# Safety & Exploration: A Comparative Study of Uses of Uncertainty in Reinforcement Learning

**Varun Tekur** [1] **Javin Pombra** [1] **Rose Hong** [1] **Weiwei Pan** [1]

## Abstract

For real-world deployments, it is essential to develop reinforcement learning algorithms that allow for both effective exploration and safety. One approach to this challenge that has not been as extensively studied as other methods is the use of uncertainty. In this paper, we evaluate the effectiveness of certain uncertainty-based RL models on different gridworld environments. Across all models and environments tested, we conclude that a Deep Ensemble-based approach using Randomized Prior functions (Osband et al., 2018) is ideal for solving environments with sparse rewards and rare solutions. Additionally, we observe that UADQN (Clements et al., 2020), which utilizes both aleatoric and epistemic uncertainty, can be effective in balancing safety and exploration in highly stochastic environments, but its ability to do so is highly dependent on appropriate penalization of negative events. Finally, we conclude that UADQN's ability to effectively estimate aleatoric uncertainty of Q-values is greatly impacted by its utilization of epistemic uncertainty to explore an environment's state-action space.

## 1. Introduction

In reinforcement learning (RL), being able to distinguish between two types of uncertainty – epistemic and aleatoric – can be extremely useful for applications that must strike a balance between exploration and risk-sensitivity. Epistemic uncertainty is uncertainty arising from the scarcity of data and is thus important for deciding whether more exploration may be necessary. Aleatoric uncertainty, on the other hand, is uncertainty due to stochasticity inherent in the environment and must be taken into consideration when handling risk-sensitive applications (Hüllermeier & Waegeman, 2020).

The role that uncertainty plays in safety and exploration impacts a variety of real-world use cases. Ensuring agent safety is a challenge that goes hand in hand with environments that have high aleatoric uncertainty: for instance, autonomous vehicles driving in regions with random gusts of wind, icy/slippery roads, or other high-risk situations driven by randomness. On the other hand, environments with rare solutions and sparse rewards necessitate high agent exploration, which can be achieved with the help of epistemic uncertainty. Real-world examples of these types of tasks, such as looking for resources in sparse landscapes or leaving a maze, are particularly hard problems. Indeed, even healthcare situations in which patient improvement only comes after long periods of time, such as chemotherapy, may necessitate high agent exploration.

While use-cases of estimates of epistemic and aleatoric uncertainties are numerous, there are still many barriers to disentangling and accurately estimating these uncertainties (Hüllermeier & Waegeman, 2020) as well as understanding how various uncertainty-based models compare against each other in terms of agent safety and exploration.

In this paper, we analyze the effectiveness of various RL models that use uncertainty in different ways on a handful of gridworld environments. Our main contributions are as follows: 1) an adaptation of various RL models evaluated on gridworld environments; 2) a comparison of the models via metrics assessing agent safety and performance; and 3) an analysis of the usage of epistemic and aleatoric uncertainty (if any) in these models and the effects on model behavior.

## 2. Related Work

### 2.1. Exploration

The role of exploration in RL has traditionally been framed in terms of a few key questions (Thrun, 1992):

1) *How should exploration strategies balance the speed and cost of learning?* This question targets notions of efficacy and efficiency, as intuition suggests that agents should suffi-

[1]John A. Paulson School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, USA. Correspondence to: Varun Tekur <v_tekur@college.harvard.edu>, Javin Pombra <javinpombra@college.harvard.edu>, Rose Hong <rosehong@college.harvard.edu>.

*Presented at the ICML 2021 Workshop on Uncertainty and Robustness in Deep Learning.*, Copyright 2021 by the author(s).

ciently explore their environments in order to perform well, while excessive exploration wastes both time and compute resources.

2) *How should strategies trade off exploration and exploitation (known as the exploration-exploitation tradeoff?)*

Several classic exploration strategies have been extensively studied in the literature (Akrour, 2017; McFarlane, 2018). These approaches have been shown to work well with simpler tabular RL methods as well as multi-armed bandits. In *Deterministic exploration* via greedy-policy, an agent always takes the best action known. *Stochastic exploration*, in contrast, involves sampling from a non-deterministic probability distribution in order to make decisions.

Additional classic methods are the *Upper Confidence Bounds (UCB) algorithm* (which favors actions with strong "potential" for optimality, measured by an upper confidence bound of the reward) and *Thompson sampling* (which selects an action according to the probability that it is optimal given the history). Oftentimes, however, these classic methods are less capable of handling more difficult environments such as those that rarely provide feedback and those with noise (Burda et al., 2018). Several methods to tackle exploration in these environments include *Intrinsic rewards* as exploration bonuses, in which we augment the environment reward with an additional bonus signal to encourage exploration (Bellemare et al., 2016; Schmidhuber, 1991), and *Bootstrapped exploration*, in which multiple Q-value heads are trained in parallel but each only consumes a bootstrapped sub-sampled set of data and has its own target network (Osband et al., 2016).

## 2.2. Safety

A significant number of recent approaches for achieving safety in deep RL have been *constraint-based*. In these approaches, the standard optimization procedure of an RL agent is augmented to exactly or approximately adhere to constraints over functions of costs, which are incurred when the agent experiences certain negative events (Altman, 1999). Some of these approaches add Lagrange multipliers to the loss function of the agent to meet the constraints (Ray et al., 2019), and a recent paper improves upon this by using a PID controller, which is generally used for dynamical systems, to edit the value of the multiplier in a more stable way (Stooke et al., 2020). Other methods are designed to meet constraints when using specific RL algorithms (such as Constrained Policy Optimization (Achiam et al., 2017) and Deep Constrained Q-learning (Kalweit et al., 2020)). Additionally, recent work has shown the effectiveness of pretraining networks to approximate costs using log data or agent behavior in low-stakes environments and utilizing these networks to constrain optimization in safety-critical scenarios (Srinivasan et al., 2020; Dalal et al., 2018).

