculated using Equation (5) from the variables obtained using MDN. It can be confirmed that Equations (5) and (6) are decoupled from Equation (4). Uncertainty is a value that is inversely proportional to the reliability and, as the value increases, it can be considered that the corresponding decision causes a malfunction. Equation (6) refers to uncertainty caused by probabilistic variations in a random event.

4. DeepSet with Gaussian Mixture Q Network (DwGM-Q)

4.1. Network Architecture

Figure 1 compares the existing DeepSet network structure with the DwGM-Q network. Feature points for surrounding vehicles are extracted from the ∅ network. Then, through pooling, permutation invariant features are selected. After that, the feature vector extracted from the ρ network and the state of the ego vehicle are concatenated. The Q network then shows the expected action value for each decision. It can be seen that the existing DeepSet-Q produced a value that corresponds to a scalar. In the case of an ensemble, the final Q value was obtained by summing all the output vectors of DeepSet-Q and then taking the average value. The average (μ), standard deviation (σ), and weight (π) of several Gaussian distributions were the proposed network results. In DwGM-Q with K Gaussian distribution, the Q value is defined as follows

$$Q(s_t, a_t) = \sum_{j=1}^{K} \pi_j(s_t,\ a_t)\mu_j(s_t, a_t) \tag{7}$$

The Q value occupied the probability distribution form. If the Q value comes out as a probability distribution, the uncertainty can be obtained.

4.2. Weight Update

Jensen–Shannon divergence (JSD) was used to find the difference in probability distribution between the target Gaussian mixture model and the current Gaussian mixture model [31]. To learn GMM in reinforcement learning, the distance of two Gaussian distributions Z_1 and Z_2 is explained in Equation (8).

$$\begin{aligned} JTD^2(Z_1,\ Z_2) = \ & \sum_{i,\ j} \pi_i\pi_j N\left(\pi_i; \pi_j,\ \sigma_i^2 + \sigma_j^2\right) + \sum_{i,\ j} w_i w_j N\left(m_i; m_j,\ s_i^2 + s_j^2\right) \\ & -2\sum_{i,\ j} \pi_i w_j N\left(\mu_i; m_j,\ \sigma_i^2 + s_j^2\right) \end{aligned} \tag{8}$$

JSD has excellent properties free from the disjoint support issue, which arises when using the KL-divergence loss, and exhibits unbiased sample gradients, contrary to the Wasserstein distance.

A training procedure of a DwGM-Q is shown in Algorithm 1.

