Epistemic Uncertainty Estimation for Human-in-the-Loop Reinforcement Learning

Ngorli Paintsil
Stanford University
ngorlip@stanford.edu

Natalie Greenfield
Stanford University
natgreen@stanford.edu

Abstract

Reinforcement Learning (RL) agents are increasingly being deployed in complex, real-world scenarios, ranging from autonomous driving systems to robotic surgery. However, the autonomous operation of these agents in high-stakes environments can be risky, necessitating the development of semi-autonomous systems that leverage human supervision. Effective shared autonomy requires RL agents to possess the capability to recognize their own limitations and to strategically query human supervisors when they encounter situations characterized by high uncertainty. This paper explores the integration of epistemic uncertainty estimation techniques into Human-in-the-Loop (HIL) RL systems, enabling agents to proactively seek guidance from human experts when their confidence in selecting optimal actions is low, particularly when those actions are likely to be incorrect.

We investigate two distinct uncertainty quantification methods: Deep Ensembles and Epistemic Neural Networks (Epinet), evaluating their performance within a custom-designed MiniGrid environment. This environment presents agents with both static and dynamic hazards, requiring strategic decision-making under conditions of partial observability. Our findings demonstrate that both approaches significantly enhance querying accuracy and reduce the likelihood of the agent entering a terminal state (e.g., colliding with a hazard) compared to a random querying baseline. Notably, Epinet exhibits superior performance across all evaluated metrics, demonstrating its potential for creating safer and more reliable shared autonomy systems.

This work addresses the critical challenge of enabling RL agents to effectively request human intervention. Traditional methods for shared autonomy often rely on fixed rules or heuristics, which can be overly conservative or fail to capture the nuances of situations where human input is truly necessary. We hypothesize that by equipping RL agents with robust epistemic uncertainty estimation capabilities, we can enable them to identify high-risk situations and defer control to a human expert, mitigating potential failures and improving overall system reliability.

Our research makes three primary contributions: (1) We develop and evaluate a novel framework for integrating epistemic uncertainty estimation into HIL RL systems. This framework allows agents to learn when their own decision-making might be suboptimal and to request assistance accordingly. (2) We conduct a comparative analysis of two state-of-the-art uncertainty quantification methods—Deep Ensembles and Epistemic Neural Networks—in the context of test-time querying. This comparison highlights the strengths and weaknesses of each approach for enabling effective shared autonomy. (3) We demonstrate that uncertainty-aware querying significantly improves safety metrics while maintaining reasonable query frequencies. This balance between agent autonomy and expert assistance is crucial for practical applications of shared autonomy.

The experimental results highlight the benefits of uncertainty-aware querying over random querying, with Epinet showing particularly strong performance. Epinet achieved a querying accuracy of 95.67%, substantially outperforming Deep Ensembles (85.76%) and a random baseline (69.13%). Furthermore, Epinet demonstrated a 4.12% terminal state avoidance rate, compared to 3.15% for Deep Ensembles and 0.85% for the random baseline. This indicates that Epinet is more effective at identifying situations where the agent is likely to make a mistake and should therefore consult the oracle.

This research has significant implications for the development of safer and more reliable autonomous systems. By enabling agents to strategically request human input based on their own uncertainty, we can create systems that are more robust to unexpected situations and less prone to catastrophic failures. The findings of this study suggest that Epistemic Neural Networks hold particular promise for enabling effective shared autonomy in a wide range of applications.

1. Introduction

Reinforcement Learning (RL) is growing in popularity and success in various fields such as game playing and robotics. However, deploying fully autonomous RL agents in critical applications, such as healthcare, autonomous driving, or aircraft autopilot, poses significant challenges due to the potential for catastrophic failures. In such high-risk or difficult-to-learn scenarios, a shared autonomy approach, where a human or expert supervisor can intervene, is often preferred over complete RL autonomy.

The fundamental challenge in shared autonomy lies in determining when an agent should give control to a human operator. Traditional approaches often rely on fixed rules or heuristics, which can be overly conservative or fail to capture the nuanced situations where human intervention is truly necessary [1]. For effective shared autonomy, an RL agent must possess the capability to identify situations where its own decision-making might be suboptimal or incorrect and it should request human intervention.

The ability to accurately and efficiently query is critical for enhancing safety, improving task performance, and creating human-robot trust in autonomous systems. While prior research has explored the use of epistemic uncertainty during the training phase to improve learning performance [2], there remains a significant gap in methods that specifically enable RL agents to request human input solely at test time based on model uncertainty.

