Test-Time Risk Adaptation with Mixture of Agents

Anonymous Author(s)
Affiliation
Address
email

Abstract

In real-world reinforcement learning (RL) applications, agents often encounter unforeseen risks during deployment, necessitating robust decision-making without the luxury of further fine-tuning. While recent risk-aware RL methods incorporate return variance as a surrogate for safety, this captures only a narrow subset of real-world risks. Addressing this gap, we introduce TRAM, **T**est-time **R**isk **A**lignment with a **M**ixture of agents, a novel framework designed to enhance risk-aware decision-making during inference. TRAM operates by optimizing a weighted combination of predicted returns and a risk metric derived from state-action occupancy measures, enabling the agent to adaptively balance performance and safety in real time. Our approach allows for a nuanced representation of diverse risk factors without necessitating additional training, which does not exist in the literature. We provide theoretical sub-optimality bounds to substantiate the efficacy of our method. Empirical evaluations demonstrate that TRAM consistently outperforms existing baselines, delivering safer policies across varying risk conditions in test environments.

1 Introduction

Reinforcement learning (RL) has achieved remarkable performance in controlled settings, but its application in real-world environments remains limited by brittleness. Policies trained in simulation or narrow settings often fail when exposed to unexpected conditions at deployment. Consider an autonomous vehicle trained under typical driving scenarios but deployed in rare edge cases, such as erratic pedestrian behavior, sensor malfunctions, or newly enforced traffic regulations. These are not just performance issues—they are risks, and today's RL systems are ill-equipped to adapt to them (1). The key challenge is that test-time (or deployment time) risks are rarely the same as those modeled during training. Environment dynamics may shift, new constraints may emerge, and task priorities may change. For example, a warehouse robot trained to maximize throughput may later operate under newly imposed spatial or speed restrictions due to safety policy updates. These risks are dynamic, diverse, and often unknown ahead of time—yet traditional RL approaches assume a fixed reward and risk structure, rendering them fragile in the face of real-world uncertainty.

Why Training-Time Risk Modeling Falls Short. Risk-sensitive RL attempts to address uncertainty by optimizing for risk-aware objectives like variance or CVaR during training (2). But this assumes the deployment-time risks are both known and static. In reality, risks evolve—whether from new regulations, hardware wear-and-tear, or environmental hazards. Worse, retraining to accommodate every possible risk variant is computationally expensive, unsafe, or infeasible in many domains (e.g., robotics, finance, healthcare). Thus, fixed training-time risk modeling is fundamentally insufficient for reliable deployment.

What Is Needed: Risk Adaptation at Test-Time. To operate safely under unpredictable and shifting risk profiles, agents must adapt their behavior dynamically at test time. Rather than hard-

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

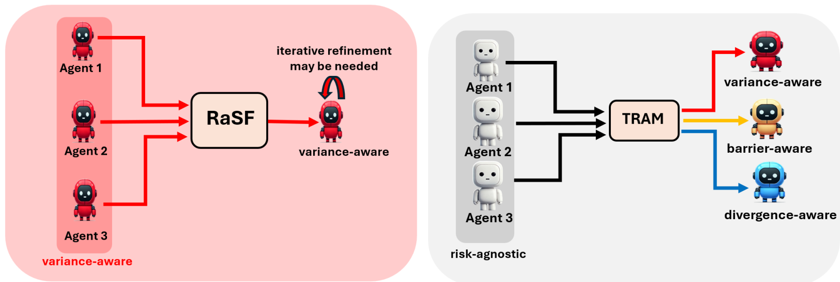

Figure 1: A high-level comparison between RaSF (2) (top) and TRAM (bottom). Unlike RaSF, TRAM (1) does not require source agents to be risk-aware, (2) supports arbitrary risk metrics beyond variance, and (3) synthesizes policies in a zero-shot fashion without additional training.

coding a single notion of risk into training, agents should be able to evaluate or synthesize behavior in real time, using deployment-specific caution signals. Such test-time risk adaptation enables safer decision-making without retraining, supporting robustness in environments where the true risks may only become apparent after deployment.

Our Proposal: TRAM. We introduce **TRAM** (Test-time Risk Alignment with a Mixture of agents), a framework that enables risk-aware behavior at test-time by composing risk-neutral source policies. TRAM does not require retraining and makes no assumptions about the risk profile beforehand. Instead, it optimizes the test-time policy via direct alignment with a specified risk measure, allowing flexible and safe adaptation under deployment-time uncertainty (see Figure 1).

We summarize our contributions as follows.

1. **Test-Time Risk-Aware Policy Framework:** We introduce TRAM, a novel test-time framework that constructs cautious policies by evaluating and composing risk-neutral source agents. Unlike prior methods, TRAM requires no retraining and makes no assumptions about the deployment-time risk profile.
2. **Generalization to Arbitrary Risk Metrics:** Our formulation accommodates a wide range of risk measures—including variance, expert divergence and occupying danger sets—enabling flexible and interpretable adaptation across safety-critical tasks.
3. **Theoretical Guarantees:** We provide sub-optimality bounds for TRAM's test-time optimization, showing that performance depends on reward mismatch and the expressivity of the deployment-time risk signal.
4. **Empirical Validation:** We evaluate TRAM across diverse RL benchmarks, demonstrating consistent improvements in safety and reliability over state-of-the-art risk-aware and test-time adaptation baselines,.

2 Related Works

Reinforcement learning agents deployed in the real world must often operate under conditions of uncertainty and unforeseen risk, particularly during the test phase. While zero-shot RL methods (3; 4; 5; 6; 7) offer impressive generalization across diverse tasks by learning unified representations, they generally lack mechanisms to incorporate risk sensitivity during test time.

Efforts to enable risk-sensitive behavior under uncertainty fall into several categories. Classical risk-aware RL methods (8; 9; 10; 11; 12; 13; 14; 15; 16) incorporate risk objectives—typically variance—into training-from-scratch, but they do not address the adaptation problem and lack mechanisms for reusing knowledge across tasks. Dual RL (17), in contrast, provides a more general framework for modeling diverse forms of risk. However, it still requires solving an optimization problem at test time, limiting its practicality in scenarios with strict inference-time constraints.