We believe that constraint-based and aleatoric uncertainty-based approaches to RL safety are orthogonal. The former attempts to protect against any unsafe event, while the latter only tackles negative events that occur stochastically, and we hypothesize that utilizing aleatoric uncertainty can help better satisfy safety constraints when probabilistic catastrophes can occur (this can be experimentally explored in future work).

Finally, some previous uses of uncertainty in RL for safety are Depeweg et al. 2018, which provides a different decomposition of aleatoric and epistemic uncertainty than UADQN does (and applies it to Batch RL), and Lütjens et al. 2019, which uses epistemic uncertainty to prevent taking under-explored actions when deploying an agent.

## 3. Models

### 3.1. DQN

Of the models studied, Deep Q-Networks (DQN) are the most standard utilization of deep learning for RL. Their key idea is to utilize an artificial neural network to iteratively approximate the optimal Q function (Mnih et al., 2013). The Q function is:

$$Q^*(s,a) = \\ \max_\pi E\left[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots \mid s_t = s, a_t = a, \pi\right],$$

which is the maximum sum of rewards $r_t$ discounted by $\gamma$ at each timestep $t$, achievable by a behavior policy $\pi = P(a \mid s)$, after making an observation $(s)$ and taking an action $(a)$ (Sutton & Barto, 2018). Deep Q learning addresses instabilities and inefficiencies in RL by 1) randomly sampling data from a replay buffer of memory and 2) only periodically updating Q values towards the target. During training, an $\epsilon$-greedy policy is used to explore by sampling random (non-greedy) actions with an $\epsilon$ probability. However, DQN makes no use of uncertainty (Mnih et al., 2015).

### 3.2. UADQN

Uncertainty-aware Deep Q Networks (UADQN) (Clements et al., 2020) is an approach based on distributional RL that disentangles epistemic and aleatoric uncertainty. Quantile regression based distributional RL involves learning $N$ quantiles (each referred to here as $y_i$) of a return distribution for a state-action pair instead of only its expected value using a neural network with parameters $\theta$ (Dabney et al., 2017). UADQN treats this neural network as a Bayesian Neural Network and places a prior over $\theta$.

For epistemic uncertainty over the Q value of a state-action pair, the authors propose taking the average across quantiles of the variance of the BNN's quantile estimate:

$$\sigma^2_{epistemic} = E_{i \sim \mathcal{U}\{1,N\}} \left[ var_{\theta \sim P(\theta|D)} \left( y_i(\theta, s, a) \right) \right]$$

Additionally, UADQN defines aleatoric uncertainty as the variance of the expected value of the BNN quantile estimates with respect to the posterior distribution over $\theta$:

$$\sigma^2_{aleatoric} = var_{i \sim \mathcal{U}\{1,N\}} \left[ E_{\theta \sim P(\theta|D)} y_i(\theta, s, a) \right]$$

UADQN computes unbiased estimates of these two uncertainty measures using two networks that are approximate samples from the posterior distribution (utilizing randomized MAP sampling). Then, when taking an action at each time step, the Q value for each action is computed using the following procedure using aleatoric factor = $\lambda$, epistemic factor = $\beta$, and Q value mean $\mu_{S,A}$ (mean of all Q value quantile estimates).

$$\mu_{S,A} = \mu_{S,A} - \lambda \sigma_{aleatoric}; Q_{S,A} \sim \mathcal{N}(\mu_{S,A}, \beta^2 \sigma^2_{epistemic})$$

### 3.3. (DQNRP) Deep Q Network With Randomized Priors

Randomized Priors for Deep RL aims to solve the fundamental problem of bootstrapped models not having a mechanism for uncertainty that does not come from the observed data. This approach trains an ensemble of models, each on perturbed versions of the data, to approximate a Bayesian posterior. To then inject a prior, each member of the ensemble is initialized together with a random but fixed prior function. Predictions in each ensemble member are then taken as the sum of a trainable neural network and its prior function (Osband et al., 2018).

With this approach, ensemble networks will perform differently in areas with or without data, similar to normal Bayesian inference; that is, ensemble models will agree in rich data regions of the state space but vary in areas with no training data (indicating use of epistemic uncertainty). DQNRP's proposed benefit is to have good performance on especially difficult tasks with rare solutions and sparse rewards, such as the chain environment described in Section 4 (Osband et al., 2018).

## 4. Environments

### 4.1. Gridworld with stochastic falls

This environment was used in experiments of the UADQN paper (Clements et al., 2020) and is a $2 \times 5$ grid with starting location in the upper left corner and goal location in the upper right corner. An agent is given a -1 reward per timestep, a 10 reward for reaching the goal, and can "fall" (a negative event) off of the environment (resetting its position to the start) with probability 0.05 if it is on the top row of the environment but not on a start or goal tile. These random restarts due to "falls" are a source of aleatoric uncertainty over Q values. We aim to explore the effects of explicitly penalizing agent falls (which the UADQN paper does not do) in this environment by testing a version of the environment with a -15 additional reward for falls (as well as a version with 0 additional reward for falls).

### 4.2. Gridworld with stochastic rewards

This is a $3 \times 4$ grid with the same start and goal locations as the previous environment. An agent can also receive an additional reward with probability 0.05 if it is on the top row of the environment but not on the goal tile. By setting the stochastic additional reward to -5 (reception of this reward is called a negative event), we explore the behavior of agents in a high aleatoric uncertainty environment when the source of this uncertainty is only due to variance in single-step rewards (and not environment restarts).

### 4.3. Chain Environment

This type of environment is used in Osband et al. 2018 and features an agent on an $N \times N$ grid that samples from $Unif(0, 1)$ at each time step. Before the start of the task, a number from $Unif(0, 1)$ is also drawn for each grid location and is the number that corresponds to the action "right" for that location. The agent moves downward at each timestep, receives a $-0.01/(N-1)$ reward for a right action, 0 reward for a left action, and 1 reward for finishing in the bottom right corner. This environment requires effective exploration for success due to a rare solution action sequence and sparse large rewards. Therefore, it is useful for evaluating different approaches to using epistemic uncertainty to improve agent exploration.

