

When to ASK: Uncertainty-Gated Language Assistance for Reinforcement Learning

Juarez Monteiro¹, Nathan Gavenski², Gianluca Zuin¹ and Adriano Veloso¹

¹*Kunumi Institute*

Belo Horizonte, MG, Brazil

{juarez, gianluca, adriano}@kunumi.com

²*King's College London*

London, England, UK

nathan.schneider_gavenski@kcl.ac.uk

Abstract—Reinforcement learning (RL) agents often struggle with out-of-distribution (OOD) scenarios, leading to high uncertainty and random behavior. While language models (LMs) contain valuable world knowledge, larger ones incur high computational costs, hindering real-time use, and exhibit limitations in autonomous planning. We introduce Adaptive Safety through Knowledge (ASK), which combines smaller LMs with trained RL policies to enhance OOD generalization without retraining. ASK employs Monte Carlo Dropout to assess uncertainty and queries the LM for action suggestions only when uncertainty exceeds a set threshold. This selective use preserves the efficiency of existing policies while leveraging the language model’s reasoning in uncertain situations. In experiments on the FrozenLake environment, ASK shows no improvement in-domain, but demonstrates robust navigation in transfer tasks, achieving a reward of 0.95. Our findings indicate that effective neuro-symbolic integration requires careful orchestration rather than simple combination, highlighting the need for sufficient model scale and effective hybridization mechanisms for successful OOD generalization.

Index Terms—Language Models, Reinforcement Learning

I. INTRODUCTION

Generalization is a vital requirement for learning systems, as it enables them to function beyond their training data. In learning, these systems assume that the observed data are drawn from an independent and identically distributed distribution, and that this distribution holds approximately during evaluation and testing. Yet, agentic applications do not exhibit this behavior. During testing, even if the initial state is included in the training process, the sequential nature of episodes may push the agent out of distribution (OOD) [1].

Generalizing to OOD states is challenging for agents because they lack prior information about how to act in these states. When predicting in OOD situations, agents exhibit high *uncertainty* and may behave randomly or be biased. Thus, ensuring that agents behave appropriately in OOD states is also a vital component to develop safe and trusted systems. Some works [2, 3, 4] aim to ensure generalization by employing expert systems or incorporating prior knowledge to guide agent behavior in OOD states. Recent advances in *language models* (LM) offer a promising avenue: pre-trained on vast corpora, these models encode rich world knowledge and common-sense reasoning that can potentially guide agents when encountering OOD states. However, integrating *large* LM (LLMs) into real-time decision-making systems poses computational challenges.

While planners offer a safe alternative [5, 6], they require careful manual encoding of domain knowledge, limiting their

adaptability to new tasks. Smaller LMs, small to medium-sized models, offer an elegant trade-off: they retain much of the world knowledge and reasoning of larger models while being computationally efficient enough for real-time decision-making [7]. Unlike planners, LMs provide flexible guidance across diverse scenarios without requiring task-specific adjustments, leveraging existing trained policies to improve practical utility across different agents and domains.

Approaches to improving generalization can be broadly categorized into *intrinsic* and *extrinsic* methods. Intrinsic methods, such as domain randomization, architectural innovations, or curriculum learning, modify the training process to encourage broader generalization, but require retraining agents with potentially larger networks or additional data [1]. Extrinsic methods, conversely, augment existing policies at inference time without requiring retraining or additional training data.

In this work, we explore an extrinsic approach that integrates LMs with already-trained PPO agents to improve generalization in OOD settings, which we call *Adaptive Safety through Knowledge* (ASK)¹. Our method leverages the semantic understanding and world knowledge encoded in LMs to guide reinforcement learning (RL) agents when they encounter unfamiliar states, combining the adaptability of RL with the implicit knowledge of language models without requiring retraining or architectural modifications.

II. PROBLEM FORMULATION

Reinforcement learning is a problem where a policy π learns parameters θ via interacting with the environment and its reward signal r [8]. In RL, the reward signal $r : S \times A \mapsto \mathbb{R}$, where S is the state space and A is the action space, allows the agent π_θ to learn the probability of each action $a \in A$ given the current state $s \in S$ by maximizing the immediate reward. In this setting, agents typically do not observe the entire state space, resulting in a situation in which they lack state information and may perform highly uncertain actions. To better formulate this change in unobserved states, we depart from the initial RL formulation and adopt one in which different contexts may not only alter how states are represented but also modify the transition functions that govern the environment’s dynamics.

¹**Data and Code Availability.** Supplementary material is available at <https://kclpure.kcl.ac.uk/ws/portalfiles/portal/368422362/appendix.pdf>. The code is available at <https://github.com/jrzmnt/ask-rl>.

A. Contextual Markov Decision Process

We formulate our problem as a *Contextual Markov Decision Process* (CMDP), a tuple $\mathcal{M} = \langle S, A, O, r, T, C, \phi \rangle$, where O is the observation space, r is the immediate reward function $r : O \times A \mapsto \mathbb{R}$, T is the transition function $T : O \times A \mapsto O$, C is the set of contexts, and ϕ is a projection function $\phi : S \times C \mapsto O$ [1]. Note that in CMDPs, the policy only observes the state produced by ϕ , that the current context $c \in C$ remains the same in an episode, and the reward function r now operates under observations and not states. In other words, the policy perceives only one context c at a time in which all states $s \in S$ are projected into it $\{\phi(s, c) \mid s \in S\}$.

In this formulation, the context serves as the seed, ID, or parameter vector that determines the current environment. Hence, it should not change within an episode; it should change only between episodes. More importantly, the context distribution $p(c)$ defines the training, evaluation, and testing collections of each task or environment instance. Most often, this distribution remains uniform through all collections; however, this is not true for all environments. Finally, we do not assume that the policy observes the context. That is, the policy does not perceive its context. Since two different contexts (c, c') might map two different states (s, s') into the same observation $(\phi(s, c) = \phi(s', c'))$, it is not always the case that the policy can acknowledge in which context it is in.

