

Exploit-UDRM: Exploiting Uncertainty-Driven Replay Memory

1 st Given Name Surname <i>dept. name of organization (of Aff.)</i> <i>name of organization (of Aff.)</i> City, Country email address or ORCID	2 nd Given Name Surname <i>dept. name of organization (of Aff.)</i> <i>name of organization (of Aff.)</i> City, Country email address or ORCID	3 rd Given Name Surname <i>dept. name of organization (of Aff.)</i> <i>name of organization (of Aff.)</i> City, Country email address or ORCID
4 th Given Name Surname <i>dept. name of organization (of Aff.)</i> <i>name of organization (of Aff.)</i> City, Country email address or ORCID		

Abstract—Uncertainty estimation is widely used in safety-critical domains for effective and robust calibration. Reinforcement learning (RL) agents also benefit from uncertainty estimation; estimating uncertainty helps to reduce the time required for training and enables the agent to obtain greater rewards over time. Existing RL models typically use temporal difference error or the distribution of transitions in order to update the replay memory buffer to train RL controllers. In this work, we propose a novel formulation of the replay buffer, which we call the uncertainty-driven replay memory (UDRM), for updating the replay buffer based on the uncertainty estimates obtained by the agent during training. Our uncertainty estimation scheme utilizes evidential deep learning to obtain the aleatoric and epistemic uncertainty values of the agent’s predictions. Our experiments on the MinAtar games demonstrate that an uncertainty-aware replay buffer update enables the RL agent to obtain higher rewards during training, using less resources, as compared to other existing uncertainty-aware RL frameworks.

Index Terms—Markov processes, Uncertainty, “fuzzy,” and probabilistic reasoning.

I. INTRODUCTION

Predictions made by machine learning models are frequently not reliable and tend to be over-confident [1], [2]. There is ongoing research on uncertainty-aware models to improve performance and trustworthiness. Uncertainty estimation can also help improve the performance of reinforcement learning agents. Reinforcement learning agents use a replay memory buffer to learn from prior experience. However, existing research populates this replay buffer based on the TD-error or on some features of the transitions. We propose a modified approach to populate the replay buffer based on the uncertainty. In this work, we make the following key contributions:

- We propose a novel uncertainty-driven replay memory framework that is populated based on the RL agent’s prediction uncertainty. The proposed framework achieves an improved performance over existing models, by obtaining higher scores during training.
- We provide empirical evidence that our UDRM framework obtains higher scores during training and has a

higher overall average cumulative reward, when compared with other baselines.

Experiments show that the proposed framework can achieve higher scores compared to existing models for the same number of time steps.

II. RELATED WORK

A. Uncertainty Estimation

Machine learning models can confidently provide incorrect predictions for situations and data not encountered during training [3]. There is some uncertainty associated with such predictions, the quantification of which can help to improve the performance of the model. There are two main types of uncertainties: aleatoric uncertainty and epistemic uncertainty. Aleatoric uncertainty is the inherent uncertainty of the input data and cannot be reduced with additional training. Epistemic uncertainty is attributed to the black-box machine learning model and can be reduced by training the model longer. Existing research uses different methods for uncertainty estimation: ensemble learning [4], entropy, and variance [5]. Such methods require multiple networks or multiple iterations, which is an added computational and resource overhead.

a) *Evidential Deep Learning (EDL)*: EDL models provide aleatoric and epistemic uncertainty estimates in one pass and do not require the overhead of multiple networks (as in ensemble networks) [2], [6]–[10]. Evidential deep learning places an evidential prior distribution over the likelihood function, trains the neural network and obtains the hyperparameters of the evidential distribution. The evidence collected for a prediction is calculated based on the Dempster-Shafer Theory of Evidence [11]. For classification, each possible state output is assigned a mass, and the total mass should equal 1. The masses are assigned based on the evidence available to support a particular state, which is obtained from the output of the neural network. The outcome is considered to be ambiguous if all the states are given an equal mass.

In such cases, the machine learning model assigns a higher uncertainty value to that prediction, since it cannot decide on one particular output. Alternatively, if the mass is high for one of the states, then the evidence is more for that state and the machine learning model should have a lower uncertainty for the related prediction. For regression, the aleatoric and epistemic uncertainties are calculated based on the inferred hyperparameters of the evidential distribution. Several research papers have focused on improving the performance of these evidential models [12]–[14].

b) Uncertainty in Reinforcement Learning: Uncertainty estimation can also prove useful in improving the performance of reinforcement learning agents [15]–[22]. UADQN, proposed by [23], uses quantile regression to estimate aleatoric and epistemic uncertainty. CEQR-DQN [24] is an improvement on the UADQN model and uses deep evidential learning to estimate the uncertainty. The estimated uncertainty is then used to guide exploration. UBER [25] introduces an uncertainty-based filtering prior to populating the replay memory buffer. Wasserstein distance is used as a measure of uncertainty [26]. If the uncertainty is greater than a set threshold, the transition is stored in the replay buffer. This ensures that the replay buffer stores useful memories based on the uncertainty estimates. UPER [27] proposes a replay buffer prioritization based on the uncertainty. Estimated uncertainty is used to provide an information gain criterion for prioritizing transitions in the replay buffer. We propose modifications to the CEQR-DQN model to further improve the performance of the reinforcement learning agent in MinAtar games.