This paper addresses this gap by focusing on the development of a system that can accurately and efficiently learn to query an expert when the agent is uncertain and would have otherwise chosen an incorrect action. We hypothesize that by equipping RL agents with robust epistemic uncertainty estimation capabilities, we can enable them to identify high-risk situations and giving control to a human expert, mitigating potential failures and improving overall system reliability.

Our work has three main contributions: (1) We develop and evaluate a framework for integrating epistemic uncertainty estimation into Human-in-the-Loop RL systems, (2) We compare two uncertainty quantification methods—Deep Ensembles and Epistemic Neural Networks—in the context of test-time querying, and (3) We demonstrate that uncertainty-aware querying significantly improves safety metrics while maintaining reasonable query frequencies.

1.1. Problem Statement

Formally, the research problem this paper addresses is: Can a reinforcement learning agent, given access to an expert human (modeled as a software oracle), learn to estimate its own epistemic uncertainty and accurately query the oracle in situations where its uncertain actions are likely to be incorrect?

This problem encompasses several key challenges: (1) accurately estimating epistemic uncertainty in partially observable environments, (2) learning appropriate thresholds for querying decisions, and (3) balancing agent autonomy with safety through human intervention.

2. Related Work

2.1. Human-in-the-Loop Reinforcement Learning

Human-in-the-Loop (HIL) reinforcement learning has emerged as a promising paradigm for deploying RL agents in safety-critical domains. Early work [3] introduced the concept of learning from human feedback, where humans provide evaluative signals to guide agent learning. Subsequent approaches expanded on this idea by incorporating human demonstrations [4] and preferences [5] to shape the agent's policy.

Several recent studies have explored uncertainty-aware querying in RL contexts. Da Silva et al. [6] proposed uncertainty-aware action advising, where agents proactively seek advice from humans or other agents based on uncertainty estimates. Though this method requires human trajectories during training. Singi et al. [7] extended this idea to robotic manipulation by developing a decision-making framework for human-in-the-loop agents using test-time uncertainty estimation. Their approach, however, relies solely on Bellman-based methods for quantifying uncertainty and does not leverage more expressive techniques such as deep ensembles or Epinet.

In contrast, our work focuses specifically on test-time uncertainty estimation and querying and leveraging advanced epistemic models to improve agent performance through selective, on-demand expert input.

2.2. Uncertainty Estimation in Deep Learning

Uncertainty quantification in deep learning has received significant attention in recent years, with methods broadly categorized into Bayesian and non-Bayesian approaches. Bayesian methods, such as Monte Carlo Dropout [8] and Variational Inference [9], provide principled frameworks for uncertainty estimation but often come with computational overhead.

Deep Ensembles [10] have emerged as a practical alternative, achieving strong performance by training multiple models with different initializations and using their disagreement as a proxy for uncertainty. This approach has been successfully applied across various domains, including computer vision and natural language processing.

More recently, Osband et al. [11] introduce the Epinet in their paper Epistemic Neural Networks (Epinet). The Epinet augments standard neural networks with stochastic indices that modulate the network's output, enabling uncertainty estimation with a single model rather than requiring

multiple independent models.

2.3. Uncertainty-Aware RL for Human Assistance

The intersection of uncertainty estimation and reinforcement learning has been explored primarily in the context of exploration and safe learning. Uncertainty-guided exploration methods [12] use epistemic uncertainty to drive exploration in novel states, while safe RL approaches [13] incorporate uncertainty to avoid potentially dangerous actions.

Recent work has begun to explore uncertainty estimation for human assistance in RL. Reddy et al. [14] developed methods for learning when to ask for help in interactive settings, while Bobu et al. [15] explored shared autonomy in robotic manipulation tasks. However, these approaches often rely on domain-specific heuristics rather than principled uncertainty estimation methods.

Our work builds upon these foundations by systematically comparing different uncertainty estimation methods in the specific context of test-time querying for shared autonomy applications.

3. Methods

3.1. MiniGrid Environment

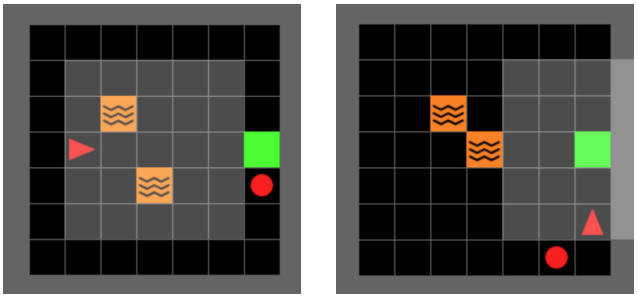
To train and evaluate the agent, we designed a custom MiniGrid environment that creates scenarios requiring strategic decision-making under uncertainty. This environment features both static and dynamic hazards that challenge the agent's ability to navigate safely while progressing toward a goal.