B. Uncertainty in Neural Networks

Measuring uncertainty helps researchers assess the extent to which model predictions carry weight given their prior experience. The two most common types of uncertainty in machine learning are: (i) *epistemic*, which is inherent to models, caused by a lack of training data, hence reducible with more data; and (ii) *aleatoric*, caused by inherent noise or ambiguity in data, thence irreducible with more data [9]. Epistemic uncertainty at an observation $o \in O$ is high for a previously unseen o , and decreases as o becomes part of the training set [10]. Kirsch et al. [11] show that this behavior conforms with using mutual information in Bayesian models and deep ensembles, and Postels et al. [12] as feature-space density in deterministic models as surrogates for epistemic uncertainty. Aleatoric uncertainty at an observation $o \in O$ is high for ambiguous or noisy samples [9]. For example, if multiple labels are observed at o , the aleatoric uncertainty will be high, thus, adding more data has no effect on it.

Uncertainty is usually measured under *Bayesian models* or *ensemble methods* [13]. Bayesian models provide a principled way to quantify uncertainty by using prior distributions over model parameters to infer the posterior distribution given training data. Ensemble models compute the average of their outputs, which are then used to estimate the entropy of the averaged vectors. However, both approaches incur high training costs. To represent uncertainty, these approaches double the number of parameters for the same network size and require more time to converge [14]. Therefore, a solution for approximating the behavior of Bayesian models and ensemble methods is to apply dropout to the same neural network during

inference N times [14]. By applying dropout during inference, we obtain multiple stochastic predictions from the same network, creating an implicit ensemble without additional parameters. The variance across these predictions provides an estimate of epistemic uncertainty:

$$\mathcal{U}_e(o) = \frac{1}{N} \sum_{i=1}^N \sum_a p_i(a \mid o) \log \frac{p_i(a \mid o)}{\bar{p}(a \mid o)}, \quad (1)$$

where $p_i(a \mid o)$ is the action distribution from the i -th forward pass with dropout, and $\bar{p}(a \mid o) = 1/N \sum_{i=1}^N p_i(a \mid o)$ is the mean distribution (we omit the agent notation π_θ from $p_i^{\pi_\theta}$ for simplicity). On the other hand, the mean prediction entropy captures aleatoric uncertainty:

$$\mathcal{U}_a(o) = \frac{1}{N} \sum_{i=1}^N \sum_a -p_i(a \mid o) \log p_i(a \mid o). \quad (2)$$

This Monte Carlo Dropout (MC Dropout) approach offers a lightweight alternative to full Bayesian methods or explicit ensembles, requiring only repeated forward passes through an already-trained network. To preserve the extrinsic nature and interpretability of ASK, our gating mechanism is implemented as a fixed uncertainty threshold, thus avoiding additional learned components or retraining. In this setting, uncertainty is used as a trigger for intervention rather than as a calibrated decision signal, making precise calibration less critical.

III. ADAPTIVE SAFETY THROUGH KNOWLEDGE

In this work, we propose Adaptive Safety through Knowledge (ASK), which leverages language models as an extrinsic method to improve the generalization of RL agents without any retraining. ASK's main contribution lies in quantifying uncertainty to identify when an agent encounters OOD observations and selectively querying for guidance only in these high-uncertainty scenarios. This approach combines the efficiency of a trained RL policy with the world knowledge of language models, activating it only when needed. By measuring both epistemic and aleatoric uncertainty at each observation, ASK determines when the agent lacks sufficient experience (high epistemic uncertainty) or faces inherently ambiguous states (high aleatoric uncertainty). In such cases, rather than allowing the agent to act with high uncertainty, we leverage the semantic understanding to provide an informed action recommendation.

We first summarize ASK. Given a trained RL policy π_θ , a language model $\mathcal{G}_\phi$, and a set of evaluation contexts $C' \in C$, where C' are contexts not seen during training or during evaluation, ASK operates as follows. For each context $c \in C'$, it initializes an environment instance and retrieves the initial observation o . At each step, the agent computes an action a using its learned policy $\pi_\theta(o)$. However, before executing this action, ASK computes the total uncertainty $\mathcal{U}_e(o) + \mathcal{U}_a(o)$ using MC Dropout (Eqs. 1 and 2). If the total uncertainty exceeds a predefined threshold τ , we override the agent's action by querying $\mathcal{G}_\phi$ with a prompt Pr and the current observation to obtain an alternative action $\mathcal{G}_\phi(Pr, o)$. This intervention occurs only when the agent is uncertain, allowing the trained policy to handle familiar observations while delegating high-uncertainty decisions to the language model's world knowledge. The agent then executes the selected action, receives the next observation

from the environment via the transition function T , and repeats this process until it reaches the goal state g .

A. State Representation and Prompting

Smaller language models do not share the same capacities as their larger counterparts. As model size decreases, its ability to follow complex instructions, perform multi-step reasoning, and robustly handle varied prompt formats [7] also decreases. For example, when using LLMs, prompts that described a grid-like environment using a matrix would work perfectly. Yet, when using smaller models, we had to simplify the observation description to better fit the model’s capacity. Therefore, careful prompt engineering is essential to maximize the utility of LMs in ASK. ASK requires two key components: translating environmental observations into natural-language descriptions that models can interpret, and enforcing a structured output format that can be reliably parsed into discrete actions.

ASK prompt design balances informativeness with simplicity, providing the language model with just enough context to make informed decisions without overwhelming its reasoning capacity. The complete prompt consists of several structured sections: (i) a role definition establishing the task and explicit output constraints to prevent verbose responses; (ii) high-level decision rules prioritizing safety and goal-directed behavior; (iii) state information using position coordinates and local spatial relationships; and (iv) the original policy’s action suggestion as an “autopilot recommendation.” Appendix A provides the full prompt.

Crucially, we include the trained policy’s action recommendation in the prompt, framed as an “autopilot suggestion.” This allows the model to validate the policy’s choice when it aligns with safe, goal-directed behavior, or override it when the suggestion appears risky given the current state. By providing this suggestion, we leverage the RL policy’s learned behavior while allowing the LM to intervene based on its knowledge.