B. Replay Memory Buffer

The concept of experience replay and replay buffers was proposed by Lin et al. [28]. Replay buffers allow reinforcement agents to replay experiences. Current research is focused on fine-tuning the replay buffer and the sampling techniques used by the RL agent. Fedus et al. [29] demonstrate the effect of increasing replay buffer capacity on performance. Existing research also demonstrates that large replay memory buffers can negatively impact an RL agent’s performance [30], [31]. Li et al. [32] demonstrate the relationship between the TD error and the importance of a transition. Bruin et al. [33] uses characteristics such as age and TD error to determine transitions to be retained in the memory buffer. Oh et al. [34] design a permutation-equivariant neural architecture, the Neural Experience Replay Sampler (NERS), which uses features of transitions to learn contexts. This context awareness allows NERS to sample transitions that are meaningful. Remember and Forget for Experience Replay [35] distinguishes the transitions into near-policy and far-policy, based on their probability of being selected by the current policy and the behavior policy. Rahimi et al. [36] remove local neighborhood samples from the replay buffer. Eysenbach et al. [37] uses a graph search over the replay buffer to identify subgoals, improving RL agents’ performance in sparse reward environments.

a) Prioritized Experience Replay: Prioritized Experience Replay (PER) [38] uses the TD error to prioritize actions.

PER performs prioritization based on the proportions and ranks of the transitions in the replay buffer. Fujimoto et al. [39] proposes updated loss functions for PER. Horgan et al. [40] adopts the prioritized experience replay in a distributed learning environment. Zha et al. [41] proposes Experience Replay Optimization that updates the agent policy and the replay policy. The replay policy is updated on the basis of the cumulative rewards and the subsets are sampled on the basis of a priority vector. Model Augmented Prioritized Experience Replay (MaPER) proposed by Oh et al. [42] modifies the critic network to predict the next state and reward, in addition to the Q-value. Attentive Experience Replay (AER) [43] prioritizes transitions based on the frequency of visits to its state according to the current policy. Hindsight Experience Replay [44] and Competitive Experience Replay [45] focus on sparse-reward environments. While the prior handles learning by labeling visited states as goals, the latter uses two agents in competition with each other to improve learning. Luo et al. [46] proposes Dynamic Experience Replay, which uses multiple replay buffers from which the agent can sample. Brittain et al. [47] prioritizes sequences of experience to efficiently learn and improve performance.

III. METHODOLOGY

A. Uncertainty-Driven Replay Memory

This section details our proposed framework for an uncertainty-driven replay memory buffer. Existing research propose updates to the replay memory based on the TD Error or based on the transition’s position in the buffer. We propose to utilize uncertainty to update the replay memory buffer. The DQN architecture we use is adapted from [48]. As we are trying to modify the replay memory buffer, we have designed our uncertainty estimation framework to match that of CEQR-DQN [24]. UA-DQN [23] uses quantile regression and posterior networks to estimate the uncertainty of the RL agent’s predictions. Evidential deep learning provides uncertainty estimates in a single pass [6], [24]. We adopt evidential deep learning to estimate aleatoric and epistemic uncertainties of the RL agent’s predictions. The estimated uncertainty is then used to add transitions to the replay memory buffer.

An RL agent benefits from exploration and can learn from transitions with negative rewards. For this reason, we allow the agent to explore without any hindrance for the first 10% of the time steps. During the exploration stage, the aleatoric and epistemic uncertainties, for each transition, are stored to calculate the initial alpha and beta values. The threshold for uncertainty is calculated as a function of the time step

$$\text{threshold} = \alpha e^{-\beta t} \quad (1)$$

where alpha and beta are recalibrated at regular intervals and t is the time step. The threshold will become stricter in later time steps. The initial α is set as the 99th percentile of the uncertainties stored during the exploration stage. We use the 99th percentile to allow for a lenient check on the uncertainty that can eventually result in a stricter threshold. This will also

ensure that the threshold does not zero out too quickly. The initial beta value is calculated as,

$$\beta_{initial} = \frac{\log(k)}{timesteps} \quad (2)$$

where $k = 2$ denotes that the threshold should halve by the end of training. After the initial exploration has been completed, each transition is added to the replay memory buffer and each uncertainty value is compared to the threshold.