A large body of *risk-aware adaptation* methods aims to generalize across tasks while modeling risk. These include teacher-guided and critic-based architectures (18; 19), robotics-specific safety

Table 1: Comparison of TRAM with prior work. Risk-aware indicates support for risk-sensitive behavior; general risk refers to support for risk types beyond variance; and test-time means minimal or no computation is required for new tasks.

Method	Risk-aware	General Risk	Test-time
Standard RL (8; 9; 10; 11; 12; 13; 14; 15; 16)	✓	✗	✗
Zero-shot RL (3; 4; 5; 6; 7)	✗	✗	✓
Dual RL (17)	✓	✓	✗
Risk-aware Adaptation (2; 18; 19; 20; 21; 22; 23; 24; 25; 26; 27)	✓	✗	✗
TRAM (our work)	✓	✓	✓

strategies (20), probabilistic risk-based action selection (21), hierarchical or option-based controllers (22; 23; 24), and successor feature approaches (25; 26; 27). Risk-aware successor features (RaSF) (2) extend this last class by optimizing a mean-variance trade-off, enabling risk-sensitive behavior across tasks that share dynamics. However, these methods often address limited notions of risk (e.g., variance) and typically require access to risk-aware training policies or incur additional computation at test time.

In contrast, our approach—**TRAM**—requires no risk-aware policies during training, supports general forms of risk beyond variance, and performs all computation in a single forward pass at test time. TRAM does not require labeled risk profiles at training and is fully agnostic during training to the specific forms of risk that may manifest at test time. A comprehensive comparison with prior methods is presented in Table 1.

3 Problem Formulation

3.1 Preliminaries

Markov Decision Process. We model the interaction between an RL agent and its environment as a Markov Decision Process (MDP) (28), defined by the tuple $(\mathcal{S}, \mathcal{A}, p, R, \gamma)$. Here, $\mathcal{S}$ and $\mathcal{A}$ denote the state and action spaces, respectively. For a given state-action pair (s, a), the transition dynamics are governed by the probability distribution $p(\cdot \mid s, a)$ over next states $s' \in \mathcal{S}$.

Upon transition $s \xrightarrow{a} s'$, the agent receives a scalar reward drawn from the random variable $R(s, a, s')$. We denote the expected reward as $r(s, a, s') = \mathbb{E}[R(s, a, s')]$. The expected reward function is often summarized as $r(s, a) = \mathbb{E}_{s' \sim p(\cdot|s,a)}[r(s, a, s')]$, and represented in matrix form as $r \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{A}|}$. The discount factor $\gamma \in [0, 1)$ down-weights future rewards.

The agent's behavior is described by a policy $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$, where $\pi(a \mid s)$ is the probability of selecting action a in state s. The performance of π is evaluated through the **return** $G_t = \sum_{i=0}^{\infty} \gamma^i R_{t+i+1}$, the discounted cumulative reward starting at time t.

The **action-value function** (or Q-function) of a policy π is defined as:

$$Q^\pi(s, a) \equiv \mathbb{E}^\pi \left[G_t \mid S_t = s, A_t = a\right], \quad (1)$$

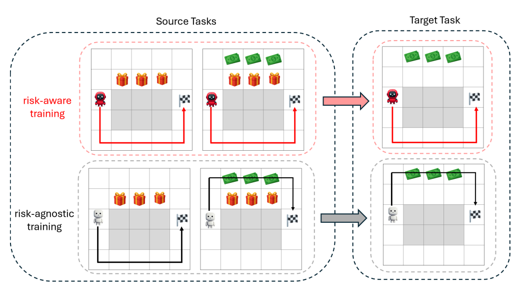

Figure 2: Illustration of a transfer setting with two source policies (left) and a single target task (right). **Visual symbol:** the **gift** icon represents a *mysterious reward*—it yields either a bomb (cost) or cash (high reward) with equal probability. Thus, even for the same policy, different runs may result in different returns due to stochasticity. **Top:** *Risk-aware training* (as in (2)) leads both source policies to avoid the upper path due to its high risk or lower expected return. As a result, the synthesized target policy also avoids the now-optimal upper path, inheriting the conservative tendencies of the source agents. **Bottom:** *Risk-agnostic training* yields more diverse source agents, each optimizing only expected return. This diversity improves coverage of the state-action space and allows the target policy to adaptively identify the safer but higher-reward upper path.

where the expectation is taken over trajectories induced by following policy π after taking action a in state s at time t.

The Q-function satisfies the Bellman equation:

$$Q^\pi(s,a) = \mathbb{E}_{\substack{s' \sim p(\cdot|s,a) \\ a' \sim \pi(\cdot|s')}} \left[r(s,a,s') + \gamma \cdot Q^\pi(s',a') \right]. \tag{2}$$

An **optimal policy** π^* maximizes expected return for all state-action pairs, satisfying $Q^{\pi^*}(s,a) = \max_\pi Q^\pi(s,a)$ for all (s,a).

3.2 Test-Time Adaptation Problem

We use the adaptation formulation presented in prior works (2; 25; 26; 29). Specifically, we consider a collection of N *source agents*, where each agent π_i^* is the optimal policy for a corresponding source task defined by the MDP:

$$\{(\mathcal{S}, \mathcal{A}, p, R_1, \gamma), \ldots, (\mathcal{S}, \mathcal{A}, p, R_N, \gamma)\}.$$

All tasks share the same dynamics and action/state spaces, but differ in their reward functions. Each source agent is trained independently and is optimal with respect to its own reward function, possibly under risk-neutral (25) or risk-sensitive (2) criteria.

At test time, a new target task $(\mathcal{S}, \mathcal{A}, p, R_\mathrm{T}, \gamma)$ is presented. No new training is allowed, and the optimal policy π_T^* for this task is unknown.[1] The best chance is to *adapt* the behavior of the source agents to produce a target policy π_T:

$$\pi_\mathrm{T} = f(\pi_1^*, \ldots, \pi_N^*), \tag{3}$$

where f is any mechanism that synthesizes a target policy from the available source agents.