3.1.1 Environment Design

The environment is a compact 7×7 grid world where the agent must navigate toward a fixed goal location while avoiding obstacles. Within the central 3×3 region of the grid, two lava tiles are randomly placed at the beginning of each episode. These tiles represent static hazards that immediately terminate the episode if the agent steps onto them. The random placement prevents the agent from memorizing safe paths and forces adaptation to different configurations.

In addition to the lava tiles, the environment includes a single moving ball representing a dynamic hazard. The ball is randomly placed in the grid at episode initialization and moves with a random initial direction. When the ball collides with a lava square or wall, it changes direction randomly, creating unpredictable movement patterns that increase the complexity of navigation decisions.

The agent operates under partial observability, with a 5×5 field of view centered on its current position. This constraint forces the agent to make decisions based on incomplete information, creating a highly stochastic environment with partial observability where querying an oracle can improve agent performance and where uncertainty estimation becomes crucial for safe navigation.



(a) View 1 (b) View 2

Figure 1: Minigrid environments shown side-by-side.

3.1.2 Formal Problem Definition

We formalize our MiniGrid-based environment as a Partially Observable Markov Decision Process (POMDP), defined by the tuple $(S, A, T, R, \Omega, O, \gamma)$:

- **State space** S includes the agent's position and orientation, the positions of two randomly placed static lava tiles within the central 3×3 grid region, the position and direction of a dynamically moving hazard (a ball), and a fixed goal location.
- **Action space** $A = \{\texttt{left}, \texttt{right}, \texttt{forward}\}$ governs agent movement, where the agent can turn left, turn right, or move forward at any timestep.
- **Transition function** T is deterministic for the agent, while the ball moves stochastically, bouncing with a random new direction upon hitting walls or hazards.
- **Observations** Ω consist of a 5×5 egocentric field of view centered on the agent, inducing partial observability. The observation function O is deterministic but lossy due to limited perception.
- **Discount factor** $\gamma = 0.99$

This setting emphasizes planning under uncertainty and adaptation to stochastic hazards.

3.1.3 Reward Structure

The reward function employs potential-based reward shaping to encourage progress toward the goal while preserving the optimal policy structure. The shaped reward at each step is defined as:

$$\text{Reward} = r + \Phi(s) - \gamma\Phi(s') \tag{1}$$

where $\gamma = 0.99$ is the discount factor,

$$r = \begin{cases} -1 & \text{if the episode is terminated by hazard} \\ 1 & \text{if agent reaches the goal} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

and $\Phi(s)$ represents the Manhattan distance from state s to the goal location.

Each episode is capped at 100 steps, after which the episode terminates if the goal is not reached, providing a -1 reward for timeout.

3.2. Base Model

For each uncertainty estimator used in this research, a consistent base model was employed to ensure fair comparison across approaches. The foundation model is a Proximal Policy Optimization (PPO) agent with a Gated Recurrent Unit (GRU) architecture, specifically designed to handle the partial observability inherent in the MiniGrid environment. The PPO-GRU agent utilizes a recurrent actor-critic architecture optimized for sequential decision-making under partial observability. The observation space is first processed by a Convolutional Neural Network (CNN) encoder consisting of two convolutional layers with ReLU activations, designed to extract relevant visual features from the observation window. The CNN architecture uses 32 and 64 filters respectively, both with 3×3 kernels and padding=1 to preserve spatial dimensions. An adaptive average pooling layer reduces the spatial dimensions to 4×4, resulting in a fixed-size feature representation of 64×4×4 = 1024 dimensions regardless of input size variations. These extracted CNN features are then fed into a multi-layer GRU network with 512 hidden units across 2 layers, enabling the agent to capture and leverage temporal dependencies within the sequence of partial observations. The GRU's recurrent nature allows the agent to maintain an internal state that accumulates information over time, effectively compensating for the partial observability constraint in the MiniGrid environment.

The output of the GRU is processed by separate policy (actor) and value (critic) heads, each implemented as two-layer fully connected networks. The actor head consists of a linear layer from the 512-dimensional GRU output to 256 hidden units with ReLU activation, followed by a final linear layer producing logits for the three possible actions. Similarly, the critic head uses the same 512→256→1 architecture with ReLU activation to estimate the state value function for advantage computation in the PPO algorithm.