a) *Alpha and Beta recalibration:* Alpha and beta are recalibrated at fixed intervals and require prior uncertainty values to be stored. We experiment with two different recalibration intervals for $\alpha = 50$ and 500 time steps, and keep the β recalibration interval at 5000 time steps. We did not see any drastic change in the performance due to the beta recalibration interval, and hence we provide results only for one interval. We set a window of 100 to calculate the moving average of the uncertainty based on which α and β are recalibrated. If the moving average is greater than the threshold, then β is decreased by a factor of 0.95 to slow the decay process and promote exploration. Similarly, if the moving average is less than the threshold, then β is increased to speed up the decay process and promote exploitation. α is recalibrated as the 99th percentile of the stored uncertainty values. The uncertainties used for recalibration are a combination of the aleatoric and epistemic uncertainties, to ensure a wider threshold gap.

b) *Threshold comparison.:* Since aleatoric uncertainty cannot be eradicated by the model, epistemic uncertainty is compared to the threshold. If the current epistemic uncertainty is less than the threshold and the reward is greater than 0 , then the transition is added to the replay memory buffer again (to promote exploitation). To avoid rejecting transitions that are slightly below the threshold, we also re-add transitions that are not different from the threshold by more than 0.75 .

We re-add transitions to the replay buffer to bias the distribution towards transitions that the RL agent is certain about. This will require an increase in the size of the replay memory buffer. Alternative methods such as an uncertainty-weighted buffer can be explored in the future. We re-add transitions with a reward of at least 1 to ensure that only useful transitions are present twice, in the replay memory buffer. When actions are sampled from the modified buffer, the chances of sampling certain actions are higher. We used epistemic sampling at training time [15] to sample actions. We show that this modified approach allows the RL agent to obtain higher rewards in the same environment for the same number of time steps compared to other models. The algorithm 1 outlines the working of our proposed UDRM framework, as described above. Figure 1 shows the architecture of our proposed UDRM model.

B. Loss Functions

We adopt the loss functions employed in the CEQR-DQN framework [24]. We obtain the Quantile Huber Loss by combining the Quantile Regression Loss [49] and the Huber Loss [50].

Algorithm 1 Uncertainty-Aware Replay Buffer

Input: $(s,a,r,s',done)$, aleatoric_unc, epistemic_unc

Parameter: moving_avg_window,

alpha_recalibration_interval, beta_recalibration_interval

Output: Count of actions remembered twice

```

1: Let t = current_timestep
2: Let twice_remembered = 0
3: total_unc = aleatoric_unc + epistemic_unc
4: while not in initial exploration do
5:   Add  $(s,a,r,s',done)$  to replay buffer
6:   Calculate the threshold as  $\alpha e^{-\beta t}$ 
7:   Store total uncertainty and reward
8:   Calculate moving_avg_uncertainty
9:   if  $t \% \text{beta\_recalibration\_interval}$  is 0 then
10:    if moving_avg_uncertainty > threshold then
11:      Decrease the beta values
12:    else
13:      Increase the beta value
14:    end if
15:   end if
16:   if  $t \% \text{alpha\_recalibration\_interval}$  is 0 then
17:     Recalibrate alpha
18:   end if
19:   if epistemic_unc < threshold and reward > 0 then
20:     Add  $(s,a,r,s',done)$  to replay buffer
21:     Increment twice_remembered
22:   else if abs(epistemic_unc - threshold) < 0.75 then
23:     if reward > 0 then
24:       Add  $(s,a,r,s',done)$  to replay buffer
25:       Increment twice_remembered
26:     end if
27:   end if
28: end while
29: return twice_remembered

```

Conformalized Joint Prediction (CJP) can be used to jointly train point predictions and uncertainty estimations [51]. Two new components are added to the loss function, namely the interval score loss ($INTSCORE_{loss}$) and the combined calibration loss ($COMCAL_{loss}$). We adopt these loss components from Stutts et al. [24], to train the model. $INTSCORE_{loss}$ trains the model to produce outputs that are centered within a quantile. Two percentiles, 5% and 95% , are chosen as the uncertainty bounds to calculate $INTSCORE_{loss}$. The interval score loss is calculated as:

$$INTSCORE_{loss} = (Q_u - Q_l) + \frac{2}{\alpha}(Q_l - y)\mathbb{I}\{y < Q_l\} + \frac{2}{\alpha}(y - Q_u)\mathbb{I}\{y > Q_u\} \quad (3)$$

where y is the model prediction, \mathbb{I} is the indicator function, and Q_u and Q_l are the upper and lower quantiles, respectively.