## 5. Results

We evaluate the performance of all three models – DQN, UADQN, and DQNRP – on three environment types and analyze the impact of estimated uncertainty (if any) on the performance of each model. For all models, we run 3 trials of 10000 timesteps and report results as average over these trials; if a trial failed due to unstable/exploding loss, we did not include it in this average. Additionally, final rewards are computed as averages over the last 5% of episodes.

**The use of aleatoric uncertainty by UADQN allows it to explore more safely than the other models in the presence of appropriate reward penalties for negative events**. Specifically, in the stochastic fall environment with fall reward = -15, UADQN has several parameter settings that experience less than 100 falls during training and still achieve fairly successful positive rewards (Figure 2, Figure 3), while the other models are unable to match this performance: the smallest average number of falls for a successful positive reward instance is 296.7 for DQN and 172.7 DQNRP, and both of these instances also achieve significantly smaller final rewards than instances of UADQN that are both safe and have high rewards.

**However, UADQN's safety advantage can disappear if negative events are not penalized effectively**. For example, in Figure 3, it can be seen that UADQN's safety capabilities are less effective on the stochastic fall environment when fall reward = 0 than when fall reward = -15. We conclude that this occurs because assigning a -15 reward to falls allows the differences between low and high aleatoric uncertainties over Q values to be larger, letting UADQN take advantage of its explicit penalization of high aleatoric uncertainty actions to find a safer solution to the environment. Additionally, in the stochastic reward = -5 environment, we see that UADQN is actually *less* safe on average in successful scenarios than the other models are (Table 2). We hypothesize that this is because the environment has a small negative event penalty and agents are not forced to restart when a negative event occurs (and therefore can still receive a success reward after this). So, UADQN is unable to compute significantly greater aleatoric uncertainty estimates for actions that lead to states with a nonzero negative event probability than other states (Figure 4), and the use of these undesirable aleatoric uncertainty estimates during action selection likely harms agent performance. Overall, these results emphasize the importance of useful reward assignments – and not just robust models, algorithms, and uncertainty estimators – when utilizing aleatoric uncertainty estimates over Q-values to encourage agent safety.

**Additionally, UADQN's ability to effectively estimate aleatoric uncertainty for safety appears to be dependent upon its use of epistemic uncertainty for exploration**. In the stochastic fall environment with reward = -15, UADQN agents with high epistemic factors (1.0-2.0) are able to obtain more desirable uncertainty estimates due to better ability to explore the state-action space, leading to safer performance. However, these agents are only able to combine this safe performance with high reward performance if they also have low to moderate aleatoric factors (0.0-0.5). Otherwise, they appear to be unable to learn a high-reward policy because they take actions too conservatively. These performance differences can be seen by observing the heatmaps of final rewards and total falls during training (Figure 2, Figure 3). Additionally, the graph of uncertainties in Figure 8 for UADQN with $\beta = 0.0$ and $\lambda = 0.5$ is an example of a lack of exploration preventing an agent from learning useful aleatoric uncertainty estimates: the aleatoric uncertainty estimate for $(1, 0), right$ has not converged due to lack of exploration and is greater than the estimate for $(1, 1), right$. However, it is desirable for the former (which corresponds to an action that moves the agent to a tile with no fall probability) to be smaller than the latter (which corresponds to an action that moves the agent to a tile with a 0.05 fall probability). In contrast, UADQN with $\beta = 1.0$ and $\lambda = 0.2$ (which does explore) has desirable uncertainty estimates and also experiences less falls during training (Figure 6).

**Finally, the exploratory power of DQNRP appears better than the other models for solving environments with rare solution action sequences and sparse rewards**. DQNRP easily has the best average reward on the chain environment (Table 3), and it is the only model to have instances that solve the environment by consistently reaching the goal at the end of training. This indicates that despite the fact that UADQN's use of uncertainty is effective for safety, DQRNP's ensemble-based approach to using uncertainty appears to be more effective for exploration in a difficult environment.

## 6. Conclusion

There are multiple model types that have similarly high performing instances for all scenarios except two: DQNRP significantly outperforms the other models on the chain environment (it is the only model that is able to consistently solve this environment), and UADQN slightly outperforms the other models in terms of reward and significantly outperforms them in terms of safety on the stochastic fall reward = -15 environment. We conclude that the former occurs because the exploratory power of DQNRP is better than that of the other models for solving environments with very rare solution action sequences and sparse rewards. We conclude that the latter occurs because assigning a -15 reward to falls allows differences between low and high aleatoric uncertainties over Q values to be large (and therefore allows UADQN to take advantage of its explicit penalization of high aleatoric uncertainty actions to find a safe solution to the environment). However, UADQN is less safe when fall reward = 0 and on the stochastic reward = -5 environment, indicating that its safety advantages are dependent upon effective specification of rewards for unsafe events.

Finally, we determined that in stochastic fall environments, UADQN agents with high epistemic factors (1.0-2.0) can obtain ideal aleatoric uncertainty estimates because of effective exploration of the state-action space, leading to safer performance. However, these agents are only able to also achieve high rewards by using low to moderate aleatoric factors (in order to not act too conservatively).

In the future, it would be interesting to experiment with creating a model that combines DQNRP's exploratory power with UADQN's ability to avoid risky actions and evaluate this model on environments that both have high stochasticity and rare solutions/sparse rewards.

## Acknowledgements

The authors would like to thank Weiwei Pan for her guidance and support.

## References

Achiam, J., Held, D., Tamar, A., and Abbeel, P. Constrained policy optimization. In *International Conference on Machine Learning*, pp. 22–31. PMLR, 2017.

Akrour, R. Exploration in deep reinforcement learning. *Technische Universität Darmstadt*, 2017.

Altman, E. *Constrained Markov decision processes*, volume 7. CRC Press, 1999.

Bellemare, M. G., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., and Munos, R. Unifying count-based exploration and intrinsic motivation, 2016.

Burda, Y., Edwards, H., Storkey, A., and Klimov, O. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018.