The PPO-GRU agent was trained for a total of 300,000 timesteps with a learning rate scheduler to allow sufficient exploration. To improve policy robustness to oracle actions at test time we implemneted regularization at training time. Firstly we reset the GRU to the initial hidden state 5% of step and secondly we changed the starting direction of the agent in 10% of episodes. With regularization our baseline achieved a success rate of 74.4%. Critically, during training, the PPO-GRU agent did not have access to the oracle, ensuring it learned to navigate the environment independently before uncertainty estimation methods were integrated.

3.3. Human-in-the-Loop Pipeline

The Human-in-the-Loop pipeline integrates uncertainty estimation with action selection to enable strategic querying of expert knowledge. At each timestep during evaluation, the current state observation is fed into both the baseline PPO agent and the chosen uncertainty estimator.

The pipeline operates through the following decision process: 1) The current state s_t is processed by the trained PPO agent to generate action probabilities $\pi(a|s_t)$ 2) Simultaneously, the uncertainty estimator computes an uncertainty score u_t for the current state 3) If the uncertainty value exceeds a tuned threshold τ (specific to each uncertainty method), the agent queries the oracle for the optimal action a_t^* 4) Otherwise, the agent executes its own action $a_t \sim \pi(\cdot|s_t)$

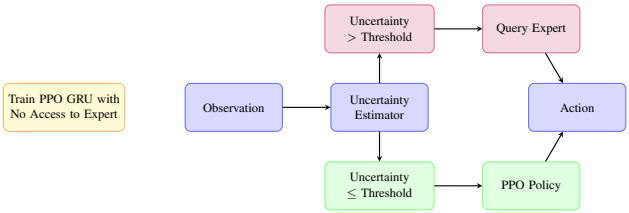


Figure 2: Human-in-the-Loop Pipeline

The oracle is implemented as an A* pathfinding algorithm with complete environment visibility, serving as a proxy for human expert knowledge. This oracle can compute optimal actions by considering all static and dynamic hazards, the goal location, and the current agent position.

Threshold values τ were determined through grid search optimization on a validation set, balancing query frequency with performance improvements. The goal was to maintain query rates between 0.2-0.4 queries per episode to ensure the agent remains largely autonomous while benefiting from strategic expert intervention.

3.4. Deep Ensemble

The first uncertainty estimation method employed was Deep Ensembles, based on earlier work done for estimating epistemic uncertainty at training time [6]. This approach leverages the disagreement among multiple independently trained models as a measure for epistemic uncertainty.

The ensemble consists of 5 independent PPO-GRU agents, each with identical architecture but different random weight initializations. Each ensemble member was trained independently on the same MiniGrid environment using the training procedure described in Section 3.2, ensuring diversity in the learned policies while maintaining consistent performance levels.

During training, each ensemble member experiences different sequences of random environment configurations leading to diverse learned representations and decision boundaries. This diversity is crucial for effective uncertainty estimation, as disagreement among ensemble members indicates regions of the state space where the optimal action is unclear, indicating uncertainty.

At test time, uncertainty estimation proceeds as follows: 1) The current state s_t is fed to all 5 ensemble members. 2) Each member i produces action logits $z_i(s_t) \in \mathbb{R}^3$. 3) The ensemble uncertainty is computed as the variance across ensemble predictions:

$$u_t = \frac{1}{|A|} \sum_{a=1}^{|A|} \text{Var}(z_{1:5}^{(a)}(s_t)) \tag{3}$$

where $z_{1:5}^{(a)}(s_t)$ represents the logits for action a across all ensemble members.

High variance indicates disagreement among ensemble members, suggesting the agent encounters a state where the optimal action is uncertain. This uncertainty signal is then compared against the threshold $\tau_{ensemble} = 0.49$ to determine whether to query the oracle. The Deep Ensemble approach served as a robust but simple baseline for uncertainty estimation.

3.5. Epinet

The second uncertainty estimation method uses the Epinet introduced by Osband et al. [11] in *Epistemic Neural Networks*. Epinet provides ensemble-like epistemic uncertainty without training multiple independent models, thereby offering a far more computationally efficient alternative to Deep Ensembles. The key innovation is the introduction of a latent stochastic index z that modulates the network's output, effectively simulating ensemble diversity within a single model.