Finally, ASK enforces a strict output format to facilitate reliable parsing and provides a fallback mechanism when the LM fails to adhere to the format instructions (by defaulting to the policy’s action). Importantly, ASK does *not* fine-tune or train the language model. It relies solely on the LM’s pre-trained knowledge, maintaining its extrinsic nature.

B. Uncertainty Estimation

ASK quantifies uncertainty using MC Dropout to obtain both epistemic and aleatoric estimates (Eqs. 1 and 2). It computes the total uncertainty $\mathcal{U}_{\text{total}}(o) = \mathcal{U}_e(o) + \mathcal{U}_a(o)$ at each observation o by performing N forward passes through the trained policy network with dropout enabled. This provides a lightweight mechanism for identifying states in which the agent lacks confidence in its action selection.

Uncertainty serves as a natural trigger for external reasoning in OOD states, rather than as a replacement for learned behavior. High epistemic uncertainty indicates that the agent has insufficient experience with observations similar to o , precisely the scenario in which the agent is most likely to be out-of-distribution and where external guidance is most

valuable. High aleatoric uncertainty, on the other hand, signals inherent ambiguity in the observation itself, such as multiple valid paths or conflicting cues, where world knowledge and common-sense reasoning can help break ties.

By using uncertainty as the intervention criterion, ASK ensures that the LM is queried only when necessary. In low-uncertainty states, the trained RL policy operates autonomously, leveraging its learned behavior efficiently. In high-uncertainty states, where the agent exhibits high variance, the LM provides informed guidance based on its semantic understanding of navigation and safety. This selective intervention preserves the efficiency of the learned policy while mitigating the risks associated with OOD generalization.

The threshold τ provides a crucial trade-off between computational cost and safety. Querying the LM at every step would be computationally expensive and unnecessary, as the trained policy performs well in familiar states. Conversely, never querying the LM would leave the agent vulnerable to unsafe or random behavior in OOD scenarios. By setting τ empirically based on the uncertainty distribution observed during training, we balance policy autonomy with external guidance, invoking the LM only when the agent’s uncertainty warrants the additional computational overhead.

IV. EXPERIMENTS

We evaluate ASK on the FrozenLake-v1 environment [15] across map sizes ranging from 4×4 to 8×8 . For each map size, we create a set of 300 contexts with different hole positions (as in Fig. 1), which we equally split into training, evaluation, and testing sets. By maintaining fixed contexts across splits, we ensure that no RL policy has access to complete information about contexts outside its training set. Each configuration is evaluated over 100 episodes, and results are reported as mean and standard deviation. FrozenLake provides a simple yet informative environment for studying how uncertainty, learned policies, and language-based guidance interact. Varying map sizes and hole layouts enable the controlled induction of distribution shifts, thus reliably stressing learned policies and leading to familiar failure modes. This makes FrozenLake a useful diagnostic environment. Our experiments aim to demonstrate that selectively invoking language guidance based on uncertainty can recover sensible behavior when both the RL policy and the language model fail on their own, a pattern expected to arise in more complex domains.

We adopt Proximal Policy Optimization (PPO) [16] as the reinforcement learning policy (from StableBaselines3 [17]). PPO is a widely used and well-established algorithm in the reinforcement learning literature, providing a strong and stable baseline for sequential decision-making. We use fixed hyperparameters across all experiments and train each policy 2×10^7 timesteps, saving the best-performing model based on the evaluation set (see Appendix B for evaluation results). For LMs, we select models from the Qwen family [18], using readily available weights from HuggingFace for reproducibility. We employ these models off-the-shelf without any task-specific training or fine-tuning. Using a single model family

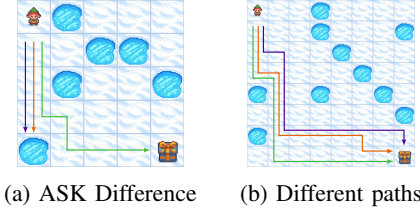


Fig. 1: Different paths for the same context, where **purple** is the LM, **orange** the PPO, and **green** the ASK approach.

allows us to systematically vary model capacity while minimizing architectural and training-related confounding factors, ensuring that observed differences are mainly attributable to scale rather than changes in model design. For the MC Dropout uncertainty estimation, we use $N = 100$ forward passes with a dropout rate of 0.2.

We evaluate each configuration by measuring average accumulated reward $\sum_i \gamma^i r_{i+1}$, where γ is the discount factor. Note that the FrozenLake reward function is sparse: the agent receives a reward of 1 upon reaching the goal and 0 otherwise [15, 19]. To assess efficiency, we also report the average episode length, capped at 100 timesteps. Episodes reaching maximum length with zero reward indicate the agent is stuck in a loop. Although the original 8×8 implementation uses 200 as maximum length, the trained PPO policies achieve the goal on ≈ 12 steps. Thus, we standardize the maximum length to 100 across all map sizes for a fair comparison. Note that generalizing across contexts from maps of different sizes is more challenging than generalizing across maps of the same size with different hole positions. Finally, we aim to assess the computational overhead and behavioral impact of LM integration. To quantify this, we consider two metrics:

Intervention rate (IR): measures the proportion of steps where uncertainty exceeds the threshold τ and the LM is consulted: $IR(ep.) = 1/|ep.| \sum_{t=1}^{|ep.|} 1 [U_e(o_t) + U_a(o_t) \geq \tau]$, where $|ep.|$ is the episode length, o_t is the observation at timestep t , and $1[\cdot]$ is the indicator function. A low intervention rate indicates that the agent operates autonomously, while a high rate suggests frequent uncertainty requiring assistance. The threshold τ was selected via Bayesian optimization with Optuna [20] over $[0.10, 1.20]$ (22 independent studies). Mid-size models (3B–14B) require high thresholds ($\tau > 0.9$) to suppress unreliable interventions, while 1.5B and 72B operate at lower values ($\tau < 0.55$), reflecting better-calibrated guidance. Notably, 72B achieves $\tau = 0.12$ on 7×7 , consulting the LM at nearly every step while maintaining 0.89 reward — a behavior only sustainable at sufficient model scale.