$COMCAL_{loss}$ balances the distance between the upper and lower quantiles ($SHARP_{obj}$) and the distance between the

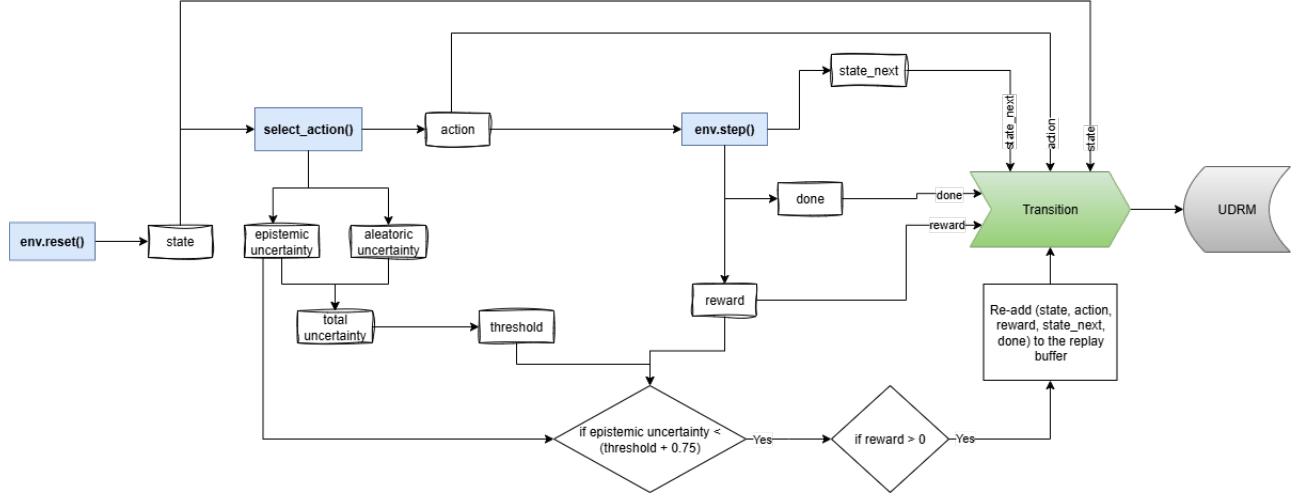


Fig. 1. Architecture of Exploitation-based UDRM

marginal coverage rate, p_m and the average probability that the prediction lies within the lower and upper quantiles, p_{avg}^{cov} (CAL_{obj}). It is calculated as follows:

$$SHARP_{obj} = \frac{1}{N} \sum_{i=1}^N \mathbb{I}\{q_i \leq 0.5\}[(1 - \hat{q}_i) - \hat{q}_i] + \mathbb{I}\{q_i > 0.5\}[\hat{q}_i - (1 - \hat{q}_i)] \quad (4)$$

$$COV_{obj} = \frac{1}{N} \sum_{i=1}^N \mathbb{I}\{p_{avg}^{cov} < q_i\}[(y_i - \hat{q}_i)\mathbb{I}\{y_i > \hat{q}_i\}] + \mathbb{I}\{p_{avg}^{cov} > q_i\}[(\hat{q}_i - y_i)\mathbb{I}\{y_i < \hat{q}_i\}] \quad (5)$$

where q is the target quantile and \hat{q}_i is the predicted quantile.

Combining the two components, \mathcal{L}_{cal} is given by:

$$\mathcal{L}_{cal} = (1 - \lambda_{cal}) \times COV_{obj} + \lambda \times SHARP_{obj} \quad (6)$$

where λ_{cal} is a hyperparameter used to balance the two components. The above combined calibration is used to calibrate both the target distribution and the mean of the evidential distribution. For the target distribution, the marginal coverage rate, p_m , is 0.5. For the evidential distribution, the marginal coverage rate is 0.9. A modified interval score function is used for the evidential distribution.

$$\mathcal{L}_{interval} = \frac{1}{N} \sum_{i=1}^N (\hat{q}_i^{95^{th}} - \hat{q}_i^{5^{th}}) + \mathbb{I}\{y < \hat{q}_i^{5^{th}}\} \frac{2}{q} (\hat{q}_i^{5^{th}} - y) + \mathbb{I}\{y > \hat{q}_i^{95^{th}}\} \frac{2}{q} (y - \hat{q}_i^{95^{th}}) \quad (7)$$

where \hat{q}_i is the predicted quantile.