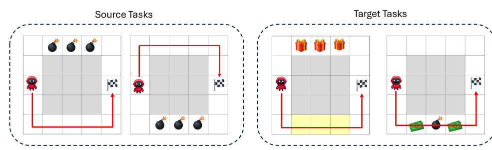

Figure 3: Failure of return variance as a general risk metric. **Visual symbols: bomb** = low or negative reward (cost), **cash** = high reward. **Left:** Source agents trained in a deterministic setting follow identical trajectories, resulting in zero return variance—even if step-level rewards vary significantly. **Middle:** In a target task with high per-step reward variability, the adapted policy chooses the higher-variance (but seemingly high-reward) path. Since return variance is zero, the agent cannot perceive the underlying risk. **Right:** In another target task, the adapted policy favors a low-return-variance path that crosses a danger zone (yellow), avoiding a safer but higher-variance path. In both cases, return variance misrepresents the true risk, leading to undesirable behavior.

Risk-Aware Adaptation. The adaptation is said to be *risk-aware* if the function f explicitly accounts for risk in the target task. A representative example is the approach proposed in (2), which augments value estimates with a penalty on return variance:

$$\pi_\mathrm{T}(s) = \arg\max_{a \in \mathcal{A}} \max_i \left(Q^{\pi_i^*}(s,a) - \frac{\beta}{2} \operatorname{Var}^{\pi_i^*}(s,a) \right), \tag{4}$$

where $\operatorname{Var}^\pi(s,a) = \operatorname{Var}[G_t \mid S_t = s, A_t = a]$, and β controls the sensitivity to risk. Since this formulation adjusts agent selection based on both return and uncertainty, it qualifies as risk-aware.

3.3 Limitations of Risk-Aware Test-Time Adaptation

We identify three key limitations in the current state-of-the-art risk-aware test-time adaptation method (2). These limitations arise when test-time policies are synthesized by selecting among source agents that were trained using a variance-regularized objective. Visual illustrations are provided in Figures 2 and 3.

L1: Risk-aware source agents reduce behavioral diversity. As shown in Figure 2, training source agents with return-variance objectives leads to overly conservative behavior across the board. This collapse in diversity limits the effectiveness of test-time agent selection, as the synthesized policy is constrained by a narrow behavioral repertoire. In contrast, risk-neutral agents—trained purely

[1] The target task typically appears during test time, when computational budgets limit training from scratch (30).

to maximize expected return—tend to exhibit more diverse trajectories, improving downstream adaptability.

L2: Return variance fails in deterministic environments. Figure 3 (left and middle) illustrates how, in deterministic settings, return variance becomes identically zero—even when rewards fluctuate at each step or when risk is structurally embedded. As a result, variance minimization fails to guide the agent toward safer or more robust behavior.

L3: Variance captures only a narrow class of risk factors. Figure 3 (right) demonstrates that return variance misses broader notions of risk, such as barrier avoidance or worst-case transitions. In this example, the agent avoids a high-variance but safe path, and instead selects a low-variance trajectory that passes through a danger zone—highlighting a critical mismatch between formal variance minimization and intuitive safety.

Summary: Risk-aware test-time adaptation methods that rely solely on variance-regularized source agents and return variance as a risk metric suffer from: (i) conservative behavior, (ii) failure in deterministic settings, and (iii) an overly narrow view of risk. Our framework, **TRAM**, addresses all three by adapting over risk-neutral agents using flexible, occupancy-based risk factors. See Appendix D for full results and examples.

4 Proposed Approach: Test-time Risk Adaption

Based on the observations made earlier, we conclude that a risk-aware adaptation framework should satisfy two key requirements: (i) the source policies π_i^* must be optimal under a *risk-agnostic* criterion, i.e., $\pi_i^*(s) = \arg\max_\pi Q^\pi(s, a)$; and (ii) the framework must support a broad class of risk models at test time, beyond the standard variance of return.

To address the second requirement, we adopt a general risk specification based on *risk factors* defined over the state-action occupancy measure d^π, where $d^\pi(s, a)$ denotes the long-term visitation frequency of state-action pair (s, a) under policy π. This formulation has been previously explored in the context of constrained or risk-sensitive RL (17), where such occupancy-based risk factors are used to shape the training objective. In contrast, TRAM leverages these risk models *only at test time*, without modifying the training process or requiring risk-aware source policies.

The occupancy-based formulation enables a wide spectrum of risk models beyond trajectory-level return variance. Examples include:

- **Barrier risk**, where the agent is penalized for visiting a danger set $\overline{S} \subset \mathcal{S}$:
$$\rho(d) = -\log\left(-d(\overline{S}) + \delta\right), \quad \text{where } d(\overline{S}) = \sum_{s,a} d(s,a)\mathbf{1}_{s\in\overline{S}}. \tag{5}$$
- **Per-step reward variance**, which captures local fluctuations in rewards:
$$\rho(d) = \text{Var}(r(s, a, s'); d) = \mathbb{E}^d\left[\left(r(s, a, s') - \mathbb{E}^d[r(s, a, s')]\right)^2\right], \tag{6}$$
where $\mathbb{E}^d := \mathbb{E}_{(s,a,s')\sim d\times p(\cdot|s,a)}$.
- **Divergence-based risks**, such as KL divergence from a known expert policy with occupancy $\bar{d}$:
$$\rho(d) = \text{KL}(d \parallel \bar{d}). \tag{7}$$

The expressiveness of these risk factors allows TRAM to support a broad range of safety, robustness, and preference constraints during test-time decision-making—while remaining fully agnostic to the objective used to train the source agents.

4.1 TRAM: Test-time Risk Alignment with a Mixture of Agents

We now introduce our proposed method, TRAM, which operationalizes the framework developed in the previous sections. Recall that our goal is to adapt at test time using a collection of pre-trained, risk-neutral source agents—without retraining them—and to do so in a way that aligns with a user-specified notion of risk.