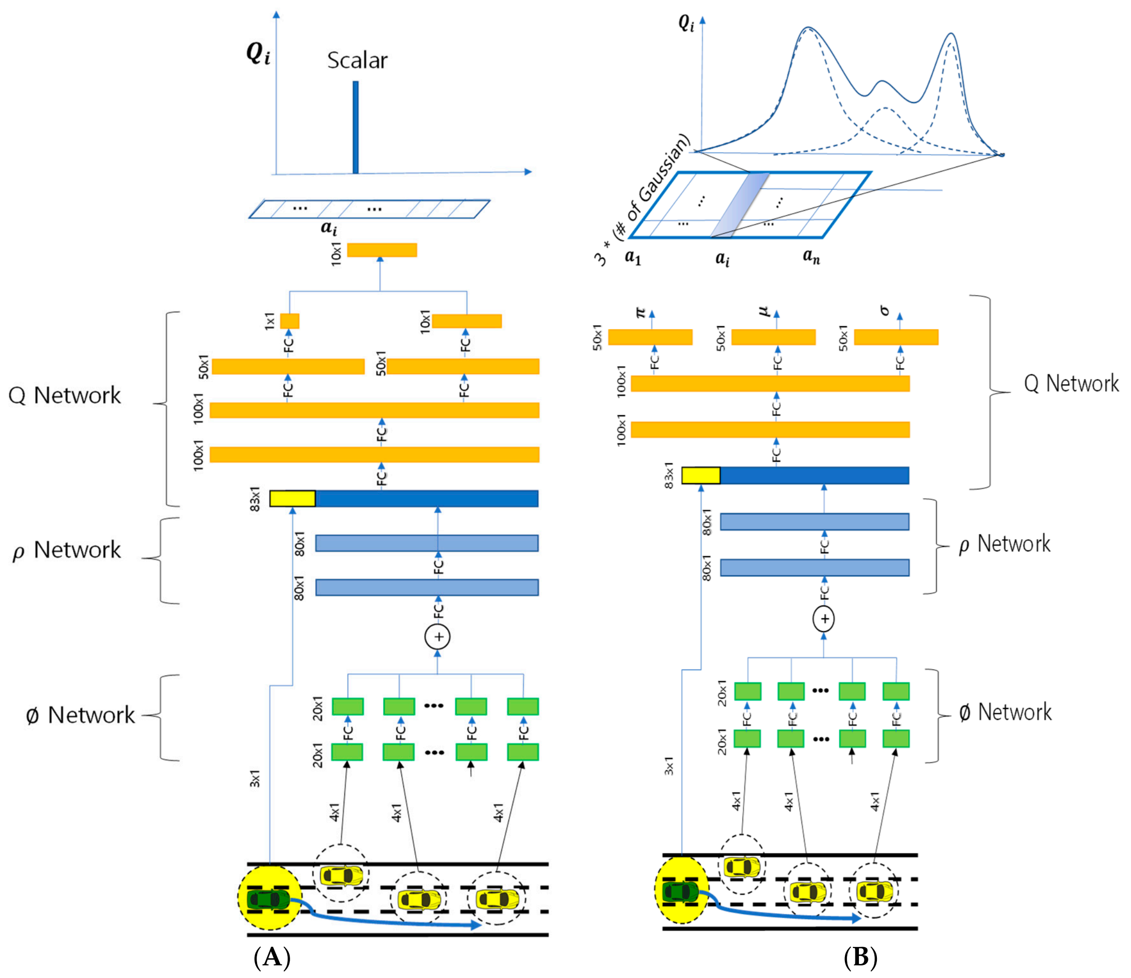

Figure 1. (**A**) DeepSet-Q and (**B**) DwGM-Q (DeepSet with Gaussian mixture Q network) architecture.

Algorithm 1: Training DwGM-Q

```
Initialize $Q = (\varnothing,\ \rho,\ Q) and\ Q^t = (\varnothing,^t\ \rho^t,\ Q^t)$
Set replay buffer $\Re$
**for** episode = 1, M **do**
initialize random state $s_i$
  **for** step = 1, T **do**
**if** $e \sim u(0,1) < \epsilon$
$\sigma_e \leftarrow \sigma(\pi,\ \mu)$ (*Equation* (5))
            *select $a_t$ as the probability*
*proportional to the value of $\sigma_e$*
**else**
     $Q(s_t, a_t) = \sum_{j=1}^{K} \pi_j(s_t,\ a_t)\mu_j(s_t, a_t)$
     select $a_t = argmax_a Q(s_t, a_t)$
     execute $a_t$ and observe reward $r_t$ and $s_{t+1}$
     Store transition $(s_t, a_t,\ r_t,\ s_{t+1})$ in $\Re$
  Sample random minibatch of transitions $(s_t, a_t,\ r_t,\ s_{t+1})$ from $\Re$
$Q(s_{t+1}, a_{t+1}) = \sum_j \pi_j(s_{t+1}, a_{t+1}),\ \mu_j(s_{t+1}, a_{t+1})$
$a^* \leftarrow \underset{a}{\text{argmax}} Q(s_{t+1}, a_{t+1})$
  $T^\pi Z(s_t, a_t) \leftarrow \sum_j \pi_j(s_t',\ a_t) N(\gamma\mu(s_{t+1}, a^*) + r,\ \ \gamma^2\sigma_j(s_{t+1},\ a^*)^2)$
  Compute JTD loss (Equation (8))
  Update weights
```

4.3. Decision Procedure

In Figure 2, c_v^{safe} is a threshold for uncertainty and determines whether to perform a_{safe}. c_v is the uncertainty estimated by DwGM-Q and calculated as follows:

$$c_v = \frac{\alpha}{M} \sum_{i=1}^{M} \sigma_t^{(i)} \tag{9}$$

where M is the number of actions, and α is the normalization factor. As the learning of DwGM-Q progresses, the GMM value approaches a single distribution. Figure 2 describes the procedure for making a decision. In input data preprocessing, the state is created with vehicle information. The details of the state are covered in Section 4. The DwGM-Q network provides the expected action value of each action in the form of GMM. From the $\pi,\ \mu,\ \sigma$ of the GMM mixture model, uncertainty can be estimated through Equations (5) and (9). The Q value is also estimated by Equation (7). If the obtained c_v is smaller than c_v^{safe}, the agent's decision is performed, assuming that the decision is reasonable. On the other hand, if it is greater than the threshold, the agent's decision is considered unreasonable, and a_{safe} is performed.

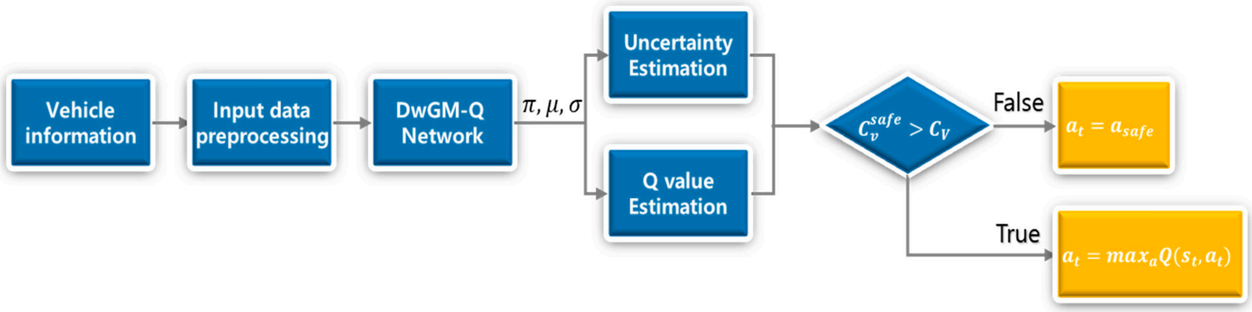

Figure 2. Flow chart of DwGM-Q decision-making process.