Overwrite rate (OR): measures how often the recommended action differs from the original policy’s suggestion when consulted: $OR(ep.) = \sum_{t=1}^{|ep.|} 1 [U_e(o_t) + U_a(o_t) \geq \tau \wedge a_t^{G_\phi} \neq a_t^{\pi_\theta}] / IR(ep.)$, where $a_t^{G_\phi}$ is the action selected by the LM and $a_t^{\pi_\theta}$ is the

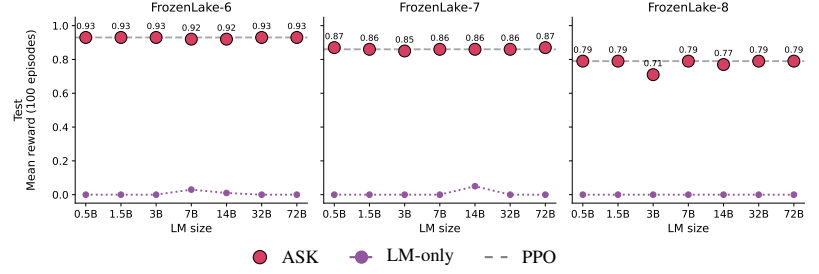


Fig. 2: In-domain performance of PPO, LM, and ASK on the FrozenLake environment across grid sizes for language model sizes ranging from 0.5B to 72B parameters. Values above markers denote mean reward over 100 episodes.

action suggested by the trained policy. This metric reveals whether the LM merely validates the policy’s choices or provides genuinely different guidance. A high overwrite rate indicates that the LM frequently disagrees with the policy in uncertain states, suggesting it offers complementary reasoning rather than redundant confirmation.

Combined, these metrics provide an efficiency-performance trade-off insight. IR quantifies computational overhead, while OR assesses the added value of LM guidance when invoked.

A. In-Domain Evaluation (Same-Size Maps)

Contrary to our expectations, Table Ia shows that integrating LMs does not consistently improve over the PPO baseline. Across all three environment sizes, most LM-augmented models achieve reward values statistically indistinguishable from the PPO baseline. On the 6×6 grid, PPO achieves 0.93 ± 0.26 reward with an episode length of 9.49 ± 1.90 steps. While several models (0.5B, 1.5B, 3B, 32B, 72B) match this reward, they do so without meaningfully reducing episode length, and mid-size models (7B, 14B) show slight performance degradation (0.92 ± 0.27). This pattern persists across larger grids: on 8×8 , where the navigation challenge is most pronounced, only the smallest models (0.5B, 1.5B) and the largest model (72B) maintain the baseline reward of 0.79 ± 0.41 , while 3B shows a substantial drop to 0.71 ± 0.46 .

To understand the role of the uncertainty mechanism, we also evaluated an LM-only baseline where the language model directly controls the agent without PPO guidance. Fig. 2 reveals that LMs alone fundamentally cannot solve these navigation tasks: on test splits, LM-only performance remains near zero across all model sizes and grid dimensions, and on evaluation sets, performance reaches at most 0.60–0.80 on the simplest 6×6 grid, degrading to 0.60–0.70 on 8×8 . This confirms our discussion in the related work (Section V) regarding the limited planning capabilities of language models in sequential decision-making tasks. Despite their strong performance on language understanding and generation, LMs lack the look-ahead search and value estimation mechanisms inherent to RL algorithms. The near-zero validation performance is particularly striking, even with explicit environmental descriptions and step-by-step reasoning prompts, the models fail to navigate effectively in distribution shifts. This underscores the

TABLE I: Comparison of PPO baseline and ASK on FrozenLake: same-size evaluation and downward generalization.

