ChatGPT

Adversarial Reinforcement Learning for Robust Policy Learning: Literature Review

Executive Summary

Adversarial Reinforcement Learning (ARL) has emerged as a powerful framework for training **robust policies** in complex environments by introducing a two-agent game between a primary agent (protagonist) and an adversary [1]. The adversary's role is to **steer the protagonist into challenging states** that the protagonist would otherwise rarely encounter, thus forcing it to learn behaviors that generalize beyond the standard training conditions [2]. This literature review surveys the state of the art in ARL and related areas, evaluates the novelty of the proposed *Value* + *Uncertainty* adversary reward method, and situates it relative to existing work in robust and curriculum learning for reinforcement learning.

In the proposed approach, the adversary is trained with a **hybrid reward** comprising two terms: (1) a **value-suppression term** (negative of the protagonist's value estimate) to push the agent toward low-value (difficult) states, and (2) an **uncertainty-exploitation term** that uses a Q-ensemble's disagreement (coefficient of variation) to target states where the protagonist's predictions are uncertain. This is a novel combination: prior ARL methods like Laux et al. (2020) used only the protagonist's value function to guide the adversary [1] [2], whereas our method explicitly incorporates **epistemic uncertainty** from an ensemble of critics as an intrinsic objective for the adversary. We also introduce **z-score normalization** of these reward signals and a **dynamic scheduling of weighting coefficients (λ)** over the course of training, ensuring a balanced progression from exploration (uncertainty-driven) to exploitation (value-driven).

Main findings: Existing literature provides strong foundations in each component of our approach, but no single prior work combines them in the same way. ARL frameworks (e.g. Pinto et al., 2017; Laux et al., 2020) demonstrate the effectiveness of adversarial agents in improving robustness [3] [4]. Robust RL research highlights worst-case optimization and domain randomization as tools for generalization [5] [6]. Ensemble-based uncertainty estimation is a well-established technique for exploration [7] and for stabilizing value learning [8]. Curiosity and intrinsic motivation studies show the benefit of **encouraging agents to seek novel or uncertain states** [7] [9]. Curriculum learning and self-play methods illustrate how a gradual increase in task difficulty can accelerate learning [10] [11]. Our approach can be seen as a synthesis of these ideas: an *adversary-driven curriculum* that not only presents the protagonist with difficult states (low value) but also systematically probes the **unknown** (high uncertainty) regions of the state space.

Novelty assessment: Compared to Laux et al. (2020) – which is the baseline for our work – our method introduces **uncertainty-guided adversarial training** on top of their value-based ARL framework [4] [12]. This extension addresses a key limitation of pure value-based adversaries: the tendency to focus only on states that the current protagonist already finds hard (low value), possibly missing states that the agent *thinks* are easy (high value) but in reality has never experienced (thus might be falsely optimistic). By rewarding the adversary for finding states with high Q-value disagreement among an ensemble of protagonist critics, we actively direct it to expose the protagonist's blind spots. This concept of an adversary *exploiting the protagonist's uncertainty* has not been explicitly implemented in earlier ARL works, aside from a few precursors in robust RL that we detail below (e.g. an adversary

encouraging high variance outcomes [13] [12]). Furthermore, the use of on-the-fly **z-score normalization** for the reward components and a time-varying schedule for their weights is novel in adversarial training contexts, ensuring stable learning despite non-stationary reward scales. These contributions position our work as a **natural extension** of Laux et al.'s ARL framework – effectively a next-generation ARL with uncertainty-aware curriculum.

In summary, the proposed approach aligns with a growing trend of combining **robust adversarial training** with **uncertainty-driven exploration** to produce policies that succeed under both expected and unexpected conditions. The following sections provide a detailed literature review, categorized by relevant subfields, a comparison of key related works (including a feature-by-feature table of 10 seminal methods), an analysis of novelty, and a discussion of how our work fills specific gaps in current research.

1. Adversarial Reinforcement Learning (ARL)

Definition & Framework: Adversarial RL refers to two-player reinforcement learning setups, often formulated as a **zero-sum Markov game**, in which a protagonist (main agent) and an adversary have opposing goals [14] [15]. Typically, the protagonist tries to maximize the task reward, while the adversary tries to minimize it (or explicitly cause the protagonist to fail). This leads to a minimax optimization: the solution is a policy pair at Nash equilibrium where the protagonist is robust against the worst-case adversary actions [14] [15]. Training is often done in an **alternating two-phase** procedure: fix protagonist, train adversary to find hard scenarios; then fix adversary, train protagonist on those scenarios [16]. Our two-stage training algorithm (adversary phase, then protagonist phase per iteration) directly follows this template.

RARL (Pinto et al., 2017): A landmark early work in ARL is **Robust Adversarial RL (RARL)** [3]. In RARL, the adversary was not an initial-state generator but rather a destabilizing force applied during the protagonist's episodes [3] [17]. Pinto et al. modeled the adversary as an agent that injects perturbation forces into the environment (e.g. pushing a robot or altering physics parameters) while the protagonist tries to accomplish its control task [3] [17]. The adversary's reward was effectively the *negative* of the protagonist's reward (a zero-sum setup) – thus, it learns to maximize the protagonist's cost or minimize its performance. This two-agent training yielded significantly more **robust policies** that could handle environmental variations and even perform better in nominal conditions without the adversary present [18] [19]. RARL showed that an **adaptive adversary** is more effective than a fixed noise or random disturbance: policies trained with an adversary outperformed those trained with mere domain randomization of forces, particularly in scenarios with test conditions far from the training distribution [20] [21]. In essence, the adversary *learns* to concentrate on worst-case perturbations, offering a concentrated curriculum of "stress tests" for the protagonist. Our approach shares RARL's philosophy of training on worst-case scenarios, but instead of physical force perturbations, our adversary **manipulates the environment state** (specifically, placing the agent at challenging start states in a maze). This is akin to adversarial environment design rather than direct action perturbation.

Laux et al. (2020) – Adversarial Initial State Generation: Melvin Laux and colleagues introduced an ARL framework that is directly analogous to our setup [1] [2]. In their method, the adversary's policy generates **challenging initial states** for the protagonist by physically moving in the environment for a certain number of steps; then the protagonist must solve the task from that new state [2]. They used Soft Actor-Critic (SAC) for both agents, training them alternately. Crucially, the adversary's reward was defined as the *negative protagonist value function* at the state where the adversary leaves the protagonist [2] [22]. In other words, if the protagonist's current value network $V_P(s')$ assigns a low value to state s' (meaning it expects low future reward or difficulty from that state), the adversary gets a high reward for bringing about s'. This **value-based adversary reward** is the exact baseline

that we also implement (we call it our “Value-Only” baseline). Formally, Laux et al.’s adversary reward can be written $r_A = -V_P(s')$ [22], which in our baseline corresponds to $-\gamma_v \hat{V}_P(s')$ with some scaling factor. The intuition is that the adversary acts as a **curriculum generator**, continually updating the initial state distribution $\mu_P(s)$ for the protagonist to include harder and harder states as the protagonist improves [2]. Laux et al. demonstrated this approach on a **continuous maze navigation task** (as well as a robotics task of disentangling objects) and found that the ARL-trained protagonist could generalize better to unseen start states compared to a non-adversarial training baseline [4] [23]. For our work, Laux et al. (2020) serves as the **primary baseline and inspiration**: our two-phase training loop and value-based adversary reward component mirror theirs. The key differences – as summarized later in Table 1 – are our introduction of an uncertainty-based reward term, reward normalization, and dynamic weighting. These additions aim to overcome limitations of the pure value-based approach (e.g. potential myopia or getting stuck in local “difficult” regions).

Zero-Sum Training and Safety: Other works in adversarial RL focus on safety and worst-case guarantees. **Gleave et al. (2020)**, for example, studied adversarial policies in multi-agent settings, where an adversary learns to produce *natural* perturbations that fool a victim agent without breaking environment physics. While not directly a curriculum method, it underscores the need to consider **intelligent adversaries as a threat** and, conversely, as a training tool to preempt such threats. There are also connections to **robust control**: the minimax nature of ARL echoes H-infinity control theory (as noted by Pinto et al. [24]), where the controller is optimized for worst-case disturbances. However, deep ARL allows learning such controllers from data.