The evidential quantile regression loss is given by:

$$\mathcal{L}_{evi} = \mathcal{L}_{NLL} + \lambda \mathcal{L}_{reg} \quad (8)$$

where λ is a tunable hyperparameter. \mathcal{L}_{NLL} is the negative log likelihood loss function [6] defined as:

$$\begin{aligned} \mathcal{L}_{NLL} = & \frac{1}{2} \log \left(\frac{\pi}{\nu} \right) - \alpha \log(\Omega) + \left(\alpha + \frac{1}{2} \right) \log((y - \gamma)^2 \nu + \Omega) \\ & + \log \left(\frac{\Gamma(\alpha)}{\Gamma(\alpha + \frac{1}{2})} \right) \end{aligned} \quad (9)$$

and \mathcal{L}_{Reg} is the regularization loss:

$$\mathcal{L}_{reg} = \phi[\mathbb{I}(y \geq \hat{y})q(y - \hat{y}) + \mathbb{I}(y < \hat{y})(1 - q)(\hat{y} - y)] \quad (10)$$

IV. EXPERIMENTAL RESULTS

We compare the performance of our proposed framework, UDRM, with relevant uncertainty-aware RL models, namely UADQN [23] and CEQR-DQN [24], since we have retained the RL agent's network structure from these models. Furthermore, we test our UDRM framework with another variant of the replay memory buffer, proportion-based prioritized experience replay [38], to demonstrate the effects of our modified approach. We also test our framework with two different values for the alpha recalibration intervals, $\alpha = 50$ and $\alpha = 500$. For each baseline model and environment benchmark, we run each experiment for 10 uniquely seeded trials for 2.5 million time steps on the five MinAtar games (Asterix, Breakout, Freeway, Seaquest and Space Invaders) and for 5000 time steps on the CartPole environment. Table I shows mean and standard deviation of average cumulative reward for all agents.

Our proposed framework achieves a higher maximum score compared to UADQN, CEQR-DQN and CEQR-DQN with PER (no calibration), as shown in Figure 2, in the Asterix, Breakout and Seaquest environments. For most of the games, UDRM achieves higher scores at each time step, compared to other models. For Asterix, Breakout, and Seaquest, UDRM with an alpha recalibration interval of 500 time steps (UDRM_500) performs better than the other models. For Freeway and SpaceInvaders, UDRM_500 is competitive against

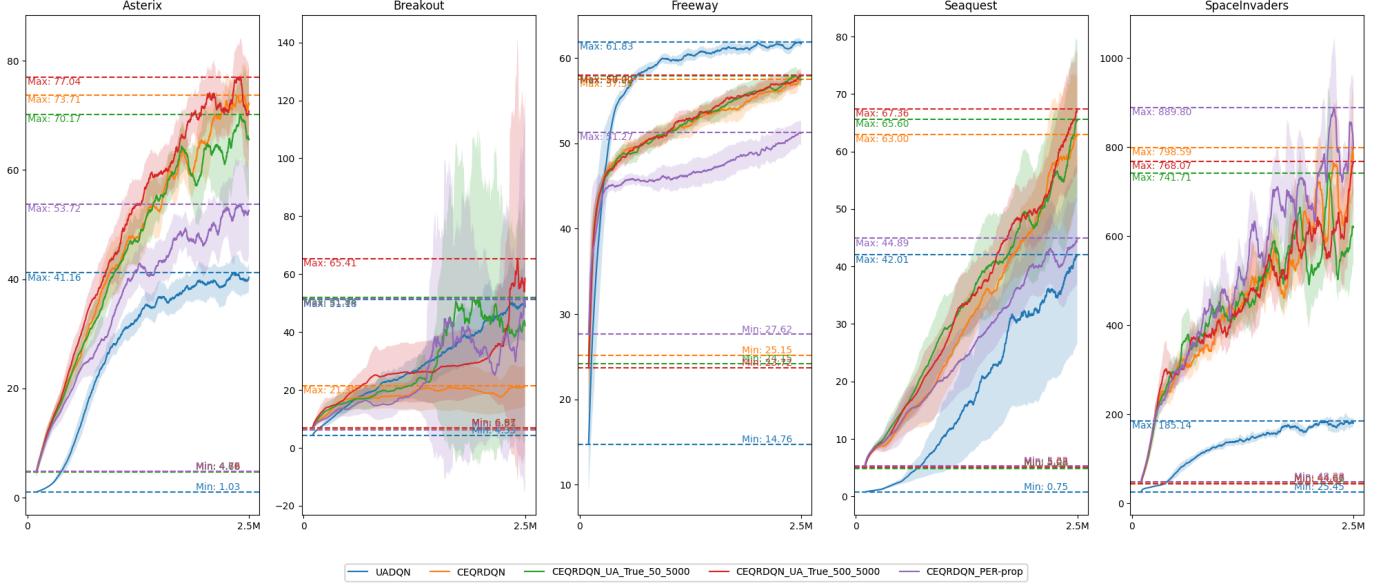


Fig. 2. Performance of UADQN, CEQR-DQN and UDRM (for 2 alpha recalibration intervals 50 and 500) on the five MinAtar games

TABLE I

AVERAGE CUMULATIVE REWARD OF OUR PROPOSED UDRM MODEL FOR CARTPOLE AND THE FIVE MINATAR GAMES OVER 10 TRIALS, COMPARED AGAINST THE UADQN, CEQR-DQN AND PER MODELS. UDRM VARIANTS OUT-PERFORM THE OTHER MODELS FOR THREE OUT OF THE SIX GAMES.