5. Environment Setup

Input, reward, decision type, and experimental environment were the identical neural networks stated in the conducted research by Hoel et al. [19].

5.1. State Input

As shown in Table 1, $(x_0,\ y_0)$ means the location information of the ego vehicle and $(x_i,\ y_i)$ denotes the location information of the i-th surrounding vehicle. The $v_{x,i},\ v_{y,i}$ is defined as the absolute speed of the i-th surrounding vehicle in the x and y directions. In addition, x_{sensor} refers to the longitudinal ROI that the sensor of the autonomous vehicle can detect. When the vector representing the information of the own vehicle at time t is x_t^{static} and the vector representing the information of the i-th own vehicle is x_t^i, $x_t^{static} = \{\varphi_1,\ \varphi_2,\ \varphi_3\}$, therefore, $x_t^i = \{\varphi_{4i},\ \varphi_{4i+1},\ \varphi_{4i+2},\ \varphi_{4i+3}\}$. The input vector is $s_t = \{x_t^{static}, x_t^1,\ x_t^2,\ \ldots,\ x_t^n\ \}$ if there are n number of surrounding vehicles in the ROI.

5.2. Action

The reinforcement learning agent was designed to have lane keeping, right and left lane change in the lateral direction, and acceleration of $\{-1, 0, 1\}$ m/s^2 in the longitudinal direction. Moreover, there was an emergency stop decision with -4 m/s^2 for lane keeping. This decision was expressed as a_{safe}, the most conservative decision. When the uncertainty exceeds the threshold, a_{safe} is executed. Therefore, the agent had ten decisions: three each in the horizontal and vertical directions, and one additional emergency stop decision.

Table 1. Neural network input.

Parameter	Value	Comments
φ_1	$\frac{2y_0}{y_{\max}} - 1$	Ego lane
φ_2	$\frac{2v_{x,0}}{x_{max}} - 1$	Ego vehicle speed
φ_3	$sgn\left(v_{y,\,0}\right)$	Lane-change info
φ_{4i}	$\frac{x_i - x_0}{x_{sensor}}$	Relative long. Position of i-th vehicle
φ_{4i+1}	$\frac{y_i - y_0}{y_{max}}$	Relative latitude. Position of i-th vehicle
φ_{4i+2}	$\frac{y_i - y_0}{y_{max}}$	Relative speed of i-th vehicle
φ_{4i+3}	$sgn\left(v_{y,\,i}\right)$	Lane-change state of i-th vehicle

5.3. Reward

The agent obtained a reward in the amount of $1 - \frac{v_{target} - v}{v_{target}}$ for every step. This agent induced the vehicle to overtake the slow vehicle while maintaining the target speed. Penalties of -1 and -10 were given to prevent reckless lane change if the time to collision (TTC) is less than 2.5 s with the vehicle in front. In addition, a penalty of -10 was given even when an emergency brake decision was performed.

6. Experiment

A SUMO simulator was applied for this experiment. This simulator can obtain the state quantity by inputting longitudinal acceleration and lane-change decisions. The surrounding vehicle's speed was set between 15 and 35 m/s (3.6 km/h = 1 m/s), and the targeted speed of the ego vehicle was set to 25 m/s. After making 100 decisions during one episode, the agent and surrounding vehicles were randomly initialized again. The ego vehicle corresponds to the green truck, as illustrated in Figure 3. Furthermore, the vehicle is displayed in yellow and red as it approaches 15 m/s and 35 m/s. Uncertainties were observed for three scenarios.