Domain Randomization vs Adversarial Training: A recurring question in this field is the benefit of an *adaptive adversary* vs. random perturbations (**domain randomization**). Domain randomization (DR) involves training the agent on a wide distribution of environment variations (e.g. random initial states, random physics parameters) so that it generalizes to any instance in that distribution [25] [26]. It is a strong baseline for sim-to-real transfer (for example, randomizing lighting, textures, or physical properties has enabled robots to transfer policies from simulation to real world). The advantage of DR is that it’s simple and does not require training another agent. However, DR may **waste time on easy or uninformative variations** and doesn’t necessarily focus on the worst cases. Adversarial training, by contrast, **concentrates experience on the hardest conditions** that the adversary can find. Pinto et al. (2017) explicitly compared RARL against a domain randomization baseline and found that the adversarial policy produced more robust protagonists, especially in scenarios that were rare under the random distribution [3] [21]. A NeurIPS 2020 review of PAIRED (discussed later) also notes that minimax (adversarial) environment design tends to yield a curriculum of increasingly complex challenges, whereas domain randomization provides a broader but less targeted training set [25] [26]. In our case, the adversary learns to *adapt* to the protagonist’s skill, continually upping the difficulty – something a static random distribution cannot do. **However, adversarial training must be used carefully**: an unconstrained adversary might produce extremely out-of-distribution or even impossible states that the protagonist can never learn to handle. To prevent this, prior works assume the environment has some structure (e.g., **irreducibility or solvability** of states [27]). Methods like PAIRED introduce a second “antagonist” agent to ensure tasks are solvable (more in Section 5). In our framework, we rely on the environment’s inherent constraints (the maze and physics) and a penalty for invalid states to keep the adversary’s behavior reasonable.

In summary, ARL provides a mechanism for **automatic curriculum learning through competition**: the adversary finds the weaknesses of the protagonist, and the protagonist improves to overcome them. This cycle ideally converges to a robust policy that can handle a wide range of situations, including worst-case scenarios engineered by an intelligent adversary.

2. Robust RL and Policy Robustness

Robust reinforcement learning focuses on ensuring that the learned policy performs well under **perturbations, model uncertainties, or worst-case conditions**. It is closely related to ARL, but not all robust RL methods use an explicit adversary; some adopt algorithmic approaches to directly optimize worst-case outcomes or reduce sensitivity to uncertainty.

Worst-Case Optimization: A common formulation is the **robust MDP**, where instead of maximizing expected reward under a nominal model, the agent maximizes reward under the *worst* model from an uncertainty set. This leads to **minimax optimization**: $\max_\pi \min_{\Delta \in \mathcal{U}} \mathbb{E}_{P+\Delta}[R]$, where Δ represents perturbations in dynamics or observations. Solving this analytically is hard, but methods exist to approximate it. RARL (above) effectively does this by sampling worst-case disturbances via an adversary. Another approach is **Maximin Q-learning**, which extends the double Q-learning idea to *N* critics, taking the minimum value across an ensemble to form a conservative estimate [28] [29]. By updating the policy against this minimum Q, the agent favors actions that are good across all (or most) models, thus hedging against model errors. *Clipped Double Q-learning* (as used in SAC and TD3) is a special case (N=2, take min) that was originally introduced to reduce overestimation bias, but it also has a robustness interpretation: it guards against the optimistic Q due to function approximation errors [30].

Domain Randomization and Distributional Robustness: As discussed, domain randomization is a pragmatic approach to robust RL. Instead of an adversary, the experimenter defines a distribution over environment parameters (dynamics, initial states, etc.) and trains the agent across all of them. The **EPOpt algorithm** (Rajeswaran et al., 2017) improved on plain randomization by biasing training towards harder instances: at each iteration, it samples a batch of environments and then **selects the fraction with lowest returns (the worst) for gradient updates** [6]. Essentially, EPOpt performs *adversarial filtering* on a random set of environments, focusing learning on challenging scenarios. They achieved robust policies for robotic locomotion that could handle variations in body mass, friction, etc., by optimizing a percentile of return rather than the mean [6]. EPOpt can be seen as implicitly training against an adversary that picks the worst-case domain from the ensemble of simulated domains, though it's not a learned adversary but rather a procedure.

Another line is **distributionally robust RL**, where one optimizes performance under an ambiguity set of transition probabilities or reward functions (often using theories from robust optimization or Wasserstein distance-based uncertainty sets). These approaches ensure a policy that does reasonably well even if the environment is slightly different from the training MDP. They are theoretically appealing but can be conservative (resulting in lower rewards in the nominal case due to hedging against worst-case).

Perturbation-based Robustness: A variety of methods explicitly train the agent to resist specific perturbations. For example, adding **adversarial noise to observations or actions** during training can make a policy more insensitive to sensor noise or adversarial attacks. Volodymyr et al. (2016) added perturbations to policies to improve robustness. **Policy distillation with noise** and **data augmentation** (e.g. random shifts in images) have also been used to make policies invariant to certain changes. These are less principled than ARL but have been effective in practice, especially in high-dimensional observation spaces like vision.

Sim-to-Real Transfer: Robustness is often evaluated in the context of sim-to-real: a policy is trained in simulation and then expected to work on a real robot. Techniques to improve transfer include domain randomization (as mentioned), **simulator calibration** (randomizing and then narrowing the domain

with real data), and **adversarial simulation** (where the simulator itself is adjusted adversarially to find failures). Pinto et al. noted that a robust policy should handle the "reality gap" by treating discrepancies as disturbances [31] [24]. Indeed, their RARL agents, trained with perturbations, transferred better to slightly different test conditions than standard agents [19]. More recent work like **OpenAI's Automatic Domain Randomization (ADR)** (2019) progressively increases the range of random variations once the agent masters the current range, effectively learning a curriculum of randomized environments. ADR was crucial in solving Rubik's Cube manipulation on a real robotic hand by training entirely in simulation. While ADR is not an adversary in the RL sense, it embodies the idea of **curriculum in domain randomization**, which parallels adversarial training's curriculum by an agent.

Risk-Sensitive and Worst-Case Criteria: Some robust RL methods alter the optimization criterion. Instead of maximizing expected reward, the agent might maximize **Expected Utility** (to encode risk aversion) or use a **CVaR (Conditional Value at Risk)** metric to focus on tail outcomes. These methods are relevant if one interprets robustness as avoiding catastrophic failures. They often require estimating not just a mean return but a distribution of returns (connecting to distributional RL). For instance, **Morimoto & Doya (2005)** introduced risk-sensitive RL using exponential utility; more recently, **Tamar et al. (2015)** incorporated CVaR in the Bellman update to handle rare disastrous outcomes. While not explicitly adversarial, these approaches could complement adversarial training by adjusting how protagonist performance is measured (e.g., our protagonist could be trained to maximize a lower percentile of returns to explicitly account for adversary moves).

Connections to Our Work: Our method's goal – robustifying the protagonist's policy against difficult starting states – squarely lies in robust RL. Instead of hand-designing a distribution of initial states or perturbations, we *learn* an adversary to generate them. This combines the strengths of domain randomization (diversity of scenarios) with targeted worst-case focus (through adversary adaptation). One distinguishing aspect of our approach in the robust RL landscape is the use of **model uncertainty (Q-ensemble variance)** as part of the adversary's objective. This means the adversary is explicitly searching for states where the protagonist's **value predictions are unreliable**. In robust control terms, it's as if the adversary has a sense of the protagonist model's confidence and pushes against its weak points – a capability absent in standard robust RL formulations which don't model the agent's uncertainty. A related prior work is by Pan et al. (2019), who proposed **Risk-Averse Robust RL (RARARL)** for autonomous driving. They defined risk as the *variance of the value function* and used an ensemble of Q-networks to estimate this variance [13] [12]. They trained a risk-averse protagonist (avoiding high-variance actions) and a risk-seeking adversary. The adversary in their case actively **explored parameter space variations that induce high variance outcomes**, thereby creating appropriate challenges for the protagonist [32] [33]. This is conceptually very close to our uncertainty-exploitation term. The difference is that Pan et al. focused on variance in returns (to avoid catastrophes for the protagonist), whereas our method uses variance/disagreement to guide an adversary in state-space. Both approaches leverage ensemble disagreement as a signal of "this situation is not well understood by the agent." Pan's work can be seen as a precursor that validates the idea of using Q-ensemble variance in adversarial training; our contribution is to integrate that idea into the ARL initial-state framework and augment it with normalization and scheduling.

In conclusion, robust RL provides the motivation for *why* adversarial training is useful (to handle worst cases and uncertainties). Our approach inherits from robust RL the mindset of probing worst-case scenarios and the techniques of ensemble-based uncertainty estimation to gauge risk.