Algorithm 1 TRAM: Test-time Risk Adaptation with a Mixture of Agents

Require: Risk-neutral source agents $\{\pi_j^*\}_{j=1}^n$; test-time risk coefficient c; risk function ρ

```
for each state s ∈ S and action a ∈ A do
    for each source agent π*_j do
        Compute Q_T^{π*_j}(s, a) in the target task
        Compute ρ_T(d^{π*_j}) in the target task
    end for
    Compute:
        π_T(a|s) = 1 if a = argmax_b max_{j=1,...,n} (Q_T^{π_j}(s, b) − c · ρ_T(d^{π_j})), 0 otherwise
end for
```

$$\pi_{\mathrm{T}}(a|s) = \begin{cases} 1 & \text{if } a = \arg\max_b \max_{j=1,\dots,n} \left(Q_{\mathrm{T}}^{\pi_j}(s,b) - c \cdot \rho_{\mathrm{T}}(d^{\pi_j})\right), \\ 0 & \text{otherwise} \end{cases}$$

Ensure: π_{T} is returned as the risk-aware test-time policy

Let $\{\pi_j^*\}_{j=1}^n$ be the set of optimal source agents, each trained independently under a different reward function. At test time, TRAM constructs a policy that selects the action with the highest adjusted value—where each agent's Q-value is penalized by a task-specific risk factor $\rho(d^{\pi_j})$. This leads to the following policy:

$$\pi_{\mathrm{T}}(a|s) = \begin{cases} 1 & \text{if } a = \arg\max_b \max_{j=1,2,\dots,n} \left(Q_{\mathrm{T}}^{\pi_j}(s,b) - c \cdot \rho_{\mathrm{T}}(d^{\pi_j})\right) \\ 0 & \text{otherwise,} \end{cases} \tag{8}$$

Here, $c \geq 0$ is a risk-weighting coefficient that balances expected return (via Q) with test-time risk (via ρ). Crucially, ρ can be instantiated using any of the general risk factors defined earlier—such as barrier risks, reward variance, or divergence from expert behavior (see Equations 5, 6, and 7).

This formulation ensures that TRAM makes a risk-aware decision by aggregating across a set of agents trained without any risk signal, while still respecting the user-defined safety or robustness constraints of the target task. The pseudocode for computing the TRAM policy is provided in Algorithm 1 below.

4.2 Theoretical Insights

We now analyze the performance guarantees of TRAM by quantifying how far its test-time policy can deviate from the optimal risk-aware policy in the target task. This deviation arises from two key sources:

- **Reward mismatch:** The difference between the reward function of the target task r_{T} and that of the closest source task r_i.
- **Risk misalignment:** The cost of introducing a test-time risk penalty that was not present during source agent training.

To capture this formally, we define the error between the risk-adjusted value of the TRAM policy and the optimal risk-aware policy as:

$$\left|\tilde{Q}_{\mathrm{T}}^{\pi_{\mathrm{T}}^*}(s,a) - \tilde{Q}_{\mathrm{T}}^{\pi_{\mathrm{T}}}(s,a)\right|,$$

where $\tilde{Q}$ denotes a Q-value adjusted by a test-time risk factor:

$$\tilde{Q}_{\mathrm{T}}^{\pi_i^*}(s,a) = Q_{\mathrm{T}}^{\pi_i^*}(s,a) - c \cdot \rho_{\mathrm{T}}(d^{\pi_i^*}).$$

The TRAM policy selects actions using:

$$\pi_{\mathrm{T}}(a|s) = \begin{cases} 1 & \text{if } a = \arg\max_b \max_{i=1,\dots,n} \tilde{Q}_{\mathrm{T}}^{\pi_i^*}(s,b) \\ 0 & \text{otherwise.} \end{cases}$$

The following theorem bounds the performance gap between the TRAM policy and the optimal risk-aware policy in the target task:

Theorem 4.1. *Let $Q_T^{\pi_i^*}$ denote the value function of source agent π_i^* evaluated in target task M_T, and let $\rho_T(d^{\pi_i^*})$ be an L-Lipschitz risk factor bounded by K. Then the TRAM policy π_T satisfies:*

$$\left|\tilde{Q}_T^{\pi_T^*}(s,a) - \tilde{Q}_T^{\pi_T}(s,a)\right| \leq \min_i \left(\frac{2}{1-\gamma} \|r_T - r_i\|_\infty + (4L+K)\cdot c \right). \tag{9}$$

Proof. See Appendix C for the full derivation. □

Theoretical insights. This bound separates the impact of reward mismatch from the influence of the test-time risk factor. When $c = 0$, TRAM reduces to reward-only adaptation as in (25). When $r_T = r_i$ and $c = 0$, the bound is zero. However, if $c > 0$, the risk-aware optimum may differ—even if the task is known exactly—highlighting the importance of aligning with risk during adaptation.

Implication: TRAM supports risk-sensitive test-time decision-making using only risk-agnostic agents, with error that scales smoothly in both the reward and risk discrepancy.

4.3 A Practical Implementation at Test Time: Successor Features (SFs) ψ

A practical instantiation of TRAM requires fast computation of action-values across source agents. Traditional value evaluation methods typically involve iterative rollouts or dynamic programming, with complexity $\mathcal{O}\left(\frac{1}{\epsilon(1-\gamma)}\right)$, where ϵ is the desired approximation error in the value function. A more scalable alternative leverages *successor features* (SFs), which exploit shared dynamics and reward structure across tasks (25; 26).

Suppose the reward function factorizes as $r(s,a,s') = \phi(s,a,s')^\top \mathbf{w}$, where ϕ is a shared feature map and $\mathbf{w}$ is a task-specific weight vector. Then, the successor feature vector of a policy π, denoted $\psi^\pi(s,a)$, is defined as:

$$\psi^\pi(s,a) = \mathbb{E}^\pi \left[\sum_{i=t}^{\infty} \gamma^{i-t} \phi(S_i, A_i, S_{i+1}) \mid S_t = s, A_t = a \right].$$

This enables efficient computation of the action-value function as a simple dot product:

$$Q^\pi(s,a) = \psi^\pi(s,a)^\top \mathbf{w}. \tag{10}$$

When TRAM is implemented using SF-based agents, we obtain the following bound:

Corollary 4.2. *Under the same assumptions as Theorem 4.1, and assuming that $\|\phi(s,a,s')\| \leq \phi_{\max}$ for all (s,a,s'), we have:*

$$\tilde{Q}_T^{\pi_T^*}(s,a) - \tilde{Q}_T^{\pi_T}(s,a) \leq \min_i \left(\frac{2\phi_{\max}}{1-\gamma} \|\mathbf{w}_T - \mathbf{w}_i\| + (4L+K)\cdot c \right), \tag{11}$$

where $\tilde{Q}$ is defined using the dot product in Equation (10) *with a test-time risk adjustment.*

Proof. See Appendix C for a detailed derivation. □

5 Experiments

To evaluate TRAM, we first test it in a controlled gridworld environment similar to (2; 17; 25). This setting enables direct comparison against prior risk-aware adaptation methods and helps answer the following:

Q1) *Does TRAM support more general notions of risk compared to RaSF (2)?*

Q2) *Can TRAM avoid the overly conservative behavior exhibited by RaSF?*

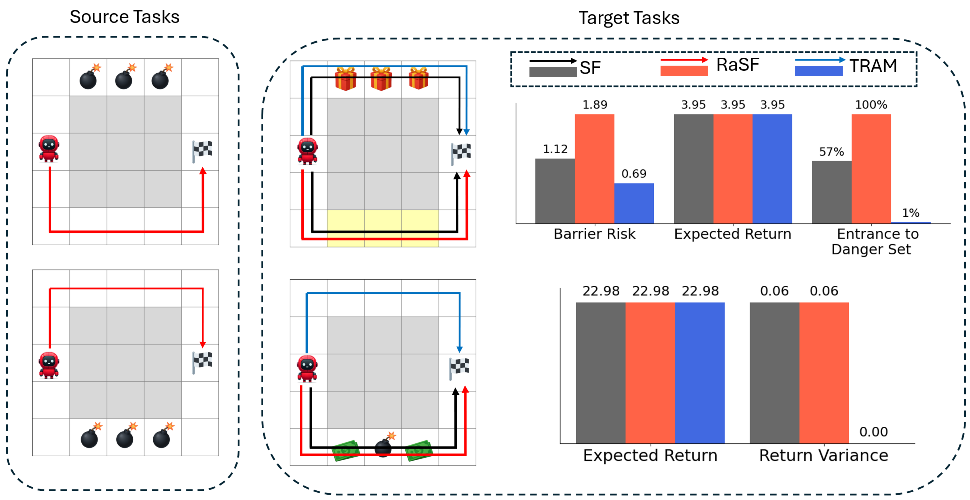

Figure 4: Visualization of Experiment 1. **Left:** Two source policies trained on distinct tasks. **Middle:** Two target tasks. **Top row:** In the first task, the agent must avoid a danger region (yellow). TRAM, using the barrier risk from Eq. (5), avoids the danger zone entirely. In contrast, RaSF—trained to minimize return variance—fails to detect spatial risk and enters the danger zone in every episode. SF also lacks risk awareness and behaves similarly. **Bottom row:** In the second task, risk arises from per-step reward variance as defined in Eq. (6). RaSF, which uses return-level variance, is blind to this finer granularity and selects a high-variance path. SF also fails due to the absence of risk modeling. TRAM, by contrast, successfully avoids the high-variance trajectory. **Right:** Bar plots show expected return and return variance, reflecting the qualitative differences in policy behavior.

Setup. The agent navigates from a start cell to a goal cell. Rewards and risks are distributed across different paths, visualized using symbols (e.g., gifts, bombs). The goal is to maximize expected return while avoiding high-risk regions. Risk arises either from danger zones or local reward variance.

Baselines. We compare against two methods: (i) RaSF, where source policies are trained with a variance-based penalty, and (ii) the risk-agnostic SF method from (25), which uses only expected return. TRAM, by contrast, uses these risk-agnostic agents but aligns them at test time using general risk factors.

Experiment 1: General risk representations. Figure 4 illustrates two target tasks. In the first (top), risk is defined via a danger zone. TRAM, using the barrier risk factor in Eq. (5), avoids this region while maintaining return. RaSF and SF frequently enter the danger zone. In the second task (bottom), risk stems from per-step reward variance. TRAM, using Eq. (6), selects the safer path, while RaSF and SF fail to detect this form of risk.

Experiment 2: Robustness to risk shifts. Figure 5 shows two more test cases. In the first (top), no risk is present. TRAM and SF exploit the high-reward path, while RaSF—trained with built-in risk aversion—remains overly conservative. In the second case (bottom), risk is introduced via stochastic rewards on the high-return path. TRAM correctly shifts to the lower, safer path. RaSF does the same, but SF fails due to its lack of risk modeling.

Conclusion. Unlike RaSF, which is restricted to a fixed form of risk and conservatively-trained agents, TRAM adapts to multiple risk types at test time and dynamically balances safety and performance using general risk factors.

5.1 Generalization to Continuous Domains

While our main experiments focus on discrete environments to highlight the limitations of prior risk-aware adaptation methods, we also demonstrate that TRAM extends naturally to high-dimensional continuous control. Specifically, we evaluate TRAM on the Reacher domain, a widely used continuous-space benchmark in the adaptation literature (2; 25; 29). This setting introduces real-valued states

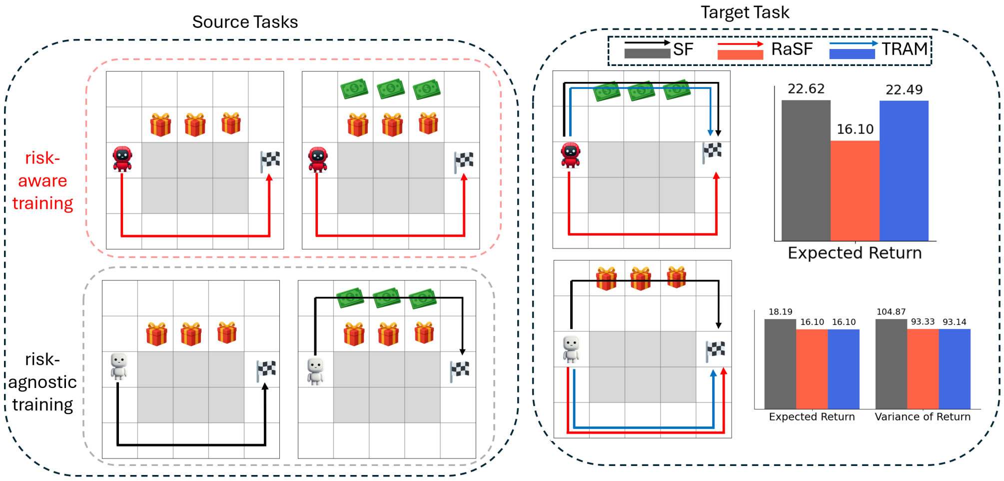

Figure 5: Experiment 2 setup. **Left:** Source policies trained on distinct tasks using either risk-agnostic or risk-aware objectives. **Middle:** Two target tasks. **Top row:** In the first target task, no risk is present. TRAM and SF successfully choose the path with the highest expected return. RaSF, however, remains overly conservative due to training with a fixed variance-based risk signal, and fails to exploit the high-return option. **Bottom row:** In the second target task, risk is introduced via stochastic rewards (gifts). TRAM and RaSF both avoid the high-variance path—since the return variability depends on the gift's sign—while SF, unaware of risk, continues to follow the high-reward but volatile route. **Right:** Bar graphs quantify the trade-off between expected return and return variance across the three methods.

and actions, nonlinear dynamics, and the need for deep function approximation—all of which are common in robotics applications.