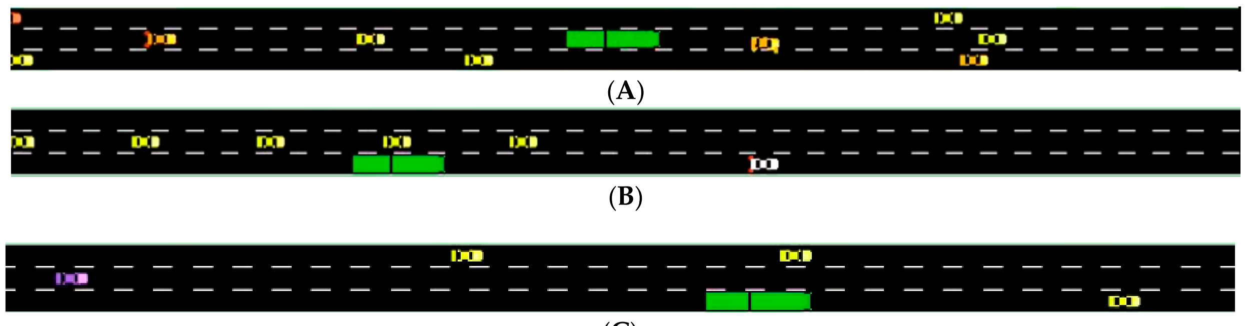

Figure 3. Three types of scenarios: (**A**) normal scenario, (**B**) stop scenario, and (**C**) speeding scenario. Green vehicle is ego vehicle, white vehicle is stationary vehicle, and purple vehicle is speeding vehicle. The other nearby vehicles are closer to red as the speed increases.

All environments were limited to three-lane highways. In scenario A (normal scenario), yellow and red represented vehicles at 15 to 35 m/s. Meanwhile, a white vehicle existed in scenario B (stop scenario), whereas a purple vehicle with a 55 m/s speed appeared in scenario C (speeding scenario). These white and purple vehicles presented in scenarios 2 and 3 correspond to surrounding vehicles missed during the learning process. This experiment presented whether the agent collides and observes the uncertainty values in unexpected situations. Parameters for training the agent are shown in Appendix A (see Tables A1 and A2). Adam optimizer was also used to optimize the reinforcement learning network.

The agent's output and uncertainty estimation results for the normal scenario are shown in Figure 4. In Figure 4B, for the situation of Figure 4C, the GMM output result for each action can be checked. The values 0–9 of Figure 4B represent ten decisions: 0, 1, 2, 3 represent 0, 1, -1, -4 m/s^2 as longitudinal direction and lane-keeping decisions as lateral direction, respectively; 4, 5, 6 represent 0, 1, -1 m/s^2, left-lane-change decision, and 7, 8, 9 represent 0, 1, -1 m/s^2, right-lane-change decision. The uncertainty values were all lower than the threshold c_v^{safe} during an episode in the normal scenario. In Figure 4B, the probability of the expected action value of the lane-change decision has many peaks. This indicates that the epistemic uncertainty increases from Equation (5). However, as the learning progressed, the value gradually decreased. In Figure 4B, the probability of the expected action value of the lane-change decision has many peaks. This indicates that the epistemic uncertainty increases from Equation (5). However, as the learning progressed, the value gradually decreased.

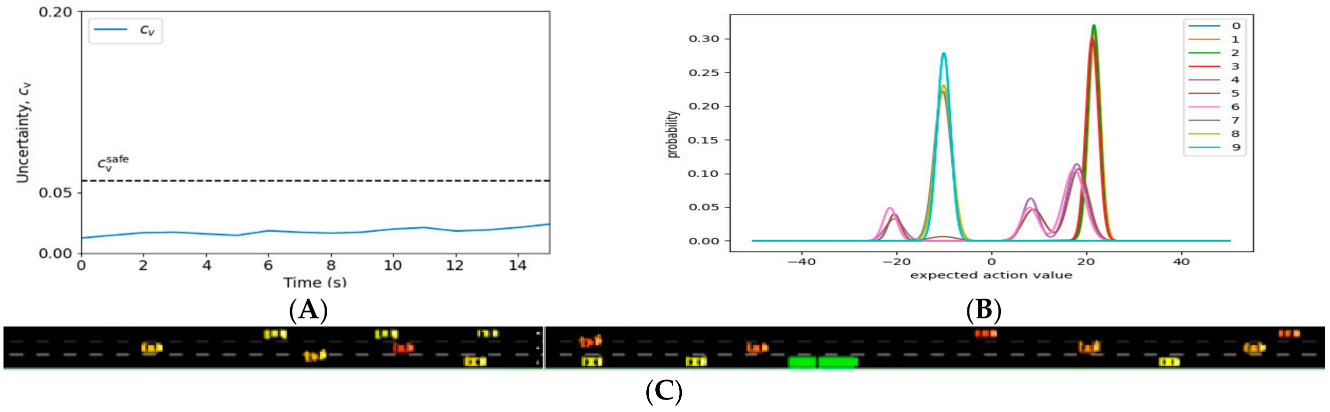

Figure 4. Uncertainty estimation result for normal scenario: (**A**) uncertainty result, (**B**) Gaussian mixture at 6 s, and (**C**) traffic condition display at 6 s.

In Figure 5A, the DwGM-Q agent observes the uncertainty value in the stopping scenario. The c_v exceeds the c_v^{safe} after the stop vehicle is detected. Figure 5B shows the output of the expected action value after the stop vehicle is detected. The GMM of most decisions has many peaks, which causes the uncertainty c_v to be increased in Figure 5A. Despite the unexpected environment that was not experienced in the learning process, the uncertainty value exceeded the threshold c_v^{safe}, causing a_{safe} to be performed, and the episode was completed without a collision.