We attach an Epinet value head to the frozen CNN-GRU policy, using the GRU hidden state $h_t \in \mathbb{R}^d$ as input. The Epinet head consists of: 1) A small shared MLP feature extractor with two 64-unit layers that processes h_t. output. 2) A continuous index $z \sim \mathcal{N}(0, I_{16})$ of dimension $d_z = 16$ sampled once per forward pass. 3) A trainable output head and a frozen prior network, both conditioned on (h_t, z), produce scalar outputs that are summed to yield the final value prediction:

$$v = f_{\text{train}}(h_t, z) + \text{prior\_scale} \cdot f_{\text{prior}}(h_t, z)$$

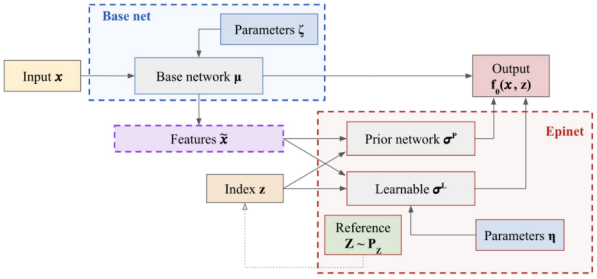


Figure 3: Epinet Architecture

We fix `prior_scale` = 1.0 and stop gradients through the prior branch to preserve the Bayesian inductive bias of the original formulation.

To train the Epinet, we run the frozen PPO-GRU policy for 50,000 environment steps, extracting GRU features ϕ_t and the action entropy at each step. Action entropy measures the uncertainty in the agent's policy's action distribution:

$$H(\pi_t) = -\sum_a \pi_t(a) \log \pi_t(a)$$

defined as the entropy of its action distribution. It serves as a proxy for the agent's internal confidence: high entropy indicates indecision, while low entropy suggests confident action selection.

This serves as a proxy for the agent's internal confidence: high entropy indicates indecision, while low entropy suggests confident selection. These entropy values are used as regression targets, enabling the Epinet to learn to estimate policy uncertainty from internal features. We minimize mean squared error using the Adam optimizer (learning rate 10^{-3}, batch size 256) for 100 epochs. The model is implemented in JAX with Haiku and Optax, using the original Epinet training library.

At test time, uncertainty estimation is performed by sampling multiple indices $z_1, z_2, ..., z_M$ where $M = 10$ and computing action logits for each sampled index $\pi_i(s_t) = \text{softmax}(f(s_t, z_i))$. Epistemic uncertainty is quantified as the sample standard deviation across these predictions:

$$u_t = \sqrt{\text{Var}(\{ f_\theta(h_t, z_i) \}_{i=1}^{M})}$$

A high value of u_t indicates internal disagreement in the value head, and thus low epistemic confidence.

The threshold for Epinet querying was set to $\tau_{Epinet} = 0.23$ based on validation set optimization. Epinet's more fine-grained uncertainty estimates typically result in a different uncertainty distribution compared to Deep Ensembles, resulted in the need for separate threshold tuning.

4. Experiments and Results

4.1. Experimental Setup

Human-in-the-loop pipeline eval was conducted using the custom MiniGrid environment described in Section 3.1. Each uncertainty estimation method was evaluated across 1000 episodes to compute final performance statistics.

4.2. Metrics

We selected three key metrics to comprehensively analyze our uncertainty estimators:

Querying Accuracy: The percentage of oracle queries where the agent's intended action differed from the oracle's recommended action. This metric captures the quality of uncertainty estimation by measuring whether the agent queries the oracle precisely when it would have made a suboptimal decision. High querying accuracy indicates that the uncertainty estimator successfully identifies situations where the agent's knowledge is insufficient.

Terminal State Avoidance: The percentage of episodes where the agent correctly chose to query the oracle instead of taking an action that would have led to episode termination (stepping on lava or collision with the moving ball). This safety-critical metric directly measures the agent's ability to avoid failures through querying.

Success Rate: The overall success rate in reaching the goal within the 100-step episode limit. This metric evaluates whether the uncertainty-aware querying system maintains or improves task performance compared to the baseline agent.

Query Frequency: The average number of oracle queries per episode, serving as a sanity check for threshold tuning. If this number is too high the agent becomes overly dependent on the oracle; if too low the agent rarely benefits from expert knowledge.

4.3. Baseline Comparison

To establish the value of principled uncertainty estimation, we implemented a random querying baseline that queries the oracle with the same frequency as our uncertainty-based methods but without considering uncertainty. This baseline uses the same threshold optimization procedure but applies the threshold to uniformly random values rather than uncertainty estimates.