The Reacher environment consists of a two-joint torque-controlled robotic arm that must reach a specified target in the plane. The dynamics are simulated via MuJoCo (31), yielding a continuous 4D state space and nontrivial transitions. We train source agents using Successor Feature Deep Q-Networks (SFDQNs) on multiple risk-agnostic tasks—without any form of risk modeling during training.

At test time, we apply TRAM with a barrier risk function to adapt these agents to a new task that includes a danger region. No additional training or fine-tuning is performed. As detailed in Appendix F, TRAM significantly reduces the failure rate (i.e., entering the danger zone) relative to standard SF adaptation (25), while achieving comparable accuracy in reaching the goal.

Takeaway: TRAM scales beyond tabular and gridworld domains. It supports expressive risk specifications, operates in real-time via deep function approximators, and adapts effectively under nonlinear continuous dynamics.

6 Conclusions

We introduced **TRAM**, a novel test-time adaptation framework that derives risk-aware policies from risk-neutral source agents. Unlike prior methods requiring risk-aware training or limited to return variance, TRAM supports general, user-defined risk factors—such as barrier constraints, per-step reward variance, or divergence from expert behavior—evaluated solely at test time.

By leveraging successor features, TRAM enables fast policy synthesis via dot-product computations, avoiding sampling or rollouts. Our theoretical analysis offers performance bounds that decouple reward mismatch from risk misalignment. Experiments across discrete and continuous domains demonstrate that TRAM captures richer risk signals and generalizes better than existing methods, with low computational cost. *Future work* includes extending TRAM to tasks with differing dynamics.

References

[1] Gong, T., J. Jeong, T. Kim, et al. Note: Robust continual test-time adaptation against temporal correlation. *Advances in Neural Information Processing Systems*, 35:27253–27266, 2022.

[2] Gimelfarb, M., A. Barreto, S. Sanner, et al. Risk-aware transfer in reinforcement learning using successor features. *Advances in Neural Information Processing Systems*, 34:17298–17310, 2021.

[3] Marom, O., B. Rosman. Zero-shot transfer with deictic object-oriented representation in reinforcement learning. *Advances in Neural Information Processing Systems*, 31, 2018.

[4] Oh, J., S. Singh, H. Lee, et al. Zero-shot task generalization with multi-task deep reinforcement learning. In *International Conference on Machine Learning*, pages 2661–2670. PMLR, 2017.

[5] Higgins, I., A. Pal, A. Rusu, et al. Darla: Improving zero-shot transfer in reinforcement learning. In *International Conference on Machine Learning*, pages 1480–1490. PMLR, 2017.

[6] Rezaei-Shoshtari, S., C. Morissette, F. R. Hogan, et al. Hypernetworks for zero-shot transfer in reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, pages 9579–9587. 2023.

[7] Touati, A., J. Rapin, Y. Ollivier. Does zero-shot reinforcement learning exist? In *The Eleventh International Conference on Learning Representations*. 2022.

[8] Bisi, L., L. Sabbioni, E. Vittori, et al. Risk-averse trust region optimization for reward-volatility reduction. *arXiv preprint arXiv:1912.03193*, 2019.

[9] Fei, Y., Z. Yang, Y. Chen, et al. Risk-sensitive reinforcement learning: Near-optimal risk-sample tradeoff in regret. *Advances in Neural Information Processing Systems*, 33:22384–22395, 2020.

[10] Jain, A., G. Patil, A. Jain, et al. Variance penalized on-policy and off-policy actor-critic. In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, pages 7899–7907. 2021.

[11] Mannor, S., J. N. Tsitsiklis. Algorithmic aspects of mean–variance optimization in markov decision processes. *European Journal of Operational Research*, 231(3):645–653, 2013.

[12] Mao, H., S. B. Venkatakrishnan, M. Schwarzkopf, et al. Variance reduction for reinforcement learning in input-driven environments. *arXiv preprint arXiv:1807.02264*, 2018.

[13] Nass, D., B. Belousov, J. Peters. Entropic risk measure in policy search. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1101–1106. IEEE, 2019.

[14] Shen, Y., M. J. Tobia, T. Sommer, et al. Risk-sensitive reinforcement learning. *Neural computation*, 26(7):1298–1328, 2014.

[15] Tamar, A., D. Di Castro, S. Mannor. Learning the variance of the reward-to-go. *Journal of Machine Learning Research*, 17(13):1–36, 2016.

[16] Whiteson, S. Mean- variance policy iteration for risk- averse reinforcement learning. 2021.

[17] Zhang, J., A. S. Bedi, M. Wang, et al. Cautious reinforcement learning via distributional risk in the dual domain. *IEEE Journal on Selected Areas in Information Theory*, 2(2):611–626, 2021.

[18] Turchetta, M., A. Kolobov, S. Shah, et al. Safe reinforcement learning via curriculum induction. *Advances in Neural Information Processing Systems*, 33:12151–12162, 2020.

[19] Srinivasan, K., B. Eysenbach, S. Ha, et al. Learning to be safe: Deep rl with a safety critic. *arXiv preprint arXiv:2010.14603*, 2020.

[20] Held, D., Z. McCarthy, M. Zhang, et al. Probabilistically safe policy transfer. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5798–5805. IEEE, 2017.

[21] García, J., F. Fernández. Probabilistic policy reuse for safe reinforcement learning. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 13(3):1–24, 2019.

[22] Mankowitz, D. J., A. Tamar, S. Mannor. Situational awareness by risk-conscious skills. *arXiv preprint arXiv:1610.02847*, 2016.

[23] Jain, A., K. Khetarpal, D. Precup. Safe option-critic: learning safety in the option-critic architecture. *The Knowledge Engineering Review*, 36:e4, 2021.

[24] Mankowitz, D., T. Mann, P.-L. Bacon, et al. Learning robust options. In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32. 2018.

[25] Barreto, A., W. Dabney, R. Munos, et al. Successor features for transfer in reinforcement learning. *Advances in neural information processing systems*, 30, 2017.