Clements, W. R., Delft, B. V., Robaglia, B.-M., Slaoui, R. B., and Toth, S. Estimating risk and uncertainty in deep reinforcement learning, 2020.

Dabney, W., Rowland, M., Bellemare, M. G., and Munos, R. Distributional reinforcement learning with quantile regression, 2017.

Dalal, G., Dvijotham, K., Vecerik, M., Hester, T., Paduraru, C., and Tassa, Y. Safe exploration in continuous action spaces. *arXiv preprint arXiv:1801.08757*, 2018.

Depeweg, S., Hernandez-Lobato, J.-M., Doshi-Velez, F., and Udluft, S. Decomposition of uncertainty in bayesian deep learning for efficient and risk-sensitive learning. In *International Conference on Machine Learning*, pp. 1184–1193. PMLR, 2018.

Hüllermeier, E. and Waegeman, W. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods, 2020.

Kalweit, G., Huegle, M., Werling, M., and Boedecker, J. Deep constrained q-learning. *arXiv e-prints*, pp. arXiv–2003, 2020.

Lütjens, B., Everett, M., and How, J. P. Safe reinforcement learning with model uncertainty estimates. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 8662–8668. IEEE, 2019.

McFarlane, R. A survey of exploration strategies in reinforcement learning. *McGill University*, 2018.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. Playing atari with deep reinforcement learning, 2013.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. Human-level control through deep reinforcement learning. *Nature*, 518 (7540):529–533, February 2015. ISSN 00280836. URL http://dx.doi.org/10.1038/nature14236.

Osband, I., Blundell, C., Pritzel, A., and Roy, B. V. Deep exploration via bootstrapped dqn, 2016.

Osband, I., Aslanides, J., and Cassirer, A. Randomized prior functions for deep reinforcement learning, 2018.

Ray, A., Achiam, J., and Amodei, D. Benchmarking safe exploration in deep reinforcement learning. *arXiv preprint arXiv:1910.01708*, 2019.

Schmidhuber, J. A possibility for implementing curiosity and boredom in model-building neural controllers, 1991.

Srinivasan, K., Eysenbach, B., Ha, S., Tan, J., and Finn, C. Learning to be safe: Deep rl with a safety critic. *arXiv preprint arXiv:2010.14603*, 2020.

Stooke, A., Achiam, J., and Abbeel, P. Responsive safety in reinforcement learning by pid lagrangian methods. In *International Conference on Machine Learning*, pp. 9133–9143. PMLR, 2020.

Sutton, R. S. and Barto, A. G. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018. URL http://incompleteideas.net/book/the-book-2nd.html.

Thrun, S. B. The role of exploration in learning control. 1992.

## A. Model-Specific Results

### A.1. DQN

We use an $\epsilon$-greedy exploration policy with a starting exploration rate of 1 and final exploration rate of 0.02. The hyperparameter we search over is the final exploration step (1000, 2500, 5000, and 10000): the exploration rate decays from its initial to final value at a constant rate until this step. In general, the final exploration step does not appear to have a significant impact on the agent's performance (with the exception of when it is deployed on the gridworld with stochastic reward of -5, as it has a reward for final exploration step = 10000 of $-1.169045$ and around 6.3 for all other values of this parameter).

For the gridworlds with stochastic rewards, the DQN achieves similarly good asymptotic average rewards compared to all UADQN agents and Randomized Prior agents with prior scale less than 10. Additionally, the DQN agent with the best performance on the stochastic falls environment with no fall penalty is 4.58 (final exploration step = 2500), which not much worse than the best performing UADQN (5.14 reward, $\lambda = 1.0$, $\beta = 0.0$) and Randomized Prior agents (5.17 reward, 20 ensembles, prior scale of 3.0) on this environment.

However, DQN does not perform similarly to the best agents for the stochastic fall reward = -15 environment (its best performing agent achieves a reward of 2.73). We hypothesize that this is because multiple factors contribute to high aleatoric uncertainty along the upper edge of this environment (both a negative reward as well as a reset of the environment occur stochastically), making it difficult for a non-uncertainty aware model to achieve optimal performance in this environment.

DQN is also unable to solve the chain environment, as it never achieves a reward close to 0.99 (which is optimal) when deployed on this environment, likely because it is unable to utilize epistemic uncertainty to efficiently explore an environment with a very rare optimal sequence of actions.

In terms of safety, DQN's lowest number of falls/injuries for the stochastic fall reward = 0 and the stochastic reward = -5 environments are comparable to the best performing models. However, it has more falls for the stochastic fall reward = -15 environment on average across all parameter settings (347.4) than both UADQN (146.44) and Randomized Prior agents (184.5), again demonstrating poorer performance on this particular environment.

### A.2. UADQN

We searched over values $0, 0.2, 0.5, 1.0, 2.0$ for both $\beta$ and $\lambda$. On all environments except the Chain environment, the best possible parameter setting of a UADQN agent is comparable to the best agents across all models 1. However, UADQN stands out from the pack from a safety standpoint in the stochastic fall reward = -15 environment. We hypothesize that this is because the additional aleatoric variance caused by adding an additional penalty to random falls off of the environment allowed UADQN to penalize actions that led to the stochastic regions of the environment even more than in the other environments, causing safer behavior during training. This phenomenon is explored in more detail in subsection B.1.

| | **Stochastic Fall, 0** | **Stochastic Fall, -15** | **Stochastic Reward, -5** |
|---|---|---|---|
| **DQN** | 4.58/403.00 | 2.73/339.00 | 6.33/293.00 |
| **UADQN** | 5.14/351.67 | 4.00/106.33 | 6.59/315.33 |
| **DQNRP** | 5.17/394.33 | 3.51/231.33 | 6.54/258.33 |

*Table 1.* Best final rewards and corresponding negative event counts over environments and models

### A.3. DQNRP

We search over two primary hyperparameters for randomized priors: prior scale and number ensembles. Figure 1 illustrates the impacts: a larger number of ensembles contributes to better performance for chain environments whereas a prior scale of 1 tends to have maximum benefits. For the latter, DQNRP in general performs better for tougher environments where a larger number of different ensembles encourages exploration. For the prior scale, too small values of prior prematurely and sub-optimally converge while too large priors become too difficult to wash away.