3. Uncertainty Quantification in RL

Understanding and quantifying uncertainty is critical for exploration and robust decision-making in reinforcement learning. There are two key types of uncertainty: **epistemic uncertainty** (uncertainty about the model or Q-function due to limited data – reducible by more exploration) and **aleatoric uncertainty** (intrinsic randomness in the environment – irreducible). Our focus is on epistemic uncertainty: the adversary exploits situations where the protagonist is unsure what will happen.

Ensemble Methods for Uncertainty: One practical way to estimate epistemic uncertainty in deep RL is through **ensembles of function approximators**. Pioneering work by Osband et al. (2016) introduced **Bootstrapped DQN**, which maintains multiple Q-network "heads" trained on different bootstrap samples of experience [30] [29]. At each state, the ensemble produces a distribution of Q-value estimates. Disagreement among these estimates signals high epistemic uncertainty (because if the agent had plenty of data in that region, all networks would likely agree). Osband showed that by **randomly picking one head per episode** (Thompson sampling style), an agent can do deep exploration – each head acts like a different hypothesis about the world, and running with one head encourages the agent to visit states that reduce that head's error. This significantly improved exploration in hard games like Montezuma's Revenge. Extensions of this idea include using **Bayesian neural networks** or **Monte Carlo dropout** to get uncertainty, but ensembles have empirically been very effective and simpler to implement.

In continuous control, recent algorithms have leveraged ensembles for both exploration and stability. **SUNRISE (Lee et al., 2021)** is an ensemble framework built on SAC, where multiple Q-networks and policies are trained in parallel [8] [34]. SUNRISE uses the **variance of Q estimations** to re-weight experience: samples that have high prediction uncertainty are given more weight in training (since they likely represent areas where the agent needs to learn more) [8]. This improved sample efficiency by focusing updates on informative transitions. They also added an exploration bonus akin to an Upper Confidence Bound (UCB): adding a bonus reward proportional to the standard deviation of Q estimates for a given state-action [34]. This concept is similar to what our adversary is doing, except in our case it's an external agent using the protagonist's Q variance as *its* reward, rather than the protagonist getting an internal bonus. Nonetheless, the principle is shared: **variance of ensemble predictions is a proxy for "novelty" or "uncertainty"** in the state.

Another relevant method is **Randomized Ensembled Double Q-Learning (REDQ)** by Chen et al. (2021), which uses a large ensemble of Q-networks (e.g. 10) in an off-policy learner [35] [36]. REDQ samples a subset of critics per update to compute targets (taking a minimum to reduce bias) and found that high update-to-data ratios become viable when using an ensemble, dramatically improving sample efficiency on continuous control tasks. While REDQ's aim is faster learning (not exploration per se), it demonstrates that ensemble disagreement can keep targets accurate even under aggressive training. In our case, we maintain 6 Q-heads for the protagonist partly inspired by such results – multiple critics reduce the noise in value estimates and allow computing a meaningful variance. Our *soft value* estimate $\hat{V}_P(s)$ is essentially an expectation over the policy and an average of Q-heads, which should be more reliable than a single critic's estimate.

Epistemic vs Aleatoric: It is important to note that our adversary is specifically exploiting **epistemic uncertainty** – situations where the protagonist's knowledge is lacking. If an environment is stochastic (aleatoric uncertainty), the ensemble would not necessarily disagree systematically (they would all converge to predicting the correct distribution of outcomes if enough data). But if the protagonist simply hasn't experienced a situation enough, the ensemble will disagree and yield a high coefficient of variation (CV). In other words, the adversary is not "confusing" the agent with randomness, but

genuinely identifying states the agent hasn't learned well. This makes the protagonist *learn to reduce epistemic uncertainty*, akin to an exploration-driven agenda set externally.

Disagreement-Based Exploration: A closely related line is intrinsic motivation via model or policy disagreement. Pathak et al. (2019) introduced a self-supervised exploration method where an agent trains an ensemble of forward dynamics models and uses the **disagreement among them as an intrinsic reward** [7]. This causes the agent to take actions that the models disagree on, which typically are actions leading to novel states. Notably, they showed this approach works even without RL – they directly optimized the policy to maximize disagreement (a differentiable objective when using a predictive model) [7]. This greatly accelerated exploration in hard exploration domains and even on a real robot [37]. Our use of Q-ensemble disagreement is analogous: it generates a kind of *intrinsic reward* for the adversary. The difference is one of perspective – in Pathak's work, the protagonist itself seeks novelty (intrinsic reward to itself), whereas in our work, the adversary seeks novelty *for the protagonist*. One might say we have an **external curiosity drive**: the adversary is like a "guide" pushing the protagonist into uncertain states, whereas in typical curiosity the agent pushes itself. This externalization via an adversary may be beneficial because the protagonist can still focus on maximizing extrinsic reward, while the adversary handles the exploration pressure.

Bayesian RL Approaches: In more theoretical terms, Bayesian RL maintains a distribution over models or value functions and chooses actions to both gather information (reduce uncertainty) and gain reward (explore-exploit tradeoff). Solutions like **Posterior Sampling for RL (PSRL)** and **Bayesian DP** are optimal in principle but hard to apply with deep nets. Ensembles can be seen as an approximate Bayesian method (each head corresponds to a sample from the posterior). Our adversary's policy effectively does a form of *Thompson sampling in state space* – it seeks states that are likely to reveal new info about the environment or agent's capabilities.

Uncertainty in Adversarial Context: One novel element in our approach is treating the adversary as the one to leverage the protagonist's uncertainty estimates. There's emerging work on uncertainty-aware adversarial training. For instance, a very recent preprint by Zhang et al. (2025) proposes **UACER (Uncertainty-Aware Critic Ensemble for Robust ARL)**, which integrates an ensemble of critics into robust adversarial training and uses a time-decaying uncertainty bonus for the protagonist's updates [38] [39]. They introduce a *TDU (Time-varying Decay Uncertainty)* aggregation that starts optimistic (including variance to encourage exploration) and gradually becomes average (to focus on exploitation) [40] [41]. The philosophy is similar to our λ scheduling: high weight on uncertainty early on to force exploration, then decay it to zero so eventually only true task reward/value matters. The convergence considerations are beyond our scope here, but it's validating to see concurrent research emphasizing **adaptive uncertainty modulation** in adversarial training. Our method's simpler scheduling (exponential ramp-up for value term; delayed exponential decay for uncertainty term) aligns with the intuition that early in training the protagonist's value estimates are unreliable (so adversary should lean more on uncertainty exploitation), whereas later the protagonist is stronger and its value function is more accurate (so the adversary can focus on pure worst-case value suppression).

In summary, uncertainty quantification via ensembles is a cornerstone technique that we leverage in a new way. It connects to a broad literature where **disagreement signals drive exploration** [7], and it is now finding its way into multi-agent adversarial setups to stabilize and guide training [42] [43]. Our contribution in this aspect is applying ensemble disagreement as a **reward shaping mechanism for an adversary**, which to our knowledge has not been explicitly done before (prior robust adversaries didn't have an uncertainty term in their reward). This could open up a more general principle: adversaries in RL can be equipped with estimators of the protagonist's learning state (like uncertainty, learning progress, etc.) to create an automated curriculum tailored to the protagonist's weaknesses.

4. Intrinsic Motivation and Curiosity-Driven Exploration

Intrinsic motivation in RL refers to augmenting the agent's reward with internal signals that encourage exploration, learning progress, or other behaviors not directly captured by the task's extrinsic reward. This area, while traditionally single-agent, provides concepts and algorithms relevant to our adversarial setting – essentially, we are outsourcing the curiosity to an adversary.

Curiosity as Prediction Error: A classic form of intrinsic reward is **prediction error of a learned model**. If the agent faces a state where its prediction (of the next state, or of some feature) is poor, that indicates novelty, and a curiosity-driven agent receives a positive reward for experiencing that state. The **Intrinsic Curiosity Module (ICM)** by Pathak et al. (2017) used an inverse dynamics model and forward model; the agent was rewarded for high forward-model error, which led it to explore unseen areas in video games. **Burda et al. (2018)** conducted a **large-scale study of curiosity-driven learning** across 54 Atari games [9]. They showed that using the *error of a random feature predictor (Random Network Distillation)* as an intrinsic reward yielded remarkably strong exploration, sometimes rivaling human-designed rewards in terms of enabling progress [44]. They also highlighted pitfalls: e.g., prediction-based bonuses can lead the agent to get stuck in stochastic environments (where error never reduces) [45].