Size	Model	(a) Same-size evaluation				(b) Downward generalization (trained on 8×8)			
		Reward	Length	IR (%)	OR (%)	Reward	Length	IR (%)	OR (%)
4 × 4	PPO	—	—	—	—	0.00 ± 0.00	46.98 ± 49.18	—	—
	0.5B	—	—	—	—	0.00 ± 0.00	41.12 ± 26.20	100.00	61.71
	1.5B	—	—	—	—	0.37 ± 0.49	6.92 ± 6.20	100.00	42.98
	3B	—	—	—	—	0.00 ± 0.00	44.58 ± 26.47	100.00	64.52
	7B	—	—	—	—	0.00 ± 0.00	61.00 ± 0.00	100.00	95.57
	14B	—	—	—	—	0.07 ± 0.26	56.59 ± 15.04	100.00	52.95
	32B	—	—	—	—	0.95 ± 0.22	7.70 ± 9.46	100.00	46.17
	72B	—	—	—	—	0.95 ± 0.22	7.99 ± 9.51	100.00	64.22
5 × 5	PPO	—	—	—	—	0.010 ± 0.099	42.41 ± 48.26	—	—
	0.5B	—	—	—	—	0.00 ± 0.00	31.61 ± 27.21	100.00	99.87
	1.5B	—	—	—	—	0.23 ± 0.42	11.53 ± 13.13	100.00	28.47
	3B	—	—	—	—	0.00 ± 0.00	50.92 ± 22.39	100.00	41.07
	7B	—	—	—	—	0.00 ± 0.00	61.00 ± 0.00	100.00	51.03
	14B	—	—	—	—	0.01 ± 0.10	58.11 ± 12.70	100.00	47.27
	32B	—	—	—	—	0.87 ± 0.34	10.12 ± 10.71	100.00	47.55
	72B	—	—	—	—	0.86 ± 0.35	16.18 ± 18.42	100.00	50.51
6 × 6	PPO	0.93 ± 0.26	9.49 ± 1.90	—	—	0.00 ± 0.00	36.80 ± 46.62	—	—
	0.5B	0.93 ± 0.26	9.49 ± 1.90	49.83	0.00	0.00 ± 0.00	31.61 ± 25.91	100.00	100.00
	1.5B	0.93 ± 0.26	9.49 ± 1.90	55.03	0.00	0.11 ± 0.31	8.60 ± 8.27	100.00	28.96
	3B	0.93 ± 0.26	9.49 ± 1.90	0.00	0.00	0.00 ± 0.00	50.38 ± 22.78	100.00	41.43
	7B	0.92 ± 0.27	11.66 ± 12.95	27.19	4.09	0.00 ± 0.00	61.00 ± 0.00	100.00	50.31
	14B	0.92 ± 0.27	9.55 ± 2.13	0.97	0.47	0.01 ± 0.10	60.31 ± 5.17	100.00	51.50
	32B	0.93 ± 0.26	9.49 ± 1.90	32.03	0.00	0.69 ± 0.46	22.92 ± 21.88	100.00	52.35
	72B	0.93 ± 0.26	9.50 ± 1.86	24.03	0.60	0.75 ± 0.44	21.58 ± 21.09	100.00	50.33
7 × 7	PPO	0.86 ± 0.35	13.08 ± 12.58	—	—	0.00 ± 0.00	30.22 ± 43.76	—	—
	0.5B	0.87 ± 0.34	11.45 ± 2.46	58.23	0.26	0.00 ± 0.00	25.28 ± 24.98	100.00	99.95
	1.5B	0.86 ± 0.35	13.08 ± 12.64	41.59	0.00	0.10 ± 0.30	8.08 ± 7.06	100.00	29.14
	3B	0.85 ± 0.36	13.03 ± 12.65	0.56	0.14	0.00 ± 0.00	53.51 ± 19.53	100.00	44.23
	7B	0.86 ± 0.35	14.25 ± 15.27	9.31	2.53	0.00 ± 0.00	61.00 ± 0.00	100.00	51.57
	14B	0.86 ± 0.35	13.08 ± 12.64	0.00	0.00	0.00 ± 0.00	61.00 ± 0.00	100.00	50.51
	32B	0.86 ± 0.35	13.08 ± 12.64	56.62	0.00	0.58 ± 0.50	30.20 ± 23.38	100.00	53.36
	72B	0.87 ± 0.34	12.22 ± 9.09	58.32	0.32	0.68 ± 0.47	27.10 ± 21.94	100.00	50.88
8 × 8	PPO	0.79 ± 0.41	12.49 ± 3.17	—	—	—	—	—	—
	0.5B	0.79 ± 0.41	12.52 ± 3.21	59.18	0.13	—	—	—	—
	1.5B	0.79 ± 0.41	12.49 ± 3.19	36.61	0.00	—	—	—	—
	3B	0.71 ± 0.46	18.14 ± 21.14	14.74	6.63	—	—	—	—
	7B	0.79 ± 0.41	15.49 ± 13.93	14.99	2.88	—	—	—	—
	14B	0.77 ± 0.42	15.10 ± 15.10	4.32	3.20	—	—	—	—
	32B	0.79 ± 0.41	12.49 ± 3.19	43.01	0.00	—	—	—	—
	72B	0.79 ± 0.41	12.50 ± 3.17	28.60	0.54	—	—	—	—

necessity of the hybrid architecture: while LMs do not improve upon PPO in the same-size scenarios, the gating mechanism is essential to prevent the catastrophic performance degradation that would result from naive LM control.

The failure to improve performance appears closely related to the LM overwrite behavior. Models that maintain baseline performance exhibit near-zero OR ($\leq 0.6\%$), suggesting they primarily defer to the PPO policy. Conversely, models with higher OR, particularly the 3B model with 6.63% overwrites on 8×8, show marked performance degradation. The episode length metric further corroborates this: when 3B overwrites PPO’s actions, episodes increase to 18.14 ± 21.14 steps, compared with the baseline’s 12.49 ± 3.17 , indicating that LM interventions often lead to suboptimal trajectories.

An interesting pattern emerges when comparing the smallest and largest models. The 0.5B model demonstrates high LM engagement (49.83% to 59.18% calls across grid sizes) paired with minimal overwrites (0.00% to 0.26%), suggesting that its uncertainty estimates align well with PPO’s competence—it queries frequently but rarely disagrees. In contrast, the 72B model exhibits comparable performance with more moderate call rates (24.03% to 58.32%) but slightly higher OR (0.32% to 0.60%). This indicates that while 72B is more selective about when to consult the LM, it is also more willing to override PPO when it does, though still with sufficient accuracy to maintain baseline performance.

Most surprisingly, mid-size models (3B, 7B, 14B) exhibit a failure mode characterized by low call rates coupled with high

overwrite percentages. For instance, 14B on 8×8 calls the LM only 4.32% of the time but overwrites 3.20% of those calls, suggesting poor calibration between uncertainty estimation and actual LM reliability. This U-shaped performance curve, where very small and very large models succeed while mid-size models struggle, was unexpected and suggests that the gating effectiveness may be size dependent. These results led us to question whether the same-size evaluation was sufficiently challenging to reveal the potential benefits of LM guidance. In-domain scenarios may be too simple, allowing PPO alone to already find near-optimal policies, leaving little room for LM-based improvement. To investigate whether LMs could provide value in more demanding generalization scenarios, we next examine performance on different-size maps.

B. Downward Generalization (Different-Size Maps)

Having established that same-size generalization provides insufficient challenge, we examine downward generalization where agents trained on 8×8 grids navigate smaller environments (4×4 through 7×7). Table Ib shows that while neither component succeeds in isolation, their combination unlocks robust generalization at scale.

The PPO baseline fails to generalize downward, achieving zero reward as episode length remain closely the same from 46.98 steps on 4×4 to 30.22 steps on 7×7. Combined with our earlier finding that LMs alone achieve near-zero performance (Fig. 2). A sharp capability threshold emerges at 32B parameters. Both 32B and 72B models achieve exceptional performance: 0.95 ± 0.22 reward on 4×4 with

efficient navigation (7.70 and 7.99 steps). This success persists across all sizes: 0.86–0.87 on 5×5 , 0.69–0.75 on 6×6 , and 0.58–0.68 on 7×7 where all other configurations fail completely. Performance degrades with environment size, yet remains functional throughout. This performance transition at 32B reflects a capability threshold for reliable spatial reasoning under distribution shift, rather than a requirement imposed by ASK itself. ASK is model-agnostic and exposes this trade-off by invoking language guidance only when policy uncertainty warrants intervention.