4.4. Results

The experimental results demonstrate clear benefits of uncertainty-aware querying over random querying, with Epinet showing superior performance across all metrics.

Querying Accuracy: Epinet achieved exceptional querying accuracy at 95.67%, substantially outperforming both the Deep Ensemble (85.76%) and the random baseline (69.13%). This 10-point improvement over Deep Ensembles and 26-point improvement over random querying demonstrates Epinet's superior ability to identify situations where the agent lacks sufficient knowledge to make optimal decisions.

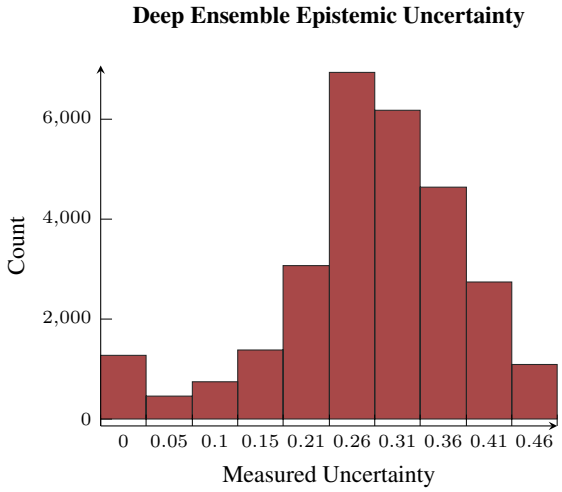


Figure 4: Histogram of sample values across measured uncertainty intervals.

Terminal State Avoidance: Both uncertainty estimation methods showed significant improvements in safety metrics. Deep Ensemble achieved 3.15% terminal state avoidance compared to 0.85% for random querying, while Epinet reached 4.12%. Though these absolute percentages appear low, they represent substantial relative improvements (3.7× and 4.8× respectively) in avoiding catastrophic failures.

Success Rate: Epinet demonstrated the highest task performance with a 75.70% success rate, compared to 72.40% for Deep Ensemble and 72.5% for random querying. The improvement suggests that uncertainty-aware querying not only enhances safety but can also improve overall task completion rates.

Query Frequency: All methods maintained reasonable query frequencies around 0.3 queries per episode, confirming that threshold optimization successfully balanced agent autonomy with expert assistance. The similar query rates across methods validate our experimental design and enable fair comparison of uncertainty estimation quality.

Table 1: Performance comparison of uncertainty estimation methods

Metric	Random	Deep Ensemble	Epinet
Querying Accuracy	69.13%	85.76%	95.67%
Terminal State Avoidance	0.85%	3.15%	4.12%
Success Rate	72.5%	72.40%	75.70%
Avg Queries per Episode	0.27	0.29	0.32

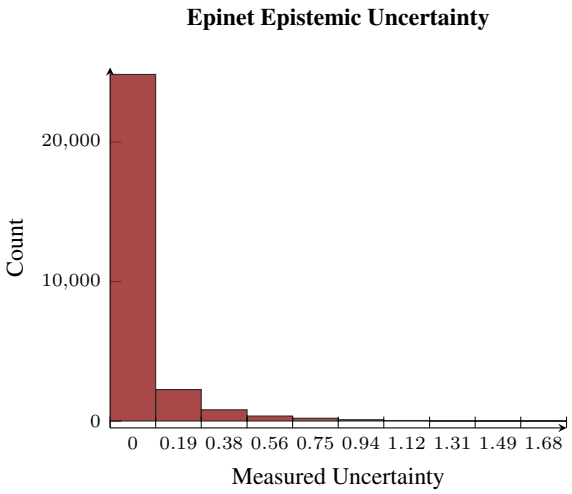


Figure 5: Histogram of sample values across measured uncertainty intervals.

4.5. Uncertainty Distribution Analysis

The uncertainty distributions of the Deep Ensemble and Epinet methods exhibit distinct shapes and ranges. The Deep Ensemble displays a approximately normal distribution with uncertainty values spanning 0.0 to 0.51, peaking around 0.26-0.31. Specifically, the peak bin (0.257-0.308) contains 6,937 samples, representing about 25% of all samples. This normal distribution without strong distinction may limit the Deep Ensemble's ability to effectively differentiate between moderately and highly uncertain samples.