Ultimately, DQNRP performs much better in the chain environment than all other models do. In other environments, a lower prior scale and number of ensembles is generally better (and achieves comparable performance to DQN and UADQN, especially when prior scale is $\leq 3.0$). This means that when stochastic risk in the environment is large, too many ensembles may push the agent into high aleatoric regions often, causing the accumulation of negative results.

## B. Environment-Specific Results

### B.1. Stochastic Fall Environment

The main question that we wanted to test with this environment was if explicitly penalizing stochastic falls can perform the same function as penalizing actions that lead to these falls based on aleatoric uncertainty, which would mean that estimating aleatoric uncertainty may not be necessary in order to optimize safety. Our results show that this is not the case, and in fact explicitly penalizing these falls allows aleatoric uncertainty to be even *more* useful in keeping the agent safe.

In figures 2 and 3, we see that in both environments, the highest rewards occur when aleatoric factor is between 0 and

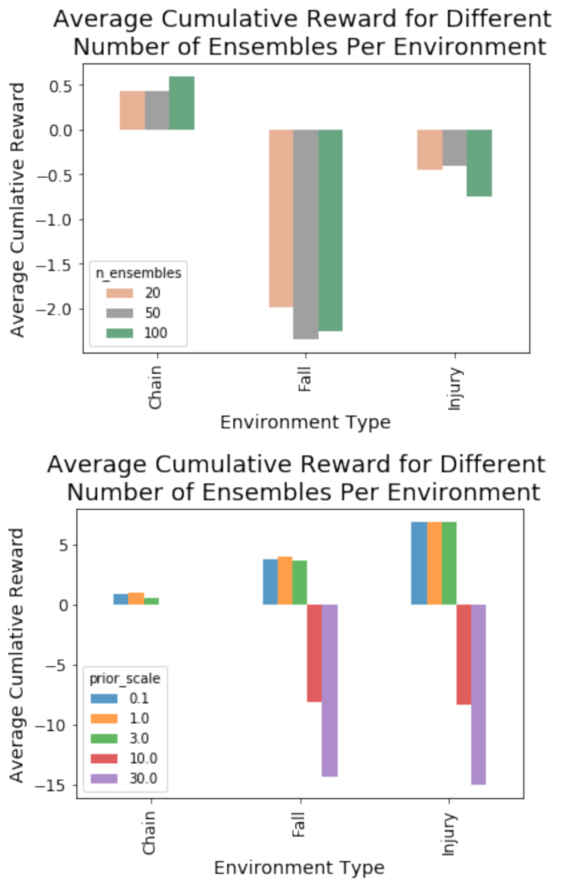


*Figure 1.* **(Top:)** Final reward for various numbers of ensembles per environment **(Bottom:)** Final reward for various prior scales per environment.

0.5. In this region of high rewards, for both environments, UADQN is the safest (lowest number of total falls) for higher epistemic factors (1.0-2.0). This may seem counter-intuitive at first because exploration can often be in tension with safety. However, we hypothesize that this trend occurs because significant exploration is needed in order to explore the high aleatoric uncertainty region of the environment enough to estimate this uncertainty well. Additionally, the total fall counts in this high reward, low fall count parameter region are much lower for fall reward = -15 environment than fall reward = 0 environment. We hypothesize that this is because the negative fall reward adds an additional source of variance to high aleatoric uncertainty regions, allowing aleatoric uncertainty estimates to be larger and thus have a larger impact on decreasing the effective Q-values of these regions. While one may argue that this same effect could be achieved by simply increasing the aleatoric factor and keeping fall reward at 0, the data shows that this does not have the same effect as setting fall reward to = -15. For the fall reward = 0 environment, almost all models with aleatoric factor $\geq 1.0$ have both negative final reward (poor

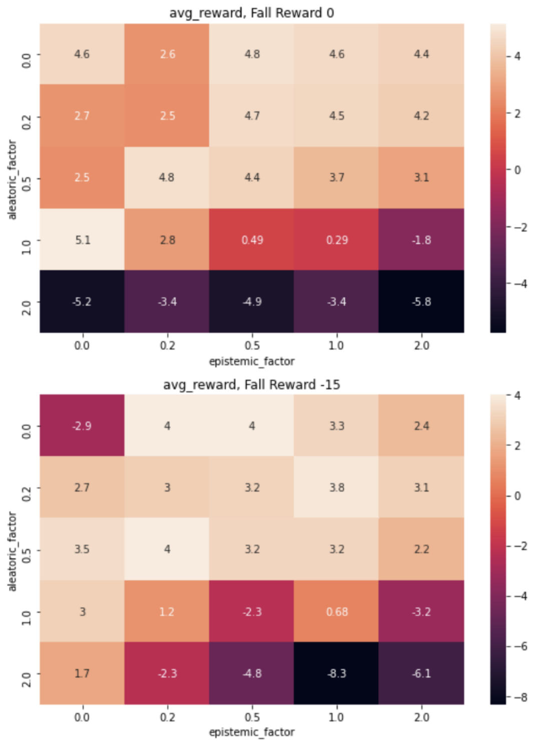


*Figure 2.* Final Reward for various UADQN parameter settings on stochastic fall environments

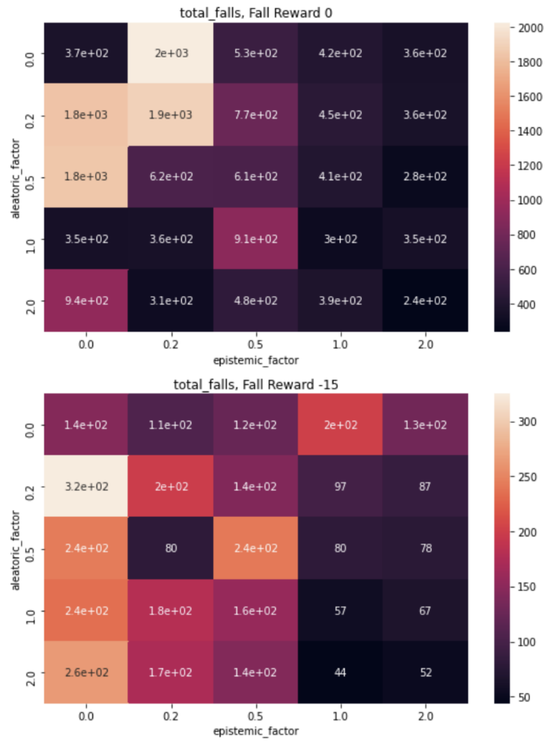


*Figure 3.* Total Number of Falls for various UADQN parameter settings on stochastic fall environments

performance) and still have a relatively large number of falls.