The relationship between model size and generalization exhibits a non-uniform pattern. The 1.5B model shows modest success (0.37 on 4×4 , 0.23 on 5×5) with short episodes. However, mid-size models (3B–14B) fail completely, achieving zero reward despite often reaching maximum episode length. This behavior, in which 1.5B outperforms 3B–14B but falls short of 32B/72B, suggests a capability phase transition.

LM overwrite patterns reveal nuanced dynamics. The 0.5B model’s very high overwrite (61–100%) indicates overconfident interventions that consistently degrade performance. The 1.5B model maintains low overwrite (28–43%), suggesting well-calibrated deference to PPO when uncertain. Successful models (32B/72B) operate with moderate overwrite (46–68%), indicating that the LM actively contributes by overriding PPO approximately half the time, and that these interventions are beneficial. Overwrite rates exceeding 80% indicate failure, where LM overconfidence leads to poor trajectories (Fig. 2).

These results show that PPO alone yields zero reward, and LMs alone fail fundamentally. Yet, their integration at scale (32B+) unlocks robust generalization. This capability threshold suggests that downward generalization requires sophisticated spatial reasoning and goal-directed planning—capabilities that emerge in LMs only at scale, yet require RL’s grounding to translate into effective navigation. Most encouragingly, 32B/72B models maintain functional performance, demonstrating genuinely transferable navigation strategies. The synergy between LM guidance and RL optimization, which is hidden in same-size scenarios due to task simplicity, becomes evident under larger distribution shifts.

V. RELATED WORK

Recent work has explored integrating language models into RL and planning tasks. Several studies demonstrate that LLMs can generate plans when provided with appropriate context [21], yet they face significant limitations in sequential decision-making. Armony et al. [22] show that even state-of-the-art LLMs struggle to match classical symbolic planners in structured domains, particularly when planning requires precise state tracking and multi-step reasoning. Similarly, Valmeekam et al. [23] demonstrate that LLMs often fail to maintain consistency in long-horizon tasks and struggle with domains requiring exact logical reasoning. These limitations motivate our approach: rather than using language models as planners, ASK queries LMs only in uncertain states where reasoning and world knowledge can complement learned policies.

Uncertainty quantification has been proposed as a mechanism for safe RL [10]. Monte Carlo Dropout [14] provides a computationally efficient method to estimate both epistemic and aleatoric uncertainty in neural networks by treating dropout as approximate Bayesian inference. While prior work uses uncertainty to trigger human oversight or conservative actions, we instead use it to gate language model queries, providing automated guidance without human supervision or the computational overhead of full Bayesian inference.

ASK differs from existing LLM-RL integration approaches in three key ways: (i) usage of small to medium language models rather than large ones, making real-time deployment feasible; (ii) selectively invoking the LM based on uncertainty rather than at every step; and (iii) treating LMs as a fallback mechanism for OOD states rather than as the primary decision-maker. This design acknowledges both the planning limitations of language models and the value of learned RL policies, using each where it excels.

VI. CONCLUSION

In this work, we presented Adaptive Safety through Knowledge (ASK), an extrinsic approach that improves OOD generalization in RL by selectively integrating LMs based on uncertainty estimates. Using Monte Carlo Dropout, ASK identifies unfamiliar states and queries an LM only when policy uncertainty exceeds a predefined threshold, preserving efficiency while leveraging language-based commonsense reasoning in critical situations. Empirically, while this integration yields no gains in-domain where PPO already performs well, it enables robust downward transfer: in scenarios where both PPO and language models alone fail, their uncertainty-gated combination achieves up to 0.95 reward, demonstrating a strong synergy that emerges only at sufficient model scale.

More broadly, our results suggest that effective neuro-symbolic systems require principled orchestration rather than naive integration. Instead of replacing specialist policies with generalist models, ASK demonstrates that combining RL for efficient decision-making with language models for reasoning under uncertainty yields more reliable and adaptable agents. Our focus is not to outperform specialized safety or planning heuristics, but to demonstrate a general, domain-agnostic integration principle that augments existing policies without retraining or task-specific engineering. In ASK, language models are not treated as planners, but as semantic critics that selectively override high-uncertainty actions, consistent with the moderate overwrite rates observed in successful configurations. The central question, therefore, is not whether language models can plan or RL agents can generalize, but when and how their complementary strengths should be combined.

Future work includes learning adaptive gating mechanisms beyond fixed thresholds, improving uncertainty calibration through ensembles or Bayesian methods, and exploring joint training of the policy and gate. Extending our approach to more complex and partially observable environments is a natural next step toward understanding the generality and robustness of uncertainty-gated language guidance.

ACKNOWLEDGMENT

This work was partially supported by UK Research and Innovation [grant number EP/S023356/1], in the UKRI Centre for Doctoral Training in Safe and Trusted Artificial Intelligence (www.safeandtrustedai.org), and by the Kunumi Institute, through individual grants awarded to the authors.