In contrast, Epinet presents a heavily right-skewed distribution with a significantly wider dynamic range, extending from 0.004 to 1.87. This distribution is dominated by low-uncertainty samples, with 24,853 samples (approximately 89% of the total) concentrated in the lowest bin (0.004-0.190). The extended tail, however, provides a clear identification genuinely uncertain samples, as only 540 samples exhibit uncertainty values above 0.75.

These characteristics suggest that Epinet's architecture more effectively captures the underlying uncertainty structure inherent in the problem domain. The bimodal nature of epistemic uncertainty, where most samples are either confidently predictable or clearly uncertain, is better reflected in Epinet's distribution than in the Deep Ensemble's more uniform spread. The findings indicate that effective uncertainty quantification methods should prioritize creating distinct separation between uncertainty regimes rather than generating smooth, continuous distributions, which can cause difficult separation between genuinely uncertain samples with those that are merely moderately difficult to predict.

Table 2: Success rates across different agent configurations

Method	Success Rate (%)
Base Model with No Regularization	81.1
Base Model with No Regularization + Epinet + Oracle	63.9
Base Model	74.4
Base Model + Epinet + Oracle	75.7

4.6. Success Rate Investigation

During the development of our base model, we observed a significant degradation in performance when incorporating oracle querying without appropriate regularization. As shown in Table 2, the unregularized base model achieved an 81.1% success rate, whereas the same model with oracle queries dropped sharply to 63.9%. Qualitative analysis of the resulting trajectories revealed that the agent frequently began spinning in place after control was returned from the oracle. This behavior, which deviated greatly from the learned policy, likely stemmed from two factors: (1) the oracle, implemented using A* search, operates with full observability and thus introduces state transitions unfamiliar to the policy, which was trained in a partially observable setting; and (2) oracle actions may disrupt the internal GRU hidden state, leading to unstable behavior. Manually resetting the GRU hidden state after oracle handover qualitatively reduced spinning.

To address these issues, we introduced a regularization strategy that resets both the GRU hidden state and the agent's initial direction upon resuming control. Although this regularization slightly degraded the base model's performance (from 81.1% to 74.4%), it substantially improved the performance of the oracle-augmented model, which achieved a 75.7% success rate—surpassing the unregularized oracle variant. These results led us to use the regularized model as our base model for all other experiments and suggests that the observed degradation was indeed due to a combination of out-of-distribution state transitions and hidden state corruption. Despite these improvements, however, none of the oracle-augmented models consistently outperformed the original base model.

5. Discussion

5.1. Method Comparison

The superior performance of Epinet across all metrics can be attributed to several factors. First, Epinet's shared feature representation allows for more nuanced uncertainty estimation, as different epistemic heads can specialize while leveraging common low-level features. This architectural advantage enables more precise identification of genuinely uncertain states.

Second, the stochastic index sampling mechanism in Epinet provides a more direct estimate of epistemic uncer-

tainty compared to the ensemble disagreement used in Deep Ensembles. The 16 epistemic heads in Epinet can capture finer-grained uncertainty patterns than the 5 independent models in the Deep Ensemble approach.

Third, Epinet's training procedure explicitly encourages diversity among epistemic heads through the regularization term, potentially leading to better calibrated uncertainty estimates compared to the implicit diversity arising from different initialization in Deep Ensembles.

5.2. Safety Implications

The improvements in terminal state avoidance, while appearing modest in absolute terms, represent significant safety enhancements. In safety-critical applications, even small reductions in failure rates can have substantial impact on system reliability and user trust.

The ability of both uncertainty estimation methods to identify potentially catastrophic situations (terminal states) before they occur demonstrates the practical value of uncertainty-aware querying for shared autonomy applications. This capability is particularly valuable in domains where recovery from failures is difficult or impossible.

5.3. Computational Considerations

While Epinet demonstrates superior performance, computational efficiency considerations are important for practical deployment. Epinet requires only a single model with multiple head sampling, making it more memory-efficient than Deep Ensembles. However, the need to sample multiple indices and compute predictions for each adds some computational overhead compared to standard single-model inference.

Deep Ensembles, while requiring multiple models, benefit from embarrassingly parallel computation across ensemble members. In scenarios with abundant computational resources, this parallelizability might offset the increased memory requirements.

5.4. Limitations

Several limitations should be acknowledged in our current approach. First, the MiniGrid environment, while designed to test uncertainty estimation, represents a simplified scenario compared to real-world applications. The discrete action space, deterministic dynamics (except for ball movement), and limited state complexity may not capture the full challenges of uncertainty estimation in more complex domains.