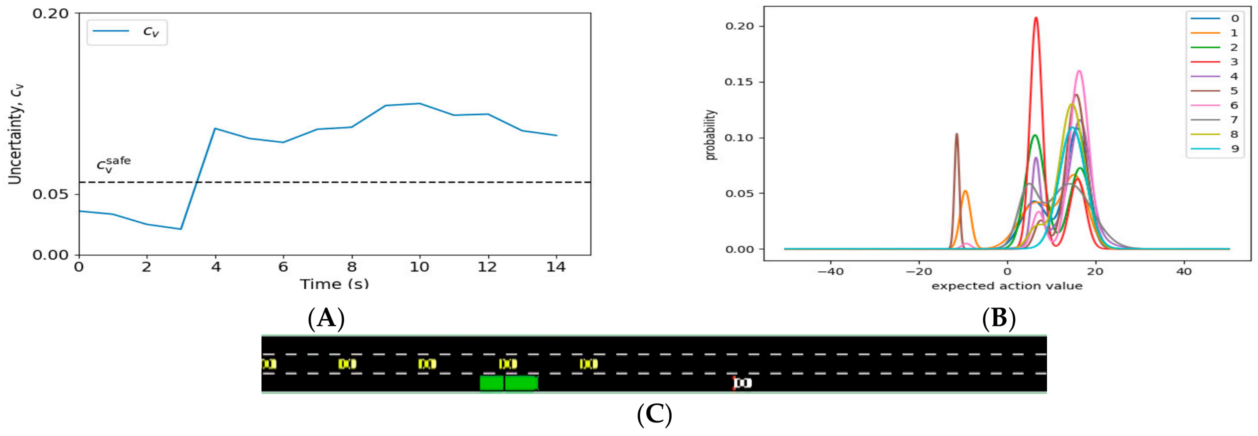

Figure 5. Uncertainty estimation result for stop scenario: (**A**) uncertainty result, (**B**) Gaussian mixture at maximum uncertainty at 4 s, and (**C**) Traffic condition display at 4 s.

Figure 6C shows the situation in which the speeding vehicle passes through the ego vehicle. When the speeding vehicle passes through the agent, the c_v exceeds c_v^{safe}, and a_{safe} is performed. Figure 6B shows the GMM values at that time. In Figure 6B, most GMMs are not single distributed, but have high deviation. This causes the c_v to be increased.

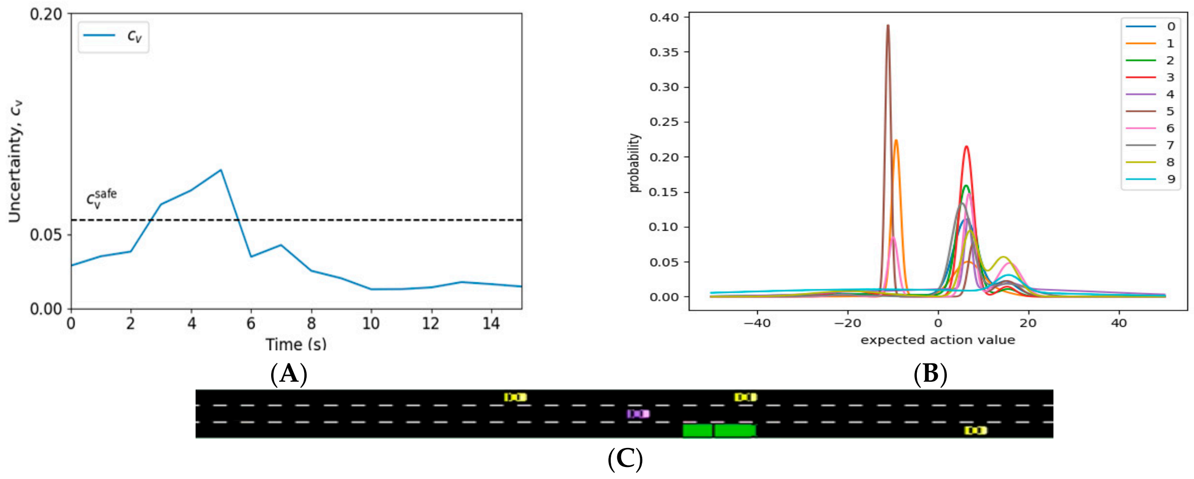

Figure 6. Uncertainty estimation result for speeding scenario: (**A**) uncertainty result, (**B**) Gaussian mixture at maximum uncertainty when c_v is highest, and (**C**) traffic condition display when c_v is highest.

Table 2 shows the performance of DeepSet-Q, RPF-Ensemble [23], and DwGM for the three types of scenarios. The value for the performance was quantified through 100 episodes.