REFERENCES

- [1] R. Kirk, A. Zhang, E. Grefenstette, and T. Rocktäschel, “A survey of zero-shot generalisation in deep reinforcement learning,” *Journal of Artificial Intelligence Research*, vol. 76, pp. 201–264, 2023.
- [2] 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.
- [3] G. Dalal, K. Dvořák, M. Vecerik, T. Hester, C. Paduraru, Y. Tassa, T. Schaul, N. Heess, and M. Riedmiller, “Safe exploration in continuous action spaces,” 2018, arXiv:1801.08757.
- [4] M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu, “Safe reinforcement learning via shielding,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.
- [5] A. Mordoch, E. Scala, R. Stern, and B. Juba, “Safe learning of PDDL domains with conditional effects,” in *Proceedings of the 34th International Conference on Automated Planning and Scheduling (ICAPS)*. AAAI Press, 2024, pp. 387–395.
- [6] F. Galuppo, G. L. Zuin, G. Drummond, D. Oliveira, and A. A. Veloso, “Enhancing genetic algorithms for feature selection with language models,” in *Proceedings of the IEEE International Conference on Big Data (BigData)*, 2025.
- [7] R. S.-Y. Chan, F. Nanni, T. Lazauskas, R. Wood, P. Yong, L. Tarassenko, M. Girolami, J. Geddes, and A. Duncan, “Retrieval-augmented reasoning with lean language models,” 2025, arXiv:2508.11386.
- [8] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. Cambridge, MA: MIT Press, 2018.
- [9] A. Der Kiureghian and O. Ditlevsen, “Aleatory or epistemic? Does it matter?” *Structural Safety*, vol. 31, no. 2, pp. 105–112, 2009.
- [10] A. Kendall and Y. Gal, “What uncertainties do we need in Bayesian deep learning for computer vision?” in *Advances in Neural Information Processing Systems 30 (NeurIPS)*, 2017, pp. 5574–5584.
- [11] A. Kirsch, C. Lyle, and Y. Gal, “Unpacking information bottlenecks: unifying information-theoretic objectives in deep learning,” 2020, arXiv:2003.12537.
- [12] J. Postels, H. Blum, C. Cadena, R. Siegwart, L. V. Gool, and F. Tombari, “Quantifying aleatoric and epistemic uncertainty using density estimation in latent space,” 2020, arXiv:2012.03082.
- [13] J. Mukhoti, A. Kirsch, J. van Amersfoort, P. H. S. Torr, and Y. Gal, “Deep deterministic uncertainty: a new simple baseline,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2023, pp. 24 384–24 394.
- [14] Y. Gal and Z. Ghahramani, “Dropout as a Bayesian approximation: representing model uncertainty in deep learning,” in *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, vol. 48. JMLR.org, 2016, pp. 1050–1059.
- [15] A. Kwiatkowski, M. Towers, J. K. Terry, J. U. Balis, G. D. Cola, T. Deleu, M. Goulão, K. Andreas, M. Krimmel, A. KG, R. D. L. Perez-Vicente, A. Pierré, S. V. Schulhoff, J. J. Tai, H. Tan, and O. G. Younis, “Gymnasium: A standard interface for reinforcement learning environments,” <https://openreview.net/forum?id=feFlfuOse1>, 2025.
- [16] Y. Chen, W. Shih, H. Lai, H. Chang, and J. Huang, “Pairs trading strategy optimization using proximal policy optimization algorithms,” in *IEEE International Conference on Big Data and Smart Computing (BigComp)*. IEEE, 2023, pp. 40–47.
- [17] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, “Stable-Baselines3: reliable reinforcement learning implementations,” *Journal of Machine Learning Research*, vol. 22, no. 268, pp. 1–8, 2021.
- [18] Q. Team, “Qwen2.5: A party of foundation models,” <https://qwenlm.github.io/blog/qwen2.5/>, 2024.
- [19] A. Schmitt, “Deontic challenges combined with ASP-planning on RL-training for the FrozenLake and extensions,” Ph.D. dissertation, Technische Universität Wien, 2025.
- [20] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, “Optuna: a next-generation hyperparameter optimization framework,” in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. New York, NY, USA: Association for Computing Machinery, 2019, pp. 2623–2631.
- [21] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch, “Language models as zero-shot planners: extracting actionable knowledge for embodied agents,” in *International Conference on Machine Learning (ICML)*, 2022, pp. 9118–9147.
- [22] M. Armony, A. Meroño-Peñuela, and G. Canal, “How far are LLMs from symbolic planners? An NLP-based perspective,” 2025, arXiv:2508.01300.
- [23] K. Valmeekam, M. Marquez, S. Sreedharan, and S. Kambhampati, “On the planning abilities of large language models: a critical investigation,” in *Advances in Neural Information Processing Systems 36 (NeurIPS)*, 2023, pp. 75 993–76 005.

APPENDIX

ADDITIONAL EXPERIMENTAL DETAILS

A. Prompt

The prompt *Pr* provides the language model with a structured description of the local navigation problem, encoding both the agent state and its surrounding environment. All variables are deterministically extracted from the FrozenLake grid and injected into the prompt during inference. The variables `agent_row` and `agent_col` (Line 17) denote the current row and column of the agent, while `goal_row` and `goal_col` (Line 18) represent the coordinates of the goal state. Together, these variables provide global positional context, allowing the model to reason about relative distance and direction toward the goal. The immediate surroundings of the agent are described by the variables `up_tile`, `down_tile`, `left_tile`, and `right_tile` (Lines 20–23), which encode the tile types directly adjacent to the agent in each cardinal direction. This information enables the model to assess the immediate safety of each possible action, including the presence of holes or map boundaries. To incorporate short-horizon foresight, the prompt additionally includes the look-ahead variables `up_up_tile`, `down_down_tile`, `left_left_tile`, and `right_right_tile` (Lines 25–28), representing the tiles located two steps away in the corresponding directions. These variables allow the model to anticipate unsafe transitions beyond the immediate move and avoid actions that lead to future dead ends. The action space is explicitly constrained to the discrete set {UP, DOWN, LEFT, RIGHT} (Line 30), and the prompt enforces strict output rules requiring the model to return exactly one valid action without additional text or explanations. Edge tiles are treated as unsafe, and when multiple safe actions are available, the model is instructed to select the action that minimizes the distance to the goal. This structured prompt design enables deterministic, safety-aware reasoning by the language model while ensuring full compatibility with the discrete action space and transition dynamics of the reinforcement learning environment.