Overall, these environments both demonstrated the effectiveness of UADQN's aleatoric uncertainty use as well as the fact that reward engineering/selection, not just agent models, algorithms, and uncertainty estimators, can be extremely useful in ensuring the safety of a UADQN agent.

### B.2. Stochastic Reward Environment

| | Final Reward | Total Negative Events |
|---|---|---|
| **DQN** | 6.278646 | 272.888889 |
| **UADQN** | 6.399632 | 321.984127 |
| **DQNRP** | 6.464193 | 277.333333 |

*Table 2.* Final Rewards and Total Negative Events for Stochastic Reward = -5 environment averaged across trials with final reward at least 5

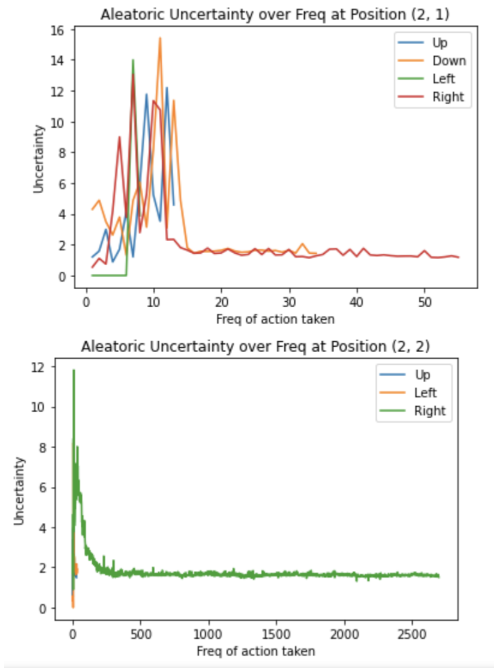


*Figure 4.* Environment is Stochastic Reward with Reward = -5 **(Top:)** UADQN Aleatoric Uncertainty vs. Action Frequency for state 2,1 with $\beta = 2.0$ and $\lambda = 1.0$ **(Bottom:)** UADQN Aleatoric Uncertainty vs. Action Frequency for state 2,2 with $\beta = 2.0$ and $\lambda = 1.0$

Interestingly, for the stochastic reward = -5 environment, UADQN is actually less safe on average than the other models for successful instances. We note that this is a 3x4 environment with the top row (y-coordinate 2) as the row with stochastic rewards. Therefore, the path directly through this dangerous row achieves a total reward of 10-3 = 7 if no stochastic negative rewards are obtained, and the shortest path that avoids this region achieves a total reward of 10-5 = 5. If we define a reward of 5 or greater as success on this environment, the average number of falls for successful UADQN agents is greater than that for successful agents of the other models (Table 2). If we further inspect aleatoric uncertainties vs frequency for a UADQN model in this environment at a state along the dangerous path, 2,2, and a state along the shortest non dangerous path, 2,1, we can obtain insights into the negative performance of UADQN on this environment ((**?**)). Namely, the aleatoric uncertainties for the action $right$ for 2,1 and 2,2 are very similar, which is undesirable because this action sends the agent to a state with no stochastic reward for the former and a state with a stochastic reward for the latter. Additionally, at state 2,1, the action $down$ sends the agent to a stochastic reward tile while the action $right$ sends the agent to a tile along the shortest non-dangerous path. However, the aleatoric uncertainty for $down$ appears to be similar to (and sometimes less than) that for the action $right$, indicating that noisy aleatoric uncertainty estimates may even encourage dangerous behavior. Overall, we hypothesize that a less explicit Q-value difference between states with and without stochastic rewards (both due to the small -5 stochastic reward instead of -15 as well as a lack of falls, which ensure that an agent can still reach the goal and achieve a success reward even if it incurs a stochastic -5 reward) causes UADQN to be unable to learn useful aleatoric uncertainty estimates. These estimates do not allow avoidance of unsafe behavior and even appear to encourage it because they are noisy.

### B.3. Chain Environment

| | Final Reward | Variance of Reward |
|---|---|---|
| **DQN** | 0.218650 | 0.022146 |
| **UADQN** | 0.044379 | 0.004779 |
| **DQNRP** | 0.489235 | 0.050075 |

*Table 3.* Final Reward and Variance of Reward at end of training for Chain Environment (Averaged Across Trials)

As seen in table 3, ultimately the Randomized Prior Ensemble approach performs significantly better than all other agents in the chain environment as measured by final reward. While DQNRP has the highest final reward on average, it also has the highest final reward variance on average. This behavior is in line with what we would expect from the benefit of DQNRP: deep exploration. The existence of multiple ensembles and moreover multiple prior functions means that agents (or specifically at least one model in the ensemble) will explore data-low regions without incentive. Indeed, this in line with our hyperparameter analysis that a higher number of ensembles was only better for DQNRP in the chain environment.

Interestingly, DQN does much better than the uncertainty-based approaches of UADQN. The intuition can be seen in hyperparameter trends for the chain environment. For UADQN, the "chain" environment is the only environment where a lower (or 0) epistemic scale is the best while a higher aleatoric scale leads to higher rewards. This is the opposite of what we would expect for the chain environment where more exploration needs to be encouraged. Thus, UADQN is able to avoid major risks, but in cases that are akin to "needles in a haystack," the safety mechanism seems to overpower the epistemic-based exploration.