Table 2. Experiment Results.

	Normal Scenario			**Unexpected Scenario**			
				Stop Scenario		**Speeding Scenario**	
	Mean Velocity (m/s)	**Collision Rate (%)**	**Steps Per s (SPS)**	**Collision Rate (%)**	**Steps per s (SPS)**	**Collision Rate (%)**	**Steps Per s (SPS)**
DeepSet-Q	21.9	2	213.5	100	213.6	100	213.6
DeepSet-Q (15–55 m/s)	21.8	3	213.6	100	213.5	0	213.5
RPF-Ensemble	23.4	2	165.5	0	165.3	0	165.6
DwGM-Q	22.1	2	184.4	0	184.5	0	184.5

There is no significant difference between the average speed and the collision rate for all methods in a standard scenario. The DeepSet-Q for SPS was the highest of all agents, and the proposed method was the second highest. However, the existing DeepSet-Q agent caused a crash in unexpected scenarios. On the other hand, the RPF-ensemble method or the proposed method, does not cause crashes even in unexpected scenarios. Therefore, this result shows that the proposed method can also cope with unexpected scenarios, and the computational speed is also faster than the existing method. The DwGM-Q network is faster than the ensemble method because it can obtain the uncertainty directly through a single calculation without calculating the network several times.

In addition, we have trained DeepSet-Q networks in an environment where surrounding vehicles have speeds of 15–55 m/s. This means that the DeepSet-Q agent has learned the speeding scenario. The DeepSet-Q agent performed well without colliding in the speeding scenario but collided in the stop scenario. This shows that if the DeepSet-Q agent learns an unexpected scenario, it also performs well. However, there are numerous unexpected situations, such as sensor malfunctions, various behaviors of aggressive vehicles, and complex situations that occur in real life. It is challenging to learn them all. Therefore, it is meaningful to deal with unexpected scenarios using uncertainty.

7. Conclusions

The existing DeepSet-Q algorithm failed to detect uncertainty and avoided collisions for unexpected scenarios. Existing ensemble methods avoid collisions in unexpected scenarios. However, they have the disadvantage of taking a lot of computation time due to multi-network tasks. This paper quantified the uncertainty by adding MDN to the end of the existing DeepSet-Q network to solve these problems. The experiment showed that uncertainty was well detected, and the calculation speed was faster than the existing ensemble method. The most conservative decision was selected as the break action, but it can also be replaced by something like a rule-based decision controller with higher usability.

The proposed method is proper when safety and real time are guaranteed in embedded environments such as autonomous driving. Although this paper deals with decision making in autonomous driving, it can be used in various fields, such as vehicle control and aircraft control.

Furthermore, The DwGM-Q agent could distinguish between aleatoric and epistemic uncertainty contrary to the ensemble technique. However, for DwGM-Q networks, the learning time took approximately four times longer than for DeepSet-Q. Reducing learning time with various learning techniques is also a task to be solved [32–34]. Since the uncertainty threshold is set through a heuristic, it is also necessary to establish a standard for automatically quantifying the threshold depending on the situation.

Author Contributions: Conceptualization, M.-S.K.; methodology, M.-S.K.; software, M.-S.K.; validation, M.-S.K.; formal analysis, M.-S.K.; investigation, M.-S.K.; resources, M.-S.K.; data curation, M.-S.K.; writing—original draft preparation, M.-S.K. and G.E.; writing—review and editing, M.-S.K. and G.E.; visualization, M.-S.K.; supervision, T.-H.P.; project administration, M.-S.K.; funding acquisition, T.-H.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Ministry of Science and ICT (MSIT), Korea, under the Grand Information Technology Research Center. And The APC was also funded by the Ministry of Science and ICT (MSIT), Korea, under the Grand Information Technology Research Center.

Acknowledgments: This research was supported by the Ministry of Science and ICT (MSIT), Korea, under the Grand Information Technology Research Center support program (IITP-2022-2020-0-01462) supervised by the Institute for Information and Communications Technology Planning and Evaluation (ITTP).

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the study's design, collection, analyses, the interpretation of data, in the writing of the manuscript, or in the decision to publish the results.

Appendix A

Hyperparameters for reinforcement learning are shown in Tables 1 and 2. The parameters are set from [11] and empirical experiments.

Table A1. DeepSet-Q hyperparameters.

Learning rate	1×10^4
ρ node	80
$\varnothing$ node	20
ϵ_{min}	0.1
Batch size	64
Buffer size	100,000

Table A2. DwGM-Q hyperparameters.