Relevance to our work: The adversary's goal of maximizing the protagonist's prediction uncertainty is analogous to a curiosity agent maximizing prediction error. The difference is the **agency**: Instead of the protagonist being rewarded for its own prediction error, we reward a separate adversary for causing prediction errors in the protagonist. This is a form of **sourced intrinsic motivation**: the source of intrinsic drive (the adversary) is decoupled from the agent experiencing it (the protagonist). Such a scheme can potentially avoid some pitfalls of intrinsic motivation (like getting stuck in stochastic loops), because the adversary, being external, can be constrained or designed to avoid truly unsolvable noise (for example, our adversary can't just inject white noise – it has to pick a physical state in the maze, which the protagonist can learn to handle eventually).

Novelty, Surprise, and Disagreement: Intrinsic rewards can also be based on **state novelty** (e.g., count-based methods). Classic count-based exploration (Bellemare et al. 2016) assigned bonus $\propto 1/\sqrt{N(s)}$ for visiting a state s that has been visited $N(s)$ times before. This encourages covering the state space. In high dimensions, pseudo-counts or hashing techniques are used to approximate state visitation counts. In comparison, ensemble disagreement is like a *learned novelty measure* – if no network in the ensemble has a confident prediction for the state, it's effectively "new". **Disagreement vs. Prediction error:** one nuance is that prediction error is one agent's surprise, while disagreement is a measure of multiple predictors' lack of consensus. Disagreement tends to highlight **uncertainty due to limited data**, whereas raw error could be high even in deterministic but hard-to-learn dynamics. For our adversary, using disagreement (CV of Q estimates) means it targets states where the **agent's knowledge is inconsistent**, which is a focused notion of novelty.

Empowerment and Other Intrinsic Goals: Another intrinsic motivation concept is **empowerment** – maximizing the agent's control over the environment (information-theoretic measure of influence). While not directly related to our methodology, empowerment could be something an adversary unintentionally affects (e.g., adversary might place the protagonist in states where it has *low* empowerment, which are likely difficult states where the protagonist has little control). Some exploration strategies aim to learn skills that reach different regions of the state space (e.g., `Explore` options in hierarchical RL). In effect, our adversary forces the protagonist to broaden its skillset to cope with varied start states.

Intrinsically Motivated Goal Generation: Our approach also resonates with methods where an agent sets its own goals. For instance, **Goal Generation with GANs** (Florensa et al. 2018) trained a GAN to produce goals of appropriate difficulty (not too easy, not impossible) for a goal-conditioned agent. The discriminator's signal indicated if the agent could achieve the goal or not, leading the GAN to generate goals at the boundary of competence. In our case, the adversary could be seen as *generating goals* (end states for the first phase) for the protagonist. The **value term** in its reward ensures goals are not trivial (low protagonist value means it's something the protagonist currently struggles with), and the **uncertainty term** ensures the goals are not ones the protagonist has already mastered implicitly or visited often. This is analogous to the GAN approach where you want goals the agent can just barely not do – here "not do" is captured by low value, and "not seen before" by high uncertainty. We even implemented a form of **automatic curriculum** via the λ schedules, similar in spirit to gradually shifting the goal distribution in curriculum learning.

In summary, the literature on intrinsic motivation provides the *mechanisms* (prediction error, ensemble disagreement, novelty bonuses) that we leverage, but whereas those are typically used for the agent's *own* exploration, we apply them in a two-agent setting. This could be viewed as a hybrid of curiosity and self-play: the adversary is like a curious exploration module embodied as an agent adversarially interacting with the protagonist.

5. Curriculum Learning and Self-Play

Curriculum learning in RL involves presenting tasks in an order of increasing difficulty such that the agent can progressively learn skills and handle more complex scenarios. In our framework, the adversary serves as an automated curriculum generator, *tuning the difficulty* of the protagonist's start state over time. There are multiple strands of research on automatic curricula and self-play that relate to this idea:

Asymmetric Self-Play (ASP): Sukhbaatar et al. (2018) proposed an **asymmetric self-play** training scheme where two instances of the same agent (Alice and Bob) play different roles [10]. Alice's role is to set a task by taking a sequence of actions in the environment (without extrinsic reward), then Bob is spawned in Alice's final state and must *undo* or achieve some goal related to Alice's trajectory [46] [10]. Alice is rewarded if Bob fails, and Bob is rewarded for succeeding, with the constraint that tasks must be solvable (Alice herself could achieve them since it's essentially her trajectory in reverse). This leads to an unsupervised curriculum: Alice will try to devise tasks that are just at the edge of Bob's capability – challenging but achievable – because if she makes an impossible task, Bob gets reward by default (or Alice gets none). Over time, Bob learns to solve very complex sequences since Alice increases difficulty gradually, and these skills transfer to external tasks [47] [10]. This is remarkably similar to our adversary-protagonist setup, with key differences: (1) In ASP, Alice and Bob are the same policy (weights shared and just two "roles"), whereas in ARL, protagonist and adversary have different goals and policies. (2) ASP is cooperative-competitive: Alice isn't strictly adversarial; she is guided to keep tasks achievable. In our case, we did not explicitly force the adversary to keep tasks solvable beyond the inherent environment rules – though practically, if the adversary makes something too impossible, the protagonist will always fail and eventually the value function for those states might go to near zero, which ironically gives the adversary less incentive (since the value term saturates at low values). This acts as a natural regulator: once a state is truly hopeless for the protagonist, V_P will be very low and the adversary doesn't gain as much by going there versus a slightly easier state where V_P is higher but still low. This phenomenon is akin to ASP's requirement to keep the task at the edge of solvability. We could strengthen that by explicitly modeling an antagonist agent (like PAIRED does) or giving adversary negative reward if protagonist fails outright, but our current design implicitly handles it via value normalization and clipping (e.g., we penalize invalid states with an extra -10).

Self-Play in Games: Classic self-play (e.g., in AlphaGo or competitive games) is two agents improving by playing against each other. While typically symmetric (same game, zero-sum), the curriculum emerges as each gets better, the other faces a tougher opponent, and both must adapt. In single-agent curriculum, self-play ideas have been applied by treating past versions of the agent as teachers or competitors. **PLR (Prioritized Level Replay)** by Jiang et al. (2021) for Procgen games stored generated levels and replayed those on which the agent had intermediate performance, to maintain a curriculum of training levels at an appropriate difficulty. This idea of focusing on neither too easy nor too hard aligns with our dynamic λs: early on, we emphasize exploring completely unknown states (even if they lead to failure), later we emphasize value suppression (which might avoid states the protagonist has no hope in and focus on those it's close to overcoming).

Unsupervised Environment Design (UED): A recent formalism by Dennis et al. (2020) (PAIRED) and follow-ups defines the problem of automatically generating environments (tasks) without human labels to produce a curriculum that yields robust, general agents [11]. **PAIRED** (Partner-AI Regret Environment Design) introduced a teacher (environment generator) that aims to maximize the *regret* of the protagonist relative to a second agent (antagonist) that is trained on the same tasks [11]. In other words, the teacher wins if the protagonist performs much worse than an expert antagonist in the environment the teacher created [48]. By doing so, the teacher *implicitly* ensures the environment is solvable (since the antagonist can solve it) but hard for the protagonist. Empirically, PAIRED generated a series of increasingly complex maze levels that the protagonist gradually learned to solve, achieving **zero-shot transfer** to significantly different test mazes at times [49] [50]. PAIRED is a three-agent system and more complex than our two-agent setup, but conceptually it's similar: our adversary could be seen as playing the role of both teacher and antagonist combined (since it both *proposes* a state and ensures it can reach that state by construction). We do not explicitly have an antagonist agent, but one could imagine using the adversary itself as a proxy "can solve this state" since it came from a valid trajectory.

Other Curriculum/Teacher-Student frameworks: There have been many approaches like **Adaptive Level Generation**, **Goal-based curricula**, and so on. For example, **POET (Wang et al., 2019)** evolved a population of environments and agents together, generating an endless curriculum of increasingly difficult obstacle courses [27]. Environments that were too hard or too easy were discarded or tuned. POET showed the benefit of continuously challenging the agent with novel tasks that standard training would not cover. It also emphasized **transfer**: occasionally an agent trained on one environment could solve another's better, so they would transfer it – highlighting how a diversity of challenges can create generally skilled agents. Our work is more constrained (one task family – maze navigation), but the philosophy of unending, varied challenge is inherited from such approaches.