[26] Barreto, A., D. Borsa, J. Quan, et al. Transfer in deep reinforcement learning using successor features and generalised policy improvement. In *International Conference on Machine Learning*, pages 501–510. PMLR, 2018.

[27] Barreto, A., S. Hou, D. Borsa, et al. Fast reinforcement learning with generalized policy updates. *Proceedings of the National Academy of Sciences*, 117(48):30079–30087, 2020.

[28] Puterman, M. L. Markov decision processes. *Handbooks in operations research and management science*, 2:331–434, 1990.

[29] Zhang, S., H. D. Fernando, M. Liu, et al. Sf-dqn: Provable knowledge transfer using successor feature for deep reinforcement learning. *arXiv preprint arXiv:2405.15920*, 2024.

[30] Chakraborty, S., S. S. Ghosal, M. Yin, et al. Transfer q star: Principled decoding for llm alignment. *arXiv preprint arXiv:2405.20495*, 2024.

[31] Todorov, E., T. Erez, Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

[32] Sutton, R. S., A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[33] Bellman, R. *Dynamic Programming*. Dover Publications, 1957.

[34] Nachum, O., B. Dai. Reinforcement learning via fenchel-rockafellar duality. *arXiv preprint arXiv:2001.01866*, 2020.

[35] Nachum, O., B. Dai, I. Kostrikov, et al. Algaedice: Policy gradient from arbitrary experience. *arXiv preprint arXiv:1912.02074*, 2019.

[36] Devroye, L., A. Mehrabian, T. Reddad. The total variation distance between high-dimensional gaussians with the same mean. *arXiv preprint arXiv:1810.08693*, 2018.

[37] Wen, Z., B. Van Roy. Efficient reinforcement learning in deterministic systems with value function generalization. *Mathematics of Operations Research*, 42(3):762–782, 2017.

[38] Nagarajan, P., G. Warnell, P. Stone. Deterministic implementations for reproducibility in deep reinforcement learning. *arXiv preprint arXiv:1809.05676*, 2018.

Contents

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly describe the core contributions of TRAM, namely risk-aware adaptation at test time using risk-neutral agents and general risk factor alignment without test-time optimization. These claims are substantiated by theory and experiments.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are acknowledged in the conclusion, which discusses extending TRAM to tasks with different transition dynamics. This highlights a current assumption in our approach and outlines a direction for future work.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: All theoretical results (Theorem 4.1, Corollary 4.2) are fully stated with assumptions, and complete proofs are provided in Appendix C.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper provides all environment configurations, risk specifications, and algorithmic components for the gridworld experiments. For the Reacher experiment, we shall include the full setup and reference base implementations.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The paper provides all environment configurations, risk specifications, and algorithmic components for the gridworld experiments. For the Reacher experiment, we shall include the full setup and reference base implementations.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper provides all environment configurations, risk specifications, and algorithmic components for the gridworld experiments. For the Reacher experiment, we shall include the full setup and reference base implementations.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

Justification: All results are aggregated over multiple rollouts to capture stochasticity. Bar graphs in Figures 4, 5, and others reflect mean and variance across runs, supporting statistical robustness.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The appendix provides compute details, including number of CPUs/GPUs used, runtime for both tabular and continuous experiments, and environments needed to replicate the results.

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work follows standard RL evaluation practices and conforms to NeurIPS ethical guidelines. No private or sensitive data is used, and there are no foreseeable risks to safety, privacy, or fairness.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss the broader impact in H.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The work does not release high-risk data or models. The paper focuses on MDP-based simulation environments.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All prior works and open-source codebases (e.g., MuJoCo, SF-DQN) are cited appropriately. We follow all licensing terms for code reuse.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The paper provides all environment configurations, risk specifications, and algorithmic components for the gridworld experiments. For the Reacher experiment, we shall include the full setup and reference base implementations.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

Justification: No human subjects or crowdsourced components are involved in this work.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This research does not involve human subjects.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research?

    Answer: [NA]

    Justification: Large language models were not used in the research methodology or experimentation.

Appendix

A Q-LP Formulation of RL

The problem of computing $Q^\pi(s,a)$ is known as *policy evaluation*, or the *prediction problem* (32). While most works focus on dynamic programming (DP) methods for policy evaluation (33), we consider an alternative approach based on the linear programming (LP) formulation of the RL problem (34), as the dual variables of the LP problem facilitate risk-aware behavior.

The primal LP problem, which we refer to as the *Q-LP problem*, is defined below:

$$\begin{aligned} \min_{Q} \quad & (1-\gamma)\cdot \mathbb{E}_{\substack{a_0\sim\pi(s_0)\\ s_0\sim\mu_0}}[Q(s_0,a_0)] \\ \text{s.t.} \quad & Q(s,a) \geq \mathbb{E}_{\substack{s'\sim p(\cdot|s,a)\\ a'\sim\pi(a|s)}}\left[r(s,a,s') + \gamma\cdot Q^\pi(s',a')\right], \\ & \qquad \forall s\in\mathcal{S}, a\in\mathcal{A}. \end{aligned} \tag{12}$$

where μ_0 is the initial distribution over the states. The optimal Q of the problem satisfies $Q^*(s,a) = Q^\pi(s,a)$. One can refer to the appendix of (35) for a full proof of the formulation.

The dual problem of the Q-LP, with dual variable $d \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{A}|}$, is shown below:

$$\begin{aligned} \max_{d\geq 0} & \sum_{s,a} d(s,a)\cdot \mathbb{E}_{\substack{s'\sim p(\cdot|s,a)\\ a'\sim\pi(a|s)}}\left[r(s,a,s')\right], \\ \text{s.t.} \quad & d(s,a) = (1-\gamma)\mu_0(s)\pi(a|s) + \gamma\cdot \mathbb{E}_{\substack{s'\sim p(\cdot|s,a)\\ a'\sim\pi(a|s)}}\left[d(s',a')\right], \\ & \qquad \forall s\in\mathcal{S}, a\in\mathcal{A}. \end{aligned} \tag{13}$$

The dual variable d is known as the *state-action occupancy measure* Given a policy π, $d(s,a)$ represents the joint probability of occupying a state s and taking an action a from that state. Furthermore, one can recover the policy π from the occupancy measure as follows:

$$\pi(a|s) = \frac{d(s,a)}{\sum_{a'\in\mathcal{A}} d(s,a')} \quad \forall a\in\mathcal{A}, s\in\mathcal{S}, \tag{14}$$