Learning rate	1×10^4
# of Gaussian	5
ρ node	80
$\varnothing$ node	20
ϵ_{min}	0.1
Batch size	64
Buffer size	100,000

References

1. Liu, Q.; Li, X.; Yuan, S.; Li, Z. Decision-Making Technology for Autonomous Vehicles: Learning-Based Methods, Applications and Future Outlook. In Proceedings of the 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), Indianapolis, IN, USA, 19–22 September 2021; pp. 30–37. [CrossRef]
2. Treiber, M.; Hennecke, A.; Helbing, D. Congested traffic states in empirical observations and microscopic simulations. *Phys. Rev. E* **2000**, *62*, 1805–1824. [CrossRef] [PubMed]
3. Kesting, A.; Treiber, M.; Helbing, D. General lane-changing model MOBIL for car-following models. *Transport. Res. Rec.* **2007**, *1999*, 86–94. [CrossRef]
4. Urmson, C.; Anhalt, J.; Bagnell, D.; Baker, C.; Bittner, R.; Clark, M.N.; Dolan, J.; Duggins, D.; Galatali, T.; Geyer, C.; et al. Autonomous driving in urban environments: Boss and the urban challenge. *J. Field Robot.* **2008**, *25*, 425–466. [CrossRef]
5. Montemerlo, M.; Becker, J.; Bhat, S.; Dahlkamp, H.; Dolgov, D.; Ettinger, S.; Haehnel, D.; Hilden, T.; Hoffmann, G.; Huhnke, B.; et al. Junior: The Stanford entry in the urban challenge. *J. Field Robot.* **2008**, *25*, 569–597. [CrossRef]
6. Bacha, A.; Bauman, C.; Faruque, R.; Fleming, M.; Terwelp, C.; Reinholtz, C.; Hong, D.; Wicks, A.; Alberi, T.; Anderson, D.; et al. Odin: Team VictorTango's entry in the DARPA urban challenge. *J. Field Robot.* **2008**, *25*, 467–492. [CrossRef]
7. Wang, Z.; Wan, Q.; Qin, Y.; Fan, S.; Xiao, Z. Research on intelligent algorithm for alerting vehicle impact based on multi-agent deep reinforcement learning. *J. Ambient Intell. Humaniz. Comput.* **2021**, *12*, 1337–1347. [CrossRef]
8. Madani, Y.; Ezzikouri, H.; Erritali, M.; Hssina, B. Finding optimal pedagogical content in an adaptive e-learning platform using a new recommendation approach and reinforcement learning. *J. Ambient Intell. Humaniz. Comput.* **2020**, *11*, 3921–3936. [CrossRef]
9. Tian, Z.; Si, X.; Zheng, Y.; Chen, Z.; Li, X. Multi-step medical image segmentation based on reinforcement learning. *J. Ambient Intell. Humaniz. Comput.* **2020**, *11*, 1–12. [CrossRef]
10. Cai, L.; Barnes, L.E.; Boukhechba, M. Designing adaptive passive personal mobile sensing methods using reinforcement learning framework. *J. Ambient Intell. Humaniz. Comput.* **2021**, *12*, 133. [CrossRef]
11. Huegle, M.; Kalweit, G.; Mirchevska, B.; Werling, M.; Boedecker, J. Dynamic Input for Deep Reinforcement Learning in Autonomous Driving. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), The Venetian, Macao, Macao, China, 4–8 November 2019; pp. 7566–7573. [CrossRef]
12. Huegle, M.; Kalweit, G.; Werling, M.; Boedecker, J. Dynamic Interaction-Aware Scene Understanding for Reinforcement Learning in Autonomous Driving. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; pp. 4329–4335. [CrossRef]
13. McAllister, R.; Gal, Y.; Kendall, A.; Van der Wilk, M.; Shah, A.; Cipolla, R.; Weller, A. Concrete Problems for Autonomous Vehicle Safety: Advantages of Bayesian Deep Learning. In Proceedings of the 26th International Joint Conference on Artificial Intelligence, Melbourne, Australia, 19–25 August 2017; pp. 4745–4753. [CrossRef]
14. Kendall, A.; Gal, Y. What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision? In Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS), Long Beach, CA, USA, 4–9 December 2017; pp. 5580–5590. [CrossRef]
15. Mirchevska, B.; Pek, C.; Werling, M.; Althoff, M.; Boedecker, J. High-level Decision Making for Safe and Reasonable Autonomous Lane Changing using Reinforcement Learning. In Proceedings of the 2018 21st IEEE International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018; pp. 2156–2162. [CrossRef]
16. An, H.; Jung, J.-I. Decision-making system for lane change using deep reinforcement learning in connected and automated driving. *Electronics* **2019**, *8*, 543. [CrossRef]
17. Hoel, C.; Wolff, K.; Laine, L. Automated Speed and Lane Change Decision Making using Deep Reinforcement Learning. In Proceedings of the 21st IEEE International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018; pp. 2148–2155. [CrossRef]
18. Kendall, A.; Badrinarayanan, V.; Cipolla, R. Bayesian SegNet: Model Uncertainty in Deep Convolutional Encoder-Decoder Architectures for Scene Understanding. In Proceedings of the British Machine Vision Conference (BMVC), Imperial College London, London, UK, 4–7 September 2017; pp. 57.1–57.12. [CrossRef]
19. Gal, Y.; Ghahramani, Z. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In Proceedings of the 33rd International Conference on Machine Learning (ICML), New York, NY, USA, 19–24 June 2016; pp. 1050–1059. Available online: http://proceedings.mlr.press/v48/gal16.pdf (accessed on 3 March 2022).