## C. General Guiding Questions and Analysis

### C.1. Guiding Questions

1. **Do aleatoric and epistemic uncertainty estimates of the models align with what we would expect?** For example, does epistemic uncertainty of $(S, A)$ pairs decrease as they are visited more often? Does aleatoric uncertainty remain small for infrequently visited $(S, A)$ pairs and stabilize to higher values for pairs that lead to stochastic rewards than pairs that do not?

2. **How do agent *safety* and *performance* vary as aleatoric and epistemic factors of models vary?** Are any settings of these parameters for a model particularly good at optimizing both of these properties? Are there any tradeoffs that exist between these properties for the tested environments?

### C.2. Quality of Aleatoric and Epistemic Uncertainty Estimates

We see several informative differences of uncertainty estimates for $(S, A)$ pairs across model types and parameters, and we highlight several examples of these differences here (all examples are for the fall reward = -15 stochastic fall environment)

Epistemic uncertainty estimates for $(S, A)$ pairs for UADQN qualitatively tend to make sense, as they converge to zero as the number of actions increases, as seen in the example in Figure 5.

Additionally, the quality of UADQN aleatoric uncertainty estimates appears to be impacted by the epistemic factor of the model. In the next section, we explain how this effect of the epistemic factor on aleatoric estimates impacts model performance.

In Figure 6 and Figure 7, we see that aleatoric uncertainty estimates for the action of going right from states 1,0 and 1,1 converge to a particular value as that $(S, A)$ pair is visited more often, which is desirable because aleatoric uncertainty should be learnable from data (in this case, $Q(s, a)$ values) once we collect a certain amount of this data. Additionally, the value converged to for the pair $(1, 0), right$ is lower than the value converged to for the pair $(1, 1), right$, which is also desirable because $(1, 1), right$ moves onto a tile with fall probability of 0.05 and $(1, 0), right$ moves to a tile with fall probability of 0.

However, in Figure 8, we see that while the aleatoric uncertainty estimate for $Q((1, 1), right)$ converges to a value, this estimate for $Q((1, 0), right)$ does not converge to a value. This is particularly undesirable because $(1, 0), right)$ is an $(S, A)$ pair on the optimal path from the start to the goal that does not cross any tiles with positive fall probability, and therefore the agent likely avoids this $(S, A)$ pair despite the fact that it is favorable from both a performance and safety standpoint.

We then note that Figure 6 and Figure 7 correspond to UADQN agents with parameters $\beta = 1.0$ and $\lambda = 0.2$ and $\beta = 2.0$ and $\lambda = 2.0$, final average rewards of 3.81 and -6.08, and average total fall counts of 51.67 and 97.33, respectively (which are both relatively low). We conclude that since both of these agents had high epistemic factors, they were able to explore enough to learn desirable aleatoric uncertainties for relevant actions and therefore use these uncertainties to somewhat avoid traveling over tiles with a positive fall probability (however, this safety didn't translate into reward-based success for both agents, and this phenomenon is explained in more detail in the next section). In contrast, Figure 8 corresponds to a UADQN agent with parameters $\beta = 0.0$ and $\lambda = 0.5$, and we conclude that the lack of exploration for this agent did not allow it to explore the $(S, A)$ pair $(1, 0), right$ enough, leading to an undesirable aleatoric uncertainty estimate for this action that is along a safe and reasonably performant path. This likely contributed to the agent's lack of safety despite good final reward (244.00 average total falls, 3.50 final average reward).

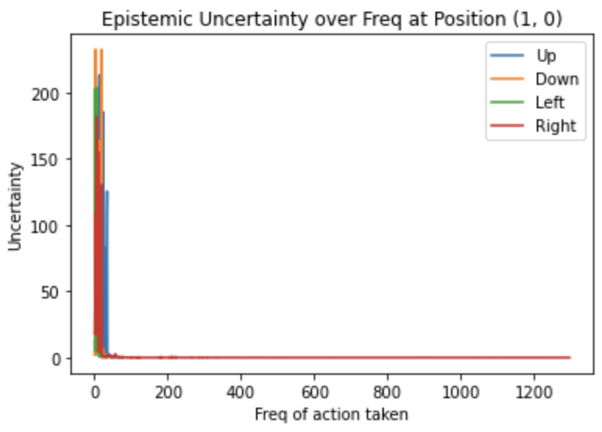


*Figure 5.* Environment is Stochastic Fall with Reward = -15. UADQN Epistemic Uncertainty vs. Action Frequency for state 1,0 with $\beta = 1.0$ and $\lambda = 0.2$

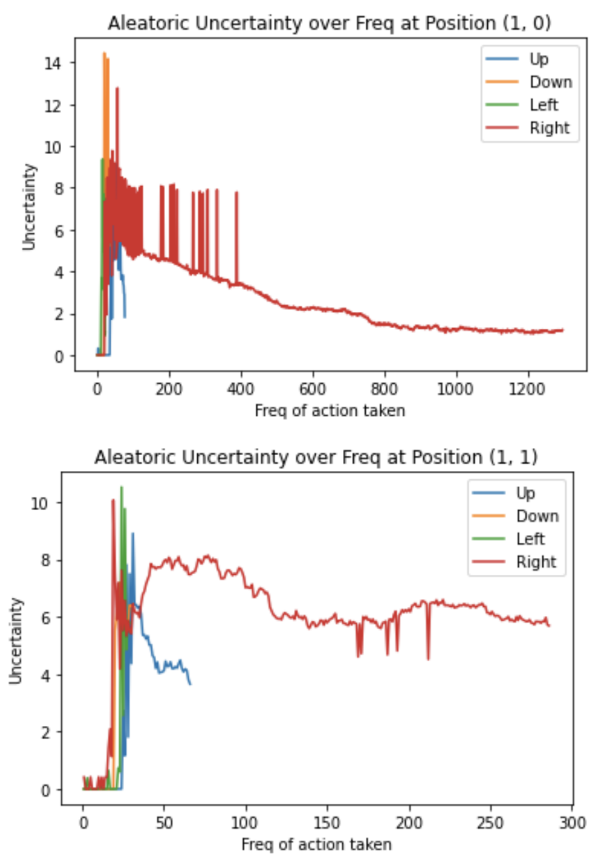


*Figure 6.* Environment is Stochastic Fall with Reward = -15 **(Top:)** UADQN Aleatoric Uncertainty vs. Action Frequency for state 1,0 with $\beta = 1.0$ and $\lambda = 0.2$ **(Bottom:)** UADQN Aleatoric Uncertainty vs. Action Frequency for state 1,1 with $\beta = 1.0$ and $\lambda = 0.2$