Second, our oracle implementation using A* pathfinding assumes perfect knowledge and optimal decision-making. Real human experts may make suboptimal decisions or have their own uncertainties, potentially affecting the validity of our querying accuracy metric.

Third, the threshold optimization procedure, while systematic, requires validation data and may not generalize across different environments or task distributions. Adaptive threshold selection methods could improve robustness and transferability.

6. Conclusion and Future Work

This work demonstrates the effectiveness of epistemic uncertainty estimation for enabling strategic human-machine collaboration in reinforcement learning systems. Our findings show that principled uncertainty estimation, particularly through Epinet, can significantly improve an agent's ability to identify situations requiring human intervention while maintaining autonomous operation in well-understood scenarios.

The key contributions of this work include: (1) a systematic comparison of uncertainty estimation methods for test-time querying in RL, (2) demonstration that uncertainty-aware querying improves both safety and performance metrics, and (3) evidence that Epinet provides superior uncertainty estimation capabilities compared to Deep Ensembles in this context.

Future work should explore several promising directions. First, evaluation in more complex and realistic environments, including continuous control tasks and real-world robotic applications, would strengthen the generalizability of our findings. Second, investigation of adaptive threshold selection methods could eliminate the need for manual threshold tuning and improve transferability across different tasks.

Third, incorporating human factors considerations, such as cognitive load and trust dynamics, could lead to more effective human-machine collaboration strategies. Finally, exploring hierarchical uncertainty estimation methods that can operate at different temporal scales could enable more sophisticated querying strategies for long-horizon tasks.

The integration of uncertainty-aware querying into practical shared autonomy systems represents a promising path toward safer and more reliable AI deployment in high-stakes domains. Our work provides a foundation for future research in this critical area.

7. Contributions & Acknowledgements

Ngorli Paintsil and Natalie Greenfield contributed equally to all aspects of this research, including the creation of the MiniGrid environment, pipeline development, implementation of algorithms, and the writing of the paper.

References

[1] L. Methnani, A. Aler Tubella, V. Dignum, and A. Theodorou, "Let me take over: Variable autonomy for meaningful hu-

man control," *Frontiers in Artificial Intelligence*, vol. 4, p. 737072, 2021.

[2] I. Osband, C. Blundell, A. Pritzel, and B. Van Roy, "Deep exploration via bootstrapped DQN," in *Advances in Neural Information Processing Systems*, 2016, pp. 4026–4034.

[3] W. B. Knox and P. Stone, "Interactively shaping agents via human reinforcement: The TAMER framework," in *Proceedings of the Fifth International Conference on Knowledge Capture*, 2009, pp. 9–16.

[4] J. Ho and S. Ermon, "Generative adversarial imitation learning," in *Advances in Neural Information Processing Systems*, 2016, pp. 4565–4573.

[5] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," in *Advances in Neural Information Processing Systems*, 2017, pp. 4299–4307.

[6] F. L. Da Silva, P. Hernandez-Leal, B. Kartal, and M. E. Taylor, "Uncertainty-aware action advising for deep reinforcement learning agents," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, 2020, pp. 5792–5799.

[7] S. Singi, Z. He, A. Pan, S. Patel, G. A. Sigurdsson, R. Piramuthu, S. Song, and M. Ciocarlie, "Decision making for human-in-the-loop robotic agents via uncertainty-aware reinforcement learning," *arXiv preprint arXiv:2303.06710*, 2023.

[8] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *International Conference on Machine Learning*, 2016, pp. 1050–1059.

[9] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural network," in *International Conference on Machine Learning*, 2015, pp. 1613–1622.

[10] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Advances in Neural Information Processing Systems*, 2017, pp. 6402–6413.

[11] I. Osband, Z. Wen, S. Mohammad Asghari, V. Dwaracherla, M. Ibrahimi, X. Lu, and B. Van Roy, "Epistemic neural networks," *arXiv preprint arXiv:2107.08924*, 2023.

[12] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," in *International Conference on Machine Learning*, 2017, pp. 2778–2787.

[13] J. García and F. Fernández, "A comprehensive survey on safe reinforcement learning," *Journal of Machine Learning Research*, vol. 16, no. 1, pp. 1437–1480, 2015.

[14] S. Reddy, A. D. Dragan, and S. Levine, "Learning human objectives by evaluating hypothetical behavior," in *International Conference on Machine Learning*, 2020, pp. 8020–8029.

[15] A. Bobu, A. Bajcsy, J. F. Fisac, S. Deglurkar, and A. D. Dragan, "Learning to take good pictures of people," in *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, 2018, pp. 121–129.