20. Lakshminarayanan, B.; Pritzel, A.; Blundell, C. Simple and Scalable Predictive Uncertainty Estimation Using Deep Ensembles. In Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS), Long Beach, CA, USA, 4–9 December 2017; pp. 6405–6416. [CrossRef]
21. Choi, S.; Lee, K.; Lim, S.; Oh, S. Uncertainty-Aware Learning from Demonstration Using Mixture Density Networks with Sampling-Free Variance Modeling. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane Convention & Exhibition Centre, Brisbane, Australia, 21–26 May 2018; pp. 6915–6922. [CrossRef]
22. Kahn, G.; Villaflor, A.; Pong, V.; Abbeel, P.; Levine, S.; Uncertainty-Aware Reinforcement Learning for Collision Avoidance. Berkeley Artificial Intelligence Research, University of California, Berkeley, Submitted on 3 February 2017. Available online: https://arxiv.org/pdf/1702.01182.pdf (accessed on 3 March 2022).
23. Hoel, C.-J.; Wolff, K.; Laine, L. Tactical Decision-Making in Autonomous Driving by Reinforcement Learning with Uncertainty Estimation. In Proceedings of the 2020 IEEE Intelligent Vehicles Symposium (IV), Las Vegas, NV, USA, 23–26 June 2020; pp. 1563–1569. [CrossRef]
24. Hoel, C.-J.; Tram, T.; Sjöberg, J. Reinforcement Learning with Uncertainty Estimation for Tactical Decision-Making in Intersections. In Proceedings of the 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), Rhodes, Greece, 20–23 September 2020; pp. 1–7. [CrossRef]
25. Jazayeri, F.; Shahidinejad, A.; Ghobaei-Arani, M. Autonomous computation offloading and auto-scaling the in the mobile fog computing: A deep reinforcement learning-based approach. *J. Ambient Intell. Humaniz. Comput.* **2021**, *12*, 8265–8284. [CrossRef]
26. Alam, T.; Ullah, A.; Benaida, M. Deep reinforcement learning approach for computation offloading in blockchain-enabled communications systems. *J. Ambient Intell. Humaniz. Comput.* **2022**, *13*, 313. [CrossRef]
27. Watkins, C.J.C.H.; Dayan, P. Q-learning. *Mach. Learn.* **1992**, *8*, 279–292. [CrossRef]
28. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; Riedmiller, M. Playing Atari with Deep Reinforcement Learning. 2013. Available online: https://www.cs.toronto.edu/~{}vmnih/docs/dqn.pdf (accessed on 3 March 2022).
29. Bellemare, M.G.; Dabney, W.; Munos, R. A Distributional Perspective on Reinforcement Learning. In Proceedings of the 34th International Conference on Machine Learning (ICML), Sydney, NSW, Australia, 6–11 August 2017; pp. 449–458. Available online: http://proceedings.mlr.press/v70/bellemare17a/bellemare17a.pdf (accessed on 16 March 2022).
30. Bishop, C. Mixture density networks. Neural Computing Research Group Report, Birmingham, UK. 1994. Available online: https://publications.aston.ac.uk/id/eprint/373/1/NCRG_94_004.pdf (accessed on 3 March 2022).
31. Choi, Y.; Lee, K.; Oh, S. Distributional Deep Reinforcement Learning with a Mixture of Gaussians. In Proceedings of the 2019 IEEE International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 9791–9797. [CrossRef]
32. Da Silva, F.L.; Hernandez-Leal, P.; Kartal, B.; Taylor, M.E. Uncertainty-Aware Action Advising for Deep Reinforcement Learning Agents. *Proc. Conf. AAAI Artif. Intell.* **2020**, *34*, 5792–5799. [CrossRef]
33. Wang, X.; Song, J.; Qi, P.; Peng, P.; Tang, Z.; Zhang, W.; Li, W.; Pi, X.; He, J.; Gao, C.; et al. SCC: An efficient deep reinforcement learning agent mastering the game of StarCraft II. In Proceedings of the 38th International Conference on Machine Learning, Virtual, 18-24 July 2021; pp. 10905–10915. [CrossRef]
34. Yang, D.; Qin, X.; Xu, X.; Li, C.; Wei, G. Sample Efficient Reinforcement Learning Method via High Efficient Episodic Memory. *IEEE Access* **2020**, *8*, 129274–129284. Available online: https://ieeexplore.ieee.org/document/9141230 (accessed on 3 March 2022). [CrossRef]