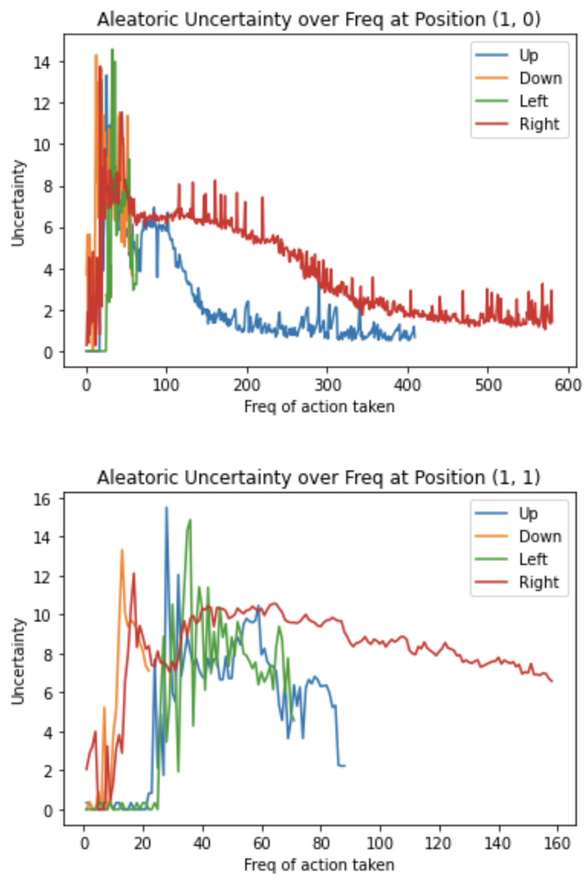
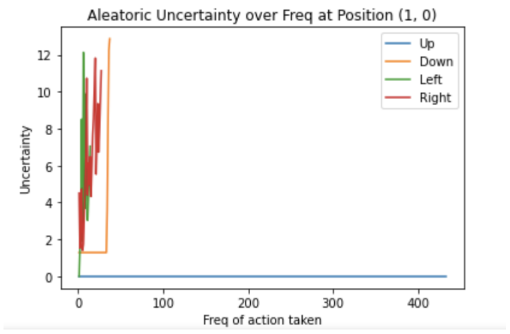


*Figure 7.* Environment is Stochastic Fall with Reward = -15 **(Top:)** UADQN Aleatoric Uncertainty vs. Action Frequency for state 1,0 with $\beta = 2.0$ and $\lambda = 2.0$ **(Bottom:)** UADQN Aleatoric Uncertainty vs. Action Frequency for state 1,1 with $\beta = 2.0$ and $\lambda = 2.0$

## C.3. Impact of Aleatoric and Epistemic Factors

We also analyze the impact of the weights that are placed on aleatoric and epistemic uncertainty in UADQN. For these weights, we see certain trends emerge in both instances of the high-aleatoric uncertainty stochastic fall environments.

As seen in Figure 2 and Figure 3, as epistemic factor ($\beta$) increases and aleatoric factor ($\lambda$) stays fixed, when aleatoric factor is small (0-0.5), we see reward increase and then decrease/stay the same. In this case, we also see the total number of falls decrease. However, as epistemic factor increases and aleatoric factor stays the same when aleatoric factor is large (1.0-2.0), we see the reward decrease and the total number of falls stay roughly the same. We hypothesize that this behavior generally occurs because increasing epistemic factor allows the model to explore the environment enough to compute aleatoric uncertainty estimates that make sense (this phenomenon is seen in the previous section for agents with high $\beta$ values), and the impact of these estimates on performance is dependent upon $\lambda$. For smaller $\lambda$, we hypothesize that when epistemic factor begins to increase, the agent is able to effectively use the more accurate aleatoric uncertainty estimates that it computes to both successfully solve the environment task and not fall often. However, for larger $\lambda$, we hypothesize that the more accurate aleatoric uncertainty estimates end up making the agent *too* conservative, yielding a lower number of falls but also a smaller final average reward due to an inability to solve the task. Thus, there is a "sweet spot" of high $\beta$ (1.0-2.0) and low-medium $\lambda$ (0.0-0.5) that allows for enough exploration to obtain useful aleatoric uncertainty estimates and also does not place too much weight on these estimates such that the agent is overly conservative. It would be interesting to later explore if these patterns are different in a larger environment in which the agent likely has more of a need for exploration in order to learn Q-values (and not just uncertainties of Q-values) effectively.



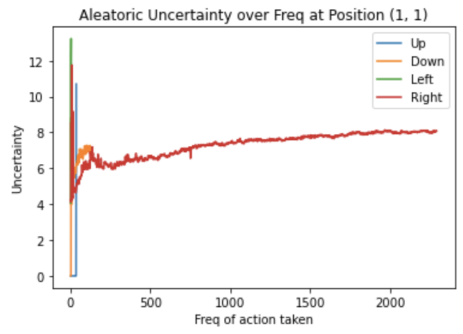


*Figure 8.* Environment is Stochastic Fall with Reward = -15 **(Top:)** UADQN Aleatoric Uncertainty vs. Action Frequency for state 1,0 with $\beta = 0.0$ and $\lambda = 0.5$ **(Bottom:)** UADQN Aleatoric Uncertainty vs. Action Frequency for state 1,1 with $\beta = 0.0$ and $\lambda = 0.5$