Game	UADQN	CEQR-DQN	CEQR-DQN (no calibration) with PER	UDRM ($\alpha=50$)	UDRM ($\alpha=500$)
CartPole	88.4499 ± 12.8856	39.141 ± 24.4919	43.1785 ± 7.618	50.3079 ± 7.1925	50.3584 ± 7.2659
Asterix	15.0796 ± 1.4093	34.1909 ± 3.7789	27.1338 ± 3.804	32.8226 ± 3.6796	36.0932 ± 3.294
Breakout	15.2167 ± 2.2581	13.4228 ± 4.0635	10.7632 ± 5.2481	14.5814 ± 5.5391	17.6408 ± 7.2645
Freeway	56.7454 ± 1.6765	51.3018 ± 1.8275	46.3032 ± 1.8215	51.4597 ± 2.0143	51.5763 ± 1.6764
Seaquest	8.0099 ± 4.6142	19.9233 ± 4.6946	18.3091 ± 4.7537	22.6576 ± 5.9426	22.0925 ± 4.1124
SpaceInvaders	78.4135 ± 4.4921	222.3827 ± 9.7841	239.0748 ± 17.011	220.3821 ± 17.0678	221.2928 ± 15.2414

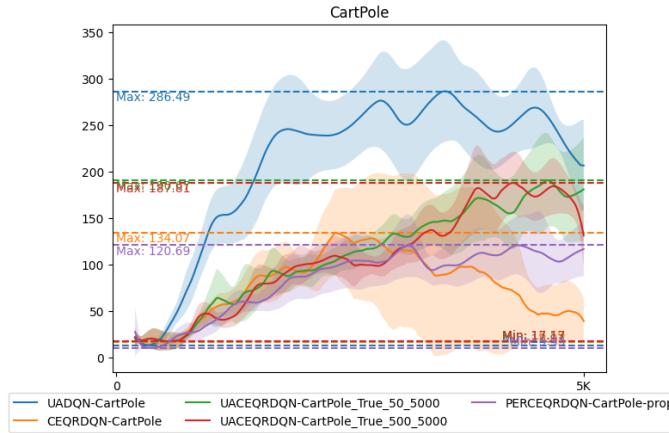


Fig. 3. Performance of UADQN, CEQR-DQN and UDRM (for 2 alpha recalibration intervals) on the CartPole environment

CEQR-DQN. UADQN performs better than the other models for CartPole (Figure 3). Although UADQN performs better for Freeway and CartPole, the resources required by UADQN are more than the resources required for UDRM. The minimum scores for UDRM are also higher than those of UADQN and

are comparable to those of CEQR-DQN. Future work can consider improving UDRM to reduce memory requirements.

V. CONCLUSION

In this work, we propose an uncertainty-driven replay memory (UDRM) buffer populated based on the uncertainties provided by the model during training. The UDRM agent stores two instances of a transition that the agent is certain about. This skews the distribution towards certain actions in the replay memory buffer. Empirical results demonstrate that our UDRM framework achieves better performance compared to other uncertainty-aware baseline models and Prioritized Experience Replay.

ACKNOWLEDGEMENTS

This material is based upon work supported by [HIDDEN]

REFERENCES

- [1] B. Lakshminarayanan, A. Pritzel, and C. Blundell, “Simple and scalable predictive uncertainty estimation using deep ensembles,” *Advances in neural information processing systems*, vol. 30, 2017.
- [2] A. Malinin and M. Gales, “Predictive uncertainty estimation via prior networks,” *arXiv*, 2018.
- [3] J. Zhang, Y. Dai, M. Xiang, D.-P. Fan, P. Moghadam, M. He, C. Walder, K. Zhang, M. Harandi, and N. Barnes, “Dense uncertainty estimation,” *arXiv*, 2021.