Multi-Task and Training-Generalist Agents: As RL moves towards generally capable agents (Open-Ended Learning, etc.), curricula and adversarial training are key. The **OpenAI Hide-and-Seek** experiments (Baker et al. 2020) famously demonstrated emergent tool use and complex strategies via self-play in a multi-agent environment. Although not framed as curriculum, each time one team found a new strategy, the environment's effective difficulty increased for the other, creating a progression. This again underscores the power of competitive dynamics to foster learning beyond what a static environment would yield.

Our Approach as Curriculum: With the dynamic weighting of the adversary's objectives, we effectively implement a curriculum intentionally. At the start, λ_σ (uncertainty weight) is high (after a warmup) and λ_v (value weight) ramps up gradually. So initially, the adversary prioritizes *novelty* (states that confuse the protagonist). This induces wide exploration. As the protagonist learns and time goes on, we decay the uncertainty bonus, shifting the adversary's focus more to pure value suppression (find the hardest states in terms of low value). This corresponds to a stage where most of the state space has been explored and the primary remaining challenge is truly difficult (but known)

situations. By the end, λ_σ might be minimal, meaning the adversary behaves like Laux et al.'s value adversary, relentlessly probing known weak spots of the protagonist. This progression from exploration to exploitation in adversarial training is reminiscent of **curriculum learning by difficulty**: start broad, then hone in on the tough cases.

In literature, we did not find prior works that *explicitly schedule an adversary's reward terms* in this way, but the need for balancing exploration vs exploitation in adversarial training is recognized [39] [41]. The UED works also faced issues of non-stationarity and aim to stabilize training (e.g., by mixing in some random levels to not overfit the adversary's strategy). Our scheduling is a simpler heuristic solution to ensure the adversary doesn't overly focus on one objective throughout training.

To conclude, curriculum learning research provides both **techniques and validation** for our method: techniques like self-play, goal proposals, and difficulty adjustment inspire how we designed the adversary's behavior, and empirical results from those works (PAIRED, ASP, POET) validate the core idea that *challenging the agent with progressively harder, but achievable tasks leads to more robust skills*. Our adversarial training is essentially an automatic curriculum where the adversary is the "teacher" adjusting difficulty, and our contributions ensure this teacher presents a well-rounded syllabus (covering both low-value and high-uncertainty scenarios).

6. Soft Actor-Critic (SAC) Variants and Architectures

Our implementation builds on **Soft Actor-Critic (SAC)**, an off-policy actor-critic algorithm with maximum entropy reinforcement learning [42]. SAC is known for its stability and state-of-the-art performance in continuous control tasks, making it a sensible choice for both protagonist and adversary agents in our framework. In this section, we briefly survey relevant SAC variants and architectural choices that relate to our approach:

Ensemble Critics in SAC: The standard SAC already employs two Q-functions (critics) and uses the minimum of the two for the target value computation (an implicit ensemble of size 2) [51] [52]. This mitigates positive bias in Q estimates. Several works have extended this idea to larger ensembles. **REDQ**, as mentioned, used 10 critics with SAC to enable a high update-to-data ratio [35]. They found SAC with more critics can make updates more reliable when each gradient step uses an average or min over many Q estimates [53]. Our protagonist uses 6 Q-heads, which is unusual in common SAC implementations but is directly inspired by these findings that *more critics can yield better uncertainty estimates and stability*. The adversary in our case uses 2 Q-heads (like vanilla SAC) since we prioritized the ensemble for the protagonist's uncertainty measurement.

Multi-Head Q-Ensembles vs Multi-Critic Networks: We chose an architecture where one network outputs multiple Q-values (heads) for the ensemble, primarily for computational convenience (like Bootstrapped DQN style). Alternatively, one could have completely separate Q networks. Prior works (Osband 2016) indicate that if heads share too many parameters, their estimates might not be sufficiently independent. We mitigate this by injecting **randomness in initialization** and using different random action samples for each head's target computation. In literature, **Anschel et al. (2017)** introduced **Averaged-DQN**, and **Lan et al. (2020)** proposed **Maxmin Q** which effectively uses an ensemble and either averages or takes minima to reduce bias [28] [30]. They highlight trade-offs between bias and variance: averaging many critics reduces variance in estimates but can introduce optimism, taking a minimum is biased low but safe. Our use of the **CV (coefficient of variation)** cleverly normalizes the standard deviation by the mean magnitude of Q, which is a scale-invariant uncertainty metric – so whether Q values are large positive or large negative, the CV captures relative disagreement.

Distributional SAC: While not explicitly used, it's worth noting that distributional RL methods (like Quantile Regression DQN, or Distributional DDPG) maintain a distribution over returns instead of a single expected value. One could interpret our ensemble as a crude way to get a distribution (each head is like a sample). A true distributional SAC could provide a risk-sensitive perspective (e.g., train protagonist to optimize CVaR). Some works (Barth-Maron 2018, Mazoure 2020) have looked at combining SAC with distributional critics to quantify value uncertainty. This is an area for potential extension – perhaps the adversary could then target states with high variance in the return distribution (which might combine aleatoric and epistemic uncertainty). For now, our ensemble covers the epistemic part.

Actor Ensembles: We briefly mention that one could ensemble the policy (actor) as well. **Lee et al. (2020)** (in SUNRISE) found that an ensemble of actors can help exploration and generalization [54], but also noted it can cause instability due to diverging behaviors [55]. We avoided actor ensembles for protagonist and adversary; instead we focus on critic ensembles. Our adversary's stochastic policy (being Gaussian as in SAC) already explores its action space, and the protagonist's replay buffer gets data from the adversary's varied moves anyway.

Temperature and Entropy Considerations: SAC introduces an entropy temperature α to balance exploration and exploitation. Both our agents use SAC's entropy regularization. One question is how entropy interacts with adversarial training: a high-entropy adversary policy would produce more diverse initial states (exploration in task space), whereas a deterministic greedy adversary would consistently exploit the same type of hardest state. We actually use *stochastic sampling during adversary training phase* and *greedy actions during protagonist training phase* to get a mix: exploration while learning the adversary, and exploitation when deploying the adversary to train the protagonist. This is akin to epsilon-greedy scheduling, but here baked into using SAC (which gradually reduces exploration as Q estimates sharpen if α is tuned or learned). In literature, there isn't a direct analog since most assume the adversary is always present as part of training. Our two-phase separation allowed us to make the adversary behave differently when generating training scenarios vs when it itself is being updated. This design choice was somewhat empirical, but it ensures the protagonist sees a fairly challenging (almost greedy) adversary during its updates, which is important for robustness (you don't want to train protagonist against a "soft" adversary and then face a "hard" adversary at test).

Algorithmic Variants: We should note **TD3, PPO, etc.** have also been used in adversarial setups (e.g., PPO in PAIRED's implementation for stability). SAC's advantage is off-policy efficiency (we can reuse experiences) and handling of continuous action spaces with ease. There have been variants like **SACfD (SAC from Demonstrations)** or **SAC-NoEntropy** that remove entropy regularization if not needed. In our experiments, SAC's stability was good despite the non-stationary environment distribution – possibly the replay buffers and target networks helped smooth things.

In summary, our algorithmic choices align with cutting-edge techniques in deep RL: leveraging *ensembles* for more reliable Q estimates and uncertainty [8], using *entropy regularization* for stable exploration, and structuring the training in an alternating fashion with experience replay, which SAC's off-policy nature facilitates. Our contributions here are not new algorithms for SAC itself, but rather how we **apply SAC in a novel multi-agent adversarial context**. The design of multi-head critics, ensemble-based value estimates, and the interplay of two SAC agents is guided by insights from the works above.

After reviewing the key literature in these categories, we now present a comparative summary of the most relevant works and how they relate to our proposed method. This will be followed by an explicit novelty and gap analysis.

7. Comparative Analysis of Key Related Works

The following table (Table 1) compares our proposed method with **10 closely related works** across several features. These works were chosen as they represent the state-of-the-art or foundational approaches in adversarial RL, robust training, or related exploration techniques:

- Laux et al. (2020) – Adversarial RL framework with value-based adversary (baseline for our work) [1] [2].
- Pinto et al. (2017) – RARL, adversary applies force perturbations (classic robust adversarial training) [3] [17].
- Pan et al. (2019) – Risk Averse RARL, adversary exploits value variance (ensemble-based risk estimation) [13] [12].
- Dennis et al. (2020) – PAIRED, adversarial environment generation with regret minimax (3-agent curriculum) [11].
- Sukhbaatar et al. (2018) – Asymmetric Self-Play, two-agent unsupervised curriculum (tasks via self-play) [10].
- Pathak et al. (2019) – Disagreement Intrinsic Rewards, single-agent explores via model ensemble disagreement [7].
- Burda et al. (2018) – Curiosity (RND), intrinsic reward via prediction error for exploration [9].
- Florensa et al. (2018) – Goal GAN, generates goals of appropriate difficulty (automatic curriculum via GAN).
- Chen et al. (2021) – REDQ, ensemble of Q for efficient SAC (demonstrates value of large ensembles in RL) [35] [56].
- Zhang et al. (2025) – UACER, uncertainty-aware ensemble for adversarial RL (very recent approach aligning with ours) [40] [41].