Listing 1 Prompt for ASK

```

1 You are a robot navigation policy.
2 Your task is to choose exactly ONE action.
3 VALID ACTIONS:
4 UP
5 DOWN
6 LEFT
7 RIGHT
8 RULES:
9 - Do NOT explain.
10 - Do NOT add text.
11 - Do NOT add markdown.
12 - Do NOT add JSON keys other than "action".
13 - EDGE tiles are unsafe and must not be chosen.
14 - If the suggested action is unsafe, choose a safe alternative.
15 - If multiple safe actions exist, choose the one that moves closer to the goal.
16 STATE:
17 Agent position: row {agent_row}, col {agent_col}
18 Goal position: row {goal_row}, col {goal_col}
19 IMMEDIATE NEIGHBORS:
20 UP: {up_tile}
21 DOWN: {down_tile}
22 LEFT: {left_tile}
23 RIGHT: {right_tile}
24 LOOK AHEAD:
25 UP->UP: {up_up_tile}
26 DOWN->DOWN: {down_down_tile}
27 LEFT->LEFT: {left_left_tile}
28 RIGHT->RIGHT: {right_right_tile}
29 OUTPUT FORMAT (MANDATORY):
30 {{"action": "UP"}} OR {{"action": "DOWN"}} OR {{"action": "LEFT"}} OR
    {{"action": "RIGHT"}}

```

B. Evaluation Results

Tab. II displays the result for the same-size map task. Tab. III displays the results for the downwards generalization. We note that the evaluation and the test PPO’s performance remains approximate. Therefore, the RL policy achieves results comparable to those obtained when it was selected during training. Finally, we also note that all RL policies achieve the goal 100% during training, e.g., reward of 1.00 ± 0.00 .

TABLE II: Comparison between PPO baseline and ASK.

Size	Model	Reward	Length	IR (%)	OR (%)
6 × 6	PPO	0.89 ± 0.31	9.22 ± 2.32	–	–
	0.5B	0.89 ± 0.31	9.22 ± 2.32	21.10	0.02
	1.5B	0.89 ± 0.31	9.22 ± 2.32	27.88	0.00
	3B	0.74 ± 0.44	13.79 ± 19.68	25.32	8.32
	7B	0.88 ± 0.32	10.14 ± 7.92	16.59	3.36
	14B	0.75 ± 0.43	25.20 ± 32.32	32.30	17.84
	32B	0.89 ± 0.31	9.22 ± 2.32	17.27	0.00
	72B	0.89 ± 0.32	12.92 ± 17.13	19.37	5.30
7 × 7	PPO	0.86 ± 0.36	12.91 ± 12.68	–	–
	0.5B	0.85 ± 0.36	12.73 ± 11.60	27.81	0.41
	1.5B	0.85 ± 0.36	12.91 ± 12.68	30.60	0.00
	3B	0.68 ± 0.47	23.56 ± 31.05	22.22	9.77
	7B	0.81 ± 0.39	16.47 ± 20.04	21.16	6.13
	14B	0.63 ± 0.48	35.63 ± 38.79	37.26	23.99
	32B	0.85 ± 0.36	12.91 ± 12.68	23.84	0.00
	72B	0.87 ± 0.34	12.99 ± 11.39	26.60	9.39
8 × 8	PPO	0.75 ± 0.36	12.01 ± 12.68	–	–
	0.5B	0.75 ± 0.43	12.02 ± 3.66	24.16	0.05
	1.5B	0.75 ± 0.43	12.01 ± 3.66	19.09	0.00
	3B	0.48 ± 0.50	31.36 ± 36.29	42.33	22.48
	7B	0.65 ± 0.48	22.89 ± 27.77	42.38	10.99
	14B	0.63 ± 0.48	26.52 ± 31.56	29.67	18.41
	32B	0.75 ± 0.43	12.01 ± 3.66	25.07	0.00
	72B	0.72 ± 0.45	19.07 ± 22.77	30.05	11.24

TABLE III: Downward generalization results.

Env	LLM Model	Reward	Length	IR (%)	OR (%)
4 × 4	0.5B	0.00 ± 0.00	51.76 ± 46.85	100.00	57.13
	1.5B	0.47 ± 0.50	7.54 ± 6.46	100.00	41.00
	3B	0.00 ± 0.00	83.40 ± 36.87	100.00	62.20
	7B	0.00 ± 0.00	100.00 ± 0.00	100.00	98.70
	14B	0.03 ± 0.17	95.22 ± 20.95	100.00	55.80
	32B	0.93 ± 0.26	9.68 ± 18.55	100.00	50.66
	72B	0.94 ± 0.24	9.98 ± 18.50	100.00	68.98
5 × 5	0.5B	0.01 ± 0.10	29.21 ± 36.82	86.76	86.75
	1.5B	0.23 ± 0.42	10.10 ± 10.55	100.00	32.43
	3B	0.00 ± 0.00	79.04 ± 40.00	89.64	39.77
	7B	0.00 ± 0.00	92.20 ± 26.59	88.44	46.62
	14B	0.01 ± 0.10	94.73 ± 21.28	87.34	48.86
	32B	0.85 ± 0.36	18.34 ± 28.91	100.00	49.51
	72B	0.85 ± 0.36	22.54 ± 33.10	100.00	51.05
6 × 6	0.5B	0.06 ± 0.24	16.83 ± 29.10	61.68	61.68
	1.5B	0.11 ± 0.31	9.76 ± 9.79	100.00	30.39
	3B	0.00 ± 0.00	80.63 ± 38.97	99.99	41.64
	7B	0.00 ± 0.00	38.16 ± 46.70	9.94	4.90
	14B	0.00 ± 0.00	100.00 ± 0.00	100.00	51.88
	32B	0.70 ± 0.46	31.15 ± 37.31	100.00	53.56
	72B	0.74 ± 0.44	29.05 ± 35.95	100.00	50.74
7 × 7	0.5B	0.00 ± 0.00	28.31 ± 42.73	0.00	0.00
	1.5B	0.08 ± 0.27	8.18 ± 7.67	100.00	28.84
	3B	0.00 ± 0.00	28.31 ± 42.73	0.00	0.00
	7B	0.00 ± 0.00	28.31 ± 42.73	0.00	0.00
	14B	0.00 ± 0.00	28.31 ± 42.73	0.00	0.00
	32B	0.00 ± 0.00	28.31 ± 42.73	0.00	0.00
	72B	0.68 ± 0.47	36.76 ± 38.98	100.00	52.18