- [4] J. Z. Liu, J. Paisley, M.-A. Kioumourtzoglou, and B. Coull, "Accurate uncertainty estimation and decomposition in ensemble learning," *arXiv*, 2019.
- [5] S. Depeweg, J. M. Hernández-Lobato, F. Doshi-Velez, and S. Udluft, "Decomposition of uncertainty in bayesian deep learning for efficient and risk-sensitive learning," in *International conference on machine learning*. PMLR, 2018, pp. 1184–1193.
- [6] A. Amini, W. Schwarting, A. Soleimany, and D. Rus, "Deep evidential regression," *Advances in neural information processing systems*, vol. 33, pp. 14 927–14 937, 2020, amini - EDL for regression.
- [7] W. Bao, Q. Yu, and Y. Kong, "Evidential deep learning for open set action recognition," *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, vol. 00, pp. 13 329–13 338, 2021.
- [8] M. Jürgens, N. Meinert, V. Bengs, E. Hüllermeier, and W. Waegeman, "Is epistemic uncertainty faithfully represented by evidential deep learning methods?" *arXiv*, 2024.
- [9] N. Meinert, J. Gawlikowski, and A. Lavin, "The unreasonable effectiveness of deep evidential regression," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, 2023, pp. 9134–9142, meinert - improvement on Amini's EDL.
- [10] M. Sensoy, L. Kaplan, and M. Kandemir, "Evidential deep learning to quantify classification uncertainty," *arXiv*, 2018, sensoy - EDL for classification.
- [11] K. Sentz and S. Ferson, "Combination of evidence in dempster-shafer theory," 2002.
- [12] F. B. Hüttel, F. Rodrigues, and F. C. Pereira, "Deep evidential learning for bayesian quantile regression," *arXiv*, 2023, clements + this = CEQR-DQN.
- [13] D. Pandey and Q. Yu, "Learn to accumulate evidence from all training samples: Theory and practice," in *40th International Conference on Machine Learning*, 2023, pp. 26 963–26 989. [Online]. Available: <https://proceedings.mlr.press/v202/pandey23a/pandey23a.pdf>
- [14] Y. Wu, B. Shi, B. Dong, Q. Zheng, and H. Wei, "The evidence contraction issue in deep evidential regression: Discussion and solution," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 19, pp. 21 726–21 734, 2024, slides: <https://underline.io/lecture/94228-the-evidence-contraction-issue-in-deep-evidential-regression-discussion-and-solution>.
- [15] B. Charpentier, R. Senanayake, M. Kochenderfer, and S. Günnemann, "Disentangling epistemic and aleatoric uncertainty in reinforcement learning," *arXiv*, 2022, epistemic-uncertainty sampling provides better results than aleatoric-uncertainty sampling.
- [16] M.-H. Ibrahim, S. Lecoeuche, J. Boonaert, and M. Batton-Hubert, "Uncertainty quantification for efficient and risk-sensitive reinforcement learning," *2023 IEEE Symposium Series on Computational Intelligence (SSCI)*, vol. 00, pp. 1429–1434, 2023.
- [17] D. Janz, J. Hron, P. Mazur, K. Hofmann, J. M. Hernández-Lobato, and S. Tschiatschek, "Successor uncertainties: Exploration and uncertainty in temporal difference learning," *arXiv*, 2018.
- [18] O. Lockwood and M. Si, "A review of uncertainty for deep reinforcement learning," *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 18, no. 1, pp. 155–162, 2022.
- [19] V.-L. Nguyen, M. H. Shaker, and E. Hüllermeier, "How to measure uncertainty in uncertainty sampling for active learning," *Machine Learning*, vol. 111, no. 1, pp. 89–122, 2022.
- [20] J. L. Pérez, J. Corrochano, J. García, R. Majadas, C. Ibañez-Llano, S. Pérez, and F. Fernández, "Discrete uncertainty quantification for offline reinforcement learning," *Journal of Artificial Intelligence and Soft Computing Research*, vol. 13, no. 4, pp. 273–287, 2023.
- [21] H. Zhang, J. Shao, S. He, Y. Jiang, and X. Ji, "DARL: Distance-aware uncertainty estimation for offline reinforcement learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 9, pp. 11 210–11 218, 2023.
- [22] X. Zhao, S. Hu, J.-H. Cho, and F. Chen, "Uncertainty-based decision making using deep reinforcement learning," *2019 22th International Conference on Information Fusion (FUSION)*, pp. 1–8, 2019.
- [23] W. R. Clements, B. V. Delft, B.-M. Robaglia, R. B. Slaoui, and S. Toth, "Estimating risk and uncertainty in deep reinforcement learning," *arXiv*, 2019.
- [24] A. C. Stutts, D. Erricolo, T. Tulabandhula, and A. R. Trivedi, "Echoes of socratic doubt: Embracing uncertainty in calibrated evidential reinforcement learning," *arXiv*, 2024, CEQR-DQN.