Table 1: **Comparison of Our Method with Related Works**

Feature	**Our Method (Value + Uncertainty ARL)**	**Laux et al. 2020 (Value-Only ARL)**	**Pinto 2017 (RARL)**	**Pan 2019 (RARARL)**	**Dennis 2020 (PAIRED)**	**Sukhb 2018 (**
Adversary's Role	Places protagonist in hard **states** (maze start positions)	Places protagonist in hard states (maze, disentangling setup)	Injects *forces* on protagonist (perturbs dynamics)	Perturbs environment parameters (e.g., driving sim) over multiple steps	Generates entire environment (layout, dynamics) for a episode	Propo tasks (seque action revers secon
Training Framework	Two SAC agents, alternate phases [16]	Two SAC agents, alternate phases [2]	Two agents (actor-critic), simultaneous or alternating (zero-sum game) [17] [57]	Two agents (DQN or DDPG variants), alternating; risk-sensitive updates	Three agents: Teacher (env gen), Protagonist, Antagonist – minimax-regret training [11]	Two ro (Alice, same self-pl trainir

Feature	Our Method (Value + Uncertainty ARL)	Laux et al. 2020 (Value-Only ARL)	Pinto 2017 (RARL)	Pan 2019 (RARARL)	Dennis 2020 (PAIRED)	Sukhb 2018 (
Adversary/ Generator's Objective	Maximize `r_A = -λ_v * V_P(s') + λ_σ * CV_Q^+(s')` (value suppression + uncertainty exploitation)	Maximize `r_A = -V_P(s')` (or scaled $-\hat{V}_P$) [22]	Maximize protagonist's cost (minimize its reward); effectively $r_A = -r_P$ (zero-sum) [20]	Maximize protagonist's "risk" (variance of return) and minimize its reward – *risk-seeking adversary* [13] [12]	Maximize **regret** = (Protag's loss - Antag's loss) – i.e., make env where protagonist does much worse than an expert [11]	Make that **B** but ar achiev intern reward setting but so tasks
Protagonist/ Agent's Objective	Maximize environment reward (task completion), standard SAC with entropy	Maximize environment reward (task), standard SAC	Maximize task reward (e.g., forward locomotion) while adversary present (minimax) [58]	Risk-averse objective: maximize reward, with penalty for variance (uses ensemble for value estimation) [59] [60]	Maximize task reward on environments generated by teacher (and for antagonist, same but competing)	Bob: maxim intern reward solvin task (r enviro to initi etc.) [4]
Use of Uncertainty	Yes – **Q-ensemble (6 heads)** for V_P and CV; adversary reward uses ensemble variance (epistemic uncertainty)	No – single critic (or double) used directly, no uncertainty bonus	No – single model (various env seeds but no explicit uncertainty est.)	Yes – **Q-ensemble (N heads)** to estimate variance of value [12]; protagonist objective includes variance penalty, adversary exploits variance	Implicitly – regret formulation means teacher tries env where protagonist uncertain enough to fail while antagonist succeeds (a proxy for protagonist's uncertainty)	No e uncert but ta natura progre difficu

Feature	Our Method (Value + Uncertainty ARL)	Laux et al. 2020 (Value-Only ARL)	Pinto 2017 (RARL)	Pan 2019 (RARARL)	Dennis 2020 (PAIRED)	Sukhb 2018 (
Normalization of Reward Signal	Yes – running **z-score** normalization for value estimates and CV (to keep rewards in stable range)	No – raw value function used directly (could be unbounded)	No – adversary reward = negative protagonist reward (bounded by task spec)	Possibly scaled variance, but not explicitly z-normalized (not mentioned)	Regret is difference of rewards (bounded by score differences); no explicit normalization reported	No – intern reward binary succes failure orient
Dynamic Curriculum or Scheduling	Yes – **λ_v and λ_σ are scheduled** (λ_v ramp-up, λ_σ warmup then decay) to adjust focus over training	No – static objective throughout training	No – adversary constantly maximizes protagonist cost; no curriculum except what emerges from training dynamics	Partial – risk aversion may be increased gradually; not a central point though	Partial – the teacher automatically creates curriculum by learning (no explicit time schedule, but an emergent curriculum) [49]	Yes - get ha Bob in (Alice implic follow capab Alice g reward when succee some fails s [10]
Environment/ Test Domain	Continuous 2D maze navigation (custom Gymnasium), multi-maze evaluation (robustness across layouts)	Similar continuous maze + a robot object disentangling task (IROS'20) [61]	MuJoCo locomotion (HalfCheetah, Ant, etc.) and classic control (InvertedPendulum) [21] [62]	TORCS driving simulator (autonomous driving scenarios with different dynamics) [63] [64]	Partially observed 2D mazes (procedural), and a continuous control car racing task [65] [66]	Gridw (Maze tasks, some contin tasks (Moun StarCr task) f evalua [67] [6

Feature	Our Method (Value + Uncertainty ARL)	Laux et al. 2020 (Value-Only ARL)	Pinto 2017 (RARL)	Pan 2019 (RARARL)	Dennis 2020 (PAIRED)	Sukhb 2018 (
Results/ Findings	– Improved protagonist success rate and robustness to unseen start states (expected, since it trains on worst-cases plus uncertainties). *[Our results TBD in thesis]*.	– ARL protagonist outperformed non-ARL in generalizing to different initial states [23] [4]. Adversary automatically learned to set up challenging states (both sim and real robot) [1].	– Robust policies that could handle perturbations; outperformed domain randomization; adversary's presence improved training stability too [18] [21].	– Protagonist learned safer driving policies; adversary uncovered edge cases (like slick patches) more efficiently than random perturbations. Ensemble-based variance helped quantify risk.	– Protagonist achieved higher **zero-shot transfer** performance on unseen test envs compared to domain randomization [49]. Emergent curriculum of increasing complexity in levels [69].	– Ager traine ASP le much on downs tasks t those (due t having maste enviro dynam [46] [1] Showe unsup self-pl replac of task specif trainir

Table Discussion: From the table, we observe that our method is unique in combining several aspects: a two-agent adversarial curriculum (like Laux, Pinto, PAIRED) with explicit **ensemble uncertainty utilization** (like Pathak, Pan, SUNRISE). Most previous ARL works either use value-based objectives (Laux) or directly protagonist reward (RARL) for adversary. Only Pan et al. introduced a notion of variance into adversary training, but their focus was risk-aversion for safety, whereas ours is exploration and robustness. PAIRED and ASP ensure tasks are feasible by design (antagonist or reversibility) – our method currently assumes environment constraints to maintain feasibility, which is an area we note for caution (though as discussed, the value normalization does discourage truly impossible states because they become low-value outliers that contribute less to normalized reward).

No prior approach in the table except UACER (2025) has a **dynamic scheduling of objectives** like our λ schedules. UACER's TDU is conceptually similar in decaying uncertainty bonus over time [40] [41]. This suggests our idea of scheduling is well-founded and likely necessary for practical training convergence in adversarial setups.

Finally, the comparison with curiosity-driven methods highlights that while those can enhance exploration, they don't directly tackle *robustness* to environment changes; combining them with an adversarial framework (as we do) yields a more targeted robustness training. Conversely, pure adversarial methods lacked an exploration drive which our uncertainty term supplies.

8. Novelty Assessment

Having surveyed related work, we can now pinpoint which aspects of our approach are **novel contributions** to the literature and which are building on (or coincident with) existing ideas:

- **Value + Uncertainty Adversary Reward:** The primary novelty is the **integration of ensemble uncertainty into the adversary's reward function** alongside the value-based term. While value-based adversarial rewards were introduced by Laux et al. (2020) [22], and ensemble variance has been used for exploration (Pathak 2019 [7]) or risk (Pan 2019 [12]), *no prior adversarial RL method has explicitly combined these two signals in the adversary's objective*. This creates a more powerful adversary that not only exploits known weaknesses (low-value states) but also probes unknown areas (high-variance states). In effect, our adversary **embeds a curiosity-driven exploration bonus** into a two-player training regime – a novel cross-pollination of intrinsic motivation and competitive self-play.

- **Z-Score Normalization for Adversary Rewards:** We apply statistical normalization (running mean and variance) to the adversary's observed value estimates and uncertainty measures, using a *running z-score* to compute z_V and z_σ^+ in the reward【Prompt§2.3.1】【Prompt§2.3.2】. This is somewhat **underrepresented in literature**: intrinsic rewards in single-agent exploration are sometimes normalized for stability (e.g., RND normalize features), but adversarial rewards are usually hand-designed without adaptive normalization. Our normalization ensures the adversary's reward components remain comparably scaled over time (preventing one term from dominating simply due to scale changes). This is a practical innovation that makes long training runs more stable and is especially helpful when combining heterogeneous signals (value which can drift as policies change, and uncertainty which can spike early on). We did not find previous ARL papers mentioning this technique, so it appears to be an original contribution in implementation.

- **Dynamic Weight Scheduling (Curriculum within ARL):** We introduced time-dependent weights $\lambda_v(t)$ and $\lambda_\sigma(t)$【Prompt§2.3.3】. This **explicit curriculum mechanism for the adversary's objective** is novel. It acknowledges that at different stages of training, the adversary should behave differently: initially encourage broad exploration (uncertainty bonus high), later focus on fine-tuning weaknesses (uncertainty bonus low, value term dominant). While human curricula often do this conceptually, encoding it into the objective function of an adversary is new. One might say PAIRED's teacher implicitly achieves a similar effect by natural learning dynamics, but we make it an explicit schedule. The only analogous work, as noted, is the very recent UACER (2025) which independently came up with a decaying uncertainty coefficient for robust RL [40] [41]. Our scheduling is simpler (piecewise constants and exponentials) whereas theirs is derived from a theoretical perspective, but the convergence of ideas in 2024-2025 underscores its novelty and importance. In summary, when comparing to Laux et al., our *static vs dynamic* weighting is a clear innovation that should yield a more effective training progression.

- **Protagonist with Large Q-Ensemble (6 heads) in ARL:** Using an *ensemble of 6 critics* for the protagonist is not conceptually novel (ensemble methods abound), but in context, earlier ARL works (Laux, Pinto) used 1 or 2 critics. Pan (2019) used an ensemble (not sure how many, possibly 5 as per some references, but definitely more than 2) purely to estimate variance [12]. We push this idea by integrating it fully into the actor-critic (SAC) training of the protagonist. Also, we don't just use the ensemble for the adversary's sake; the protagonist itself trains on all those critics (perhaps taking an average for the actor loss, etc.), which could improve its value

estimation robustness. The novelty is incremental here – combining known techniques (REDQ style critics) in a new context (two-agent adversarial training).

- **Application to Continuous Maze Navigation with Adaptive Difficulty:** On the experimental side, our work appears to be the first to apply adversarial training with an uncertainty-based objective to a continuous **maze navigation problem** with varying complexity levels. Laux et al. did test a continuous maze [2], but without uncertainty guidance. PAIRED and others did maze tasks but in a more abstract/discrete manner or fully observed settings. Our environment is custom and specifically designed to test robust policy learning across multiple maze layouts (including out-of-distribution ones). While environment novelty is not a theoretical contribution, demonstrating ARL in this domain fills a gap: most ARL prior works focused on robotics control or synthetic games, and environment-generation approaches like PAIRED were partly validated on maze tasks. Our results will directly compare to Laux's maze experiment, establishing a new baseline on that domain with our enhancements.

- **Positioning as Extension of Laux et al.:** Considering novelty in the context of incremental research, our method can be seen as an **extension/improvement of Laux et al. (2020)**. We take their framework and address its limitations (lack of exploration incentive and scaling issues of value-only reward). Thus, the novelty is not in inventing ARL from scratch, but in **proposing significant enhancements**: adding an exploration-driven component (novel in ARL), normalizing and scheduling (novel in ARL), and demonstrating improved outcomes. In academic terms, this is a meaningful contribution if we empirically show these additions yield better robustness or learning efficiency.

- **Comparison to Very Recent Work:** UACER (Zhang 2025) shares some spirit – ensemble critics and decayed uncertainty – but their method aims at robust control in simultaneous adversarial perturbations, whereas ours specifically shapes an adversary's goal. Interestingly, UACER doesn't explicitly give the adversary an uncertainty-based reward; instead, they incorporate uncertainty into the protagonist's update (optimistic critics) to handle the adversary's perturbations [39]. So, conceptually, our approach (adversary reward shaping) is distinct. The near-coincidence of ideas (time-varying uncertainty consideration) indicates our work is on the frontier of current research questions.

What is *not* novel: It's also important to recognize which elements of our approach are drawn from prior art:

- Two-phase adversarial training loop – established by prior ARL work (we directly borrowed Laux et al.'s approach).
- Using negative value function as adversary reward – introduced by Laux et al. [22].
- Q-ensembles for exploration – used in prior exploration research (Pathak, Osband) and risk (Pan).
- The notion of pushing an agent to *learn from failures* – broadly present in robust/adversarial RL literature.
- SAC algorithm and tweaks like double Q, target networks – standard, not novel by themselves.

By combining these known components in a new way, the *configuration* is novel even if individual parts have precedents.

9. Gap Analysis

We now identify gaps in the literature that our work aims to fill, and how successful we are in addressing them:

Gap 1: Adversarial curricula lacking exploration of unknowns. Prior adversarial training methods (e.g., RARL, Laux ARL) mainly target states that are *known* to be challenging for the agent (like high cost or low value states). They do not explicitly encourage the adversary to search for states that the agent has not encountered. This could lead to suboptimal robustness: the protagonist might never be tested on some regions of the state space if the adversary doesn't get a signal to go explore them. Our uncertainty-based reward component fills this gap by incentivizing the adversary to find novel, uncertain states. It ensures the training covers not just the "usual suspects" of difficulty but also surprises that the agent's value function might not predict as difficult (due to lack of experience). Essentially, we bring **exploration bonus concepts into adversarial training**, plugging the hole that adversaries had no curiosity of their own.

Gap 2: Stabilizing adversarial training reward signals. A common challenge reported in adversarial and curriculum training is **stability and convergence** (PAIRED had issues with non-stationarity [71] [11], and even RARL noted some adversarial policies could be too strong and destabilize training). One reason is that the adversary's reward can be unbounded or change scales as the protagonist learns. In Laux's value-based reward, if the protagonist's value function is poorly calibrated or evolves, the adversary's incentives oscillate. We address this with normalization (keeping rewards roughly zero-mean, unit-variance) and scheduling (not hitting the protagonist with full-strength adversary immediately). This fills the gap of **lack of adaptive tuning** in prior methods. We expect these techniques to reduce variance in training curves and prevent pathological behavior (like adversary converging to an extreme strategy that yields no learning for protagonist). If our empirical results show smoother training or improved final performance, it validates that we have partly solved this gap.

Gap 3: Combining robustness and exploration for zero-shot generalization. Domain randomization and curiosity-driven exploration each improve generalization in their own way: DR covers many possible scenarios (breadth), curiosity covers the state space (depth). But neither alone guarantees **targeted worst-case coverage**. The gap is a method that can produce policies capable of *zero-shot transfer to both unexpected and worst-case scenarios*. PAIRED comes close by unsupervised environment design, but it needed an antagonist to ensure solvability and focuses on regret rather than explicit uncertainty. Our approach provides a simpler two-agent solution that still addresses both **breadth (through uncertainty exploration)** and **depth (through value challenge)** of experience. In doing so, we aim to produce a protagonist that can handle not only the difficult states it was directly trained on, but also *new* difficult states it hasn't exactly seen. For example, maybe the adversary learned to drop the agent before a particular type of trap in the maze (which it learned was hard). With uncertainty drive, perhaps it also occasionally dropped the agent in other novel trap configurations, so the agent learned a strategy for traps in general. This could result in better zero-shot success on a new maze configuration than a baseline agent. If realized, that means we filled the gap of **robustness + generalization** that individual methods struggled with.

Gap 4: Practical guidance for adversarial RL hyperparameters. There is a lack of established best practices on how to weight adversarial objectives or transition from exploratory phase to exploit phase in training. We fill this gap by proposing a concrete schedule for weights and showing its effect. This is more of an engineering gap – but important for anyone trying to reproduce or extend ARL. By reporting that a ramp-up on value weight avoids the adversary prematurely focusing on value (which might be inaccurate initially) and that decaying the uncertainty weight prevents the adversary from continuously

chasing moving "ghosts" of uncertainty even when protagonist is mostly confident, we provide insight to the community on training regime design.