- [25] A. Remonda, C. Terrell, E. Veas, and M. Masana, "Uncertainty-based experience replay for task-agnostic continual reinforcement learning," *Transactions on Machine Learning Research*, 2025. [Online]. Available: <https://openreview.net/forum?id=WxHTSPS2pi>
- [26] M. G. Bellemare, W. Dabney, and R. Munos, "A distributional perspective on reinforcement learning," *International conference on machine learning*, pp. 449–458, 2017.
- [27] R. Carrasco-Davis, S. Lee, C. Clopath, and W. Dabney, "Uncertainty prioritized experience replay," *arXiv*, 2025.
- [28] L.-J. Lin, "Self-improving reactive agents based on reinforcement learning, planning and teaching," *Machine Learning*, vol. 8, no. 3-4, pp. 293–321, 1992.
- [29] W. Fedus, P. Ramachandran, R. Agarwal, Y. Bengio, H. Larochelle, M. Rowland, and W. Dabney, "Revisiting fundamentals of experience replay," ser. International conference on machine learning, 2020, pp. 3061–3071.
- [30] H. Eren, N. Adar, and A. Yazar, "The importance of experience replay buffer size in deep reinforcement learning," 2023. [Online]. Available: <https://www.researchgate.net/publication/379190323>
- [31] S. Zhang and R. S. Sutton, "A deeper look at experience replay," *arXiv*, 2017.
- [32] A. A. Li, Z. Lu, and C. Miao, "Revisiting prioritized experience replay: A value perspective," *arXiv*, 2021.
- [33] T. D. Bruin, J. Kober, K. Tuyls, and R. Babuška, "Experience selection in deep reinforcement learning for control," *Journal of Machine Learning Research*, vol. 19, pp. 1–56, 2018.
- [34] Y. Oh, K. Lee, J. Shin, E. Yang, and S. J. Hwang, "Learning to sample with local and global contexts in experience replay buffer," *arXiv*, 2020.
- [35] G. Novati and P. Koumoutsakos, "Remember and forget for experience replay," in *36th International Conference on Machine Learning*, 2019.
- [36] A. Rahimi-Kalhroudi, J. Rajendran, I. Momennejad, H. v. Seijen, and S. Chandar, "Replay buffer with local forgetting for adapting to local environment changes in deep model-based reinforcement learning," *arXiv*, 2023.
- [37] B. Eysenbach, R. Salakhutdinov, and S. Levine, "Search on the replay buffer: Bridging planning and reinforcement learning," *arXiv*, 2019.
- [38] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *arXiv*, 2015.
- [39] S. Fujimoto, D. Meger, and D. Precup, "An equivalence between loss functions and non-uniform sampling in experience replay," *arXiv*, 2020.
- [40] D. Horgan, J. Quan, D. Budden, G. Barth-Maron, M. Hessel, H. v. Hasselt, and D. Silver, "Distributed prioritized experience replay," *arXiv*, 2018.
- [41] D. Zha, K.-H. Lai, K. Zhou, and X. Hu, "Experience replay optimization," *arXiv*, 2019.
- [42] Y. Oh, J. Shin, E. Yang, and S. J. Hwang, "Model-augmented prioritized experience replay," in *International Conference on Learning Representations*, 2022.
- [43] P. Sun, W. Zhou, and H. Li, "Attentive experience replay," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, pp. 5900–5907, 2020.
- [44] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel, and W. Zaremba, "Hindsight experience replay," *Advances in neural information processing systems*, vol. 30, 2017.
- [45] H. Liu, A. Trott, R. Socher, and C. Xiong, "Competitive experience replay," *arXiv*, 2019.
- [46] J. Luo and H. Li, "Dynamic experience replay," in *Conference on robot learning*, ser. PMLR, 2020, pp. 1191–1200.
- [47] M. Brittain, J. Bertram, X. Yang, and P. Wei, "Prioritized sequence experience replay," *arXiv*, 2019.
- [48] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015. [Online]. Available: <https://le.qun.ch/en/blog/paper-notes-human-level-control-through-deep-reinforcement-learning/>
- [49] R. Koenker and G. B. Jr, "Regression quantiles," *Econometrica: journal of the Econometric Society*, pp. 33–50, 1978. [Online]. Available: <https://www.jstor.org/stable/pdf/1913643>
- [50] P. J. Huber, "Robust estimation of a location parameter," *The Annals of Mathematical Statistics*, vol. 35, no. 1, pp. 73–101, 1964.
- [51] A. C. Stutts, D. Erricolo, T. Tulabandhula, and A. R. Trivedi, "Lightweight, uncertainty-aware conformalized visual odometry," *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 00, pp. 7742–7749, 2023.