Gap 5: Evaluation metrics for adversarial training. Many prior works demonstrate robustness in terms of final success rates or returns in perturbed settings, but there is less discussion on *what kind of states* the adversary actually found, or how protagonist's policy improved specifically. In our thesis work (though not fully covered in this text), we intend to analyze things like adversary endpoint state distributions (heatmaps) and protagonist's value/uncertainty over training. This addresses a gap in understanding the *dynamics of training*: how does the adversary's behavior evolve and how does it correlate with protagonist improvement? While this is not a gap we solve theoretically, our inclusion of metrics like endpoint heatmaps is a step towards more interpretable evaluation of ARL, which could be beneficial in future studies. For instance, if we see that the adversary gradually shifts from random exploratory moves to consistently positioning the agent in the center of a maze (hardest area) as training progresses, and our method vs baseline differ in how quickly or extensively that happens, that's knowledge about ARL processes.

Potential gap remaining: One gap not fully closed by our work is a theoretical guarantee of solvability or that the adversary won't generate impossible tasks. PAIRED handled that with an antagonist; ASP handled it with reversibility. We rely on empirics (and a small penalty) for that. So, while we've innovated on the reward side, there is a conceptual gap about **ensuring task feasibility** in two-player curriculum that remains. For now, our results will have to show that in practice the adversary doesn't derail training by going out-of-bound.

In summary, our work fills crucial gaps by marrying **robustness and exploration** in adversarial training, introducing techniques to make training smoother and more effective, and expanding the empirical understanding of ARL in new domains. It stands on the shoulders of prior works but pushes the boundary towards agents that are *both* highly resilient and broadly knowledgeable about their environment.

10. Recommended References

Finally, we list key references that are fundamental to understanding and situating this work. These should be cited in the thesis to acknowledge prior art and provide context:

- **Pinto et al. (2017)** – *"Robust Adversarial Reinforcement Learning."* ICML 2017.
  Introduced the two-agent adversarial training paradigm for robustness, with an adversary applying disturbances. Demonstrated improved policy generalization under worst-case perturbations. 3 21

- **Laux et al. (2020)** – *"Deep Adversarial Reinforcement Learning for Object Disentangling."* IROS 2020.
  Proposed an ARL framework very similar to ours (two-phase training, adversary sets initial states via value function guidance). Serves as baseline method for value-based adversary. 1 2

- **Dennis et al. (2020)** – *"Emergent Complexity via Unsupervised Environment Design (PAIRED)."* NeurIPS 2020.
  Presented the PAIRED algorithm with a teacher adversary maximizing regret between protagonist and antagonist. Showed automatic curriculum generation and zero-shot transfer to harder tasks. 11

- **Pathak et al. (2019)** – *"Self-Supervised Exploration via Disagreement."* ICML 2019.
  Demonstrated the use of an ensemble of models to drive exploration by maximizing their

disagreement. Key evidence that ensemble variance is a powerful intrinsic reward for efficient exploration. [7]

- **Burda et al. (2018)** – *"Large-Scale Study of Curiosity-Driven Learning."* arXiv 2018 (OpenAI).
  Systematic study of intrinsic curiosity (prediction error as reward) across many environments. Highlighted the efficacy of curiosity and potential pitfalls, providing groundwork for using intrinsic rewards. [9]

- **Sukhbaatar et al. (2018)** – *"Intrinsic Motivation and Automatic Curricula via Asymmetric Self-Play."* ICLR 2018.
  Introduced the concept of an agent setting tasks for itself (Alice and Bob) to generate a curriculum. Relevant as an alternative approach to achieve similar outcomes as adversarial training. [10]

- **Pan et al. (2019)** – *"Risk Averse Robust Adversarial RL."* arXiv 2019.
  Extended adversarial RL with risk-sensitive objectives. Notably used Q-ensembles to estimate variance (risk) and shaped both protagonist and adversary objectives accordingly. Validates use of Q variance in adversarial contexts. [13] [12]

- **Osband et al. (2016)** – *"Deep Exploration via Bootstrapped DQN."* NIPS 2016.
  Classic paper on using ensemble of Q-networks (bootstrapped heads) for efficient exploration in deep RL. Provides foundation for ensemble methods in RL. (No direct citation above, but this is an important reference in context of ensemble and uncertainty.)

- **Chen et al. (2021)** – *"Randomized Ensembled Double Q-Learning (REDQ)."* ICLR 2021.
  Demonstrated that large ensembles of Q-learners can drastically improve sample efficiency in continuous control. Informs our use of a larger critic ensemble for the protagonist. [35] [8]

- **Zhang et al. (2025)** – *"UACER: Uncertainty-Aware Critic Ensemble for Robust Adversarial RL."* arXiv 2025.
  Very recent work addressing stability in adversarial training using ensembles. Parallel idea to ours in decaying uncertainty influence. It confirms the relevance of our contributions and situates our method at the cutting edge of robust RL research. [40] [41]

Each of these references underpins a facet of our work: adversarial training frameworks, intrinsic motivation, uncertainty estimation, and curriculum learning. Citing them will give due credit and also help frame the thesis in the context of established literature. They demonstrate how our research is built upon a confluence of ideas from multiple subfields to create a novel approach to robust policy learning.

---

[1] [2] [4] [22] [23] [27] [61] Deep Adversarial Reinforcement Learning for Object Disentangling.pdf
file://file_00000000c664720ab5e26d51f0469007

[3] [18] [19] [20] [31] Robust Adversarial Reinforcement Learning
https://proceedings.mlr.press/v70/pinto17a.html

[5] [17] [21] [24] [57] [58] [62] Robust Adversarial Reinforcement Learning
http://proceedings.mlr.press/v70/pinto17a/pinto17a.pdf

[6] [1610.01283] EPOpt: Learning Robust Neural Network Policies Using Model Ensembles
https://arxiv.org/abs/1610.01283

7 37 [1906.04161] Self-Supervised Exploration via Disagreement
https://arxiv.org/abs/1906.04161

8 28 29 30 34 SUNRISE: A Simple Unified Framework for Ensemble Learning in Deep Reinforcement Learning
http://proceedings.mlr.press/v139/lee21g/lee21g.pdf

9 44 45 70 [1808.04355] Large-Scale Study of Curiosity-Driven Learning
https://arxiv.org/abs/1808.04355

10 46 47 67 68 [Quick Review] Intrinsic Motivation and Automatic Curricula via Asymmetric Self-Play
https://liner.com/review/intrinsic-motivation-and-automatic-curricula-via-asymmetric-selfplay

11 48 65 66 71 Overleaf Example
https://proceedings.mlr.press/v232/mediratta23a/mediratta23a.pdf

12 13 32 33 59 60 63 64 www2.eecs.berkeley.edu
https://www2.eecs.berkeley.edu/Pubs/TechRpts/2019/EECS-2019-164.pdf

14 15 16 38 39 40 41 42 43 54 55 UACER: An Uncertainty-Aware Critic Ensemble Framework for Robust Adversarial Reinforcement Learning
https://arxiv.org/html/2512.10492v1

25 26 49 50 69 Review for NeurIPS paper: Emergent Complexity and Zero-shot Transfer via Unsupervised Environment Design
https://proceedings.neurips.cc/paper/2020/file/985e9a46e10005356bbaf194249f6856-Review.html

35 Dropout Q-Functions for Doubly Efficient Reinforcement Learning
https://ui.adsabs.harvard.edu/abs/2021arXiv211002034H/abstract

36 [PDF] RANDOMIZED ENSEMBLED DOUBLE Q-LEARNING - OpenReview
https://openreview.net/pdf/c4e6642e1eb45768ab5d7a3291c2a0088665e5c4.pdf

51 Stochastic Actor-Critic Agent - Emergent Mind
https://www.emergentmind.com/topics/stochastic-actor-critic-agent

52 Why Soft Actor-Critic (SAC) uses a double Q trick? - AI Stack Exchange
https://ai.stackexchange.com/questions/36866/why-soft-actor-critic-sac-uses-a-double-q-trick

53 [PDF] Randomized Ensembled Double Q-Learning
https://iclr.cc/media/iclr-2021/Slides/3320.pdf

56 [PDF] Dropout Q-Functions for Doubly Efficient Reinforcement Learning
https://iclr.cc/media/iclr-2022/Slides/6233.pdf