B Dual V-LP Formulation of RL

As opposed to the dual Q-LP formulation(13), which finds the state-action occupancy measure d^π for a *given* π, the dual V-LP formulation shown below finds the state-action occupancy measure d^{π^*} corresponding to the *optimal* policy π^*:

$$\begin{aligned} \max_{d\geq 0} & \sum_{s,a} d(s,a)\cdot \mathbb{E}_{\substack{s'\sim p(\cdot|s,a)\\ a'\sim\pi(a|s)}}\left[r(s,a,s')\right] - c_i\cdot\rho(d), \\ \text{s.t.} \quad & \sum_{a\in\mathcal{A}} d(s,a) = (1-\gamma)\mu_0(s) + \gamma\cdot \sum_{a\in\mathcal{A}} \mathbb{E}_{\substack{s'\sim p(\cdot|s,a)\\ a'\sim\pi(a|s)}}\left[d(s',a')\right], \\ & \qquad \forall s\in\mathcal{S}. \end{aligned} \tag{15}$$

C Proof of the Theorem and its Corollary

First, we need to define some variables to make the proofs clear:

- $Q_i^{\pi_i,\mathrm{RN}^*}$ is the action-value function of the risk-neutral optimal policy of task i obtained using any standard RL method that maximizes expected return.
- $Q_i^{\pi_i,\mathrm{RA}^*}$ is the action-value function of the risk-aware optimal policy of task i obtained by solving (15).

- $Q_i^{\pi_j,\mathrm{RN}^*}$ is the action-value function of the risk-neutral optimal policy of task j when evaluated in task i.

Define $\tilde{Q}^{\pi}(s,a) = Q(s,a) - c\rho(d^{\pi})$, so:

- $\tilde{Q}_i^{\pi_i,\mathrm{RA}^*} = Q_i^{\pi_i,\mathrm{RA}^*} - c\rho(d^{\pi_i,\mathrm{RA}^*})$
- $\tilde{Q}_i^{\pi_j,\mathrm{RN}^*} = Q_i^{\pi_j,\mathrm{RN}^*} - c\rho(d^{\pi_j,\mathrm{RN}^*})$

Lemma C.1.

$$\left|Q_i^{\pi_i,RN^*}(s,a) - Q_j^{\pi_j,RN^*}(s,a)\right| \leq \frac{1}{1-\gamma}\|r_i - r_j\|_\infty.$$

Proof of Optimal-Optimal. Let $\Delta_{ij} = \max_{s,a}\left|Q_i^{\pi_i,\mathrm{RN}^*}(s,a) - Q_j^{\pi_j,\mathrm{RN}^*}(s,a)\right|$.

Step 1: Bellman Optimality This equation follows from Bellman optimality as each of π_i, RN^* and π_j, RN^* is risk-neutral optimal in their own tasks.

$$\left|Q_i^{\pi_i,\mathrm{RN}^*}(s,a) - Q_j^{\pi_j,\mathrm{RN}^*}(s,a)\right| = \Bigg|r_i(s,a) + \gamma\sum_{s'} p(s'|s,a)\max_b Q_i^{\pi_i,\mathrm{RN}^*}(s',b) \tag{16}$$

$$-r_j(s,a) - \gamma\sum_{s'} p(s'|s,a)\max_b Q_j^{\pi_j,\mathrm{RN}^*}(s',b)\Bigg| \tag{17}$$

Step 2: Simplification Simplifying the above expression:

$$= \left|r_i(s,a) - r_j(s,a) + \gamma\sum_{s'} p(s'|s,a)\left(\max_b Q_i^{\pi_i,\mathrm{RN}^*}(s',b) - \max_b Q_j^{\pi_j,\mathrm{RN}^*}(s',b)\right)\right| \tag{18}$$

Step 3: Triangle Inequality Applying the triangle inequality:

$$\leq |r_i(s,a) - r_j(s,a)| + \gamma\sum_{s'} p(s'|s,a)\left|\max_b Q_i^{\pi_i,\mathrm{RN}^*}(s',b) - \max_b Q_j^{\pi_j,\mathrm{RN}^*}(s',b)\right| \tag{19}$$

Step 4: Maximum Difference The difference of maxima is less than the maximum of differences:

$$\leq |r_i(s,a) - r_j(s,a)| + \gamma\sum_{s'} p(s'|s,a)\max_b\left|Q_i^{\pi_i,\mathrm{RN}^*}(s',b) - Q_j^{\pi_j,\mathrm{RN}^*}(s',b)\right| \tag{20}$$

Step 5: Definition of Δ By definition of Δ_{ij}:

$$\leq \|r_i - r_j\|_\infty + \gamma\Delta_{ij}. \tag{21}$$

Step 6: Substituting Δ Since C applies $\forall s \in \mathcal{S}, \forall a \in \mathcal{A}$, it applies particularly for Δ_{ij}:

$$\Delta_{ij} \leq \|r_i - r_j\|_\infty + \gamma\Delta_{ij} \tag{22}$$

□

Lemma C.2.

$$\left|Q_j^{\pi_j,RN^*}(s,a) - Q_i^{\pi_j,RN^*}(s,a)\right| \leq \frac{1}{1-\gamma}\|r_i - r_j\|_\infty. \tag{23}$$

Proof. Let $\Delta_{ij} = \max_{s,a}\left|Q_i^{\pi_j,\mathrm{RN}^*}(s,a) - Q_j^{\pi_j,\mathrm{RN}^*}(s,a)\right|$.

Step 1: Bellman Recurrence The Bellman recurrence is applied to the action-value functions under policy π_j^*:

$$\left|Q_j^{\pi_j,\mathrm{RN}^*}(s,a) - Q_i^{\pi_j,\mathrm{RN}^*}(s,a)\right| = \Bigg| r_j(s,a) + \gamma \sum_{s'} p(s' \mid s,a) Q_j^{\pi_j,\mathrm{RN}^*}(s', \pi_j^*(s')) \tag{24}$$

$$-r_i(s,a) - \gamma \sum_{s'} p(s'|s,a) \max_b Q_i^{\pi_j,\mathrm{RN}^*}(s',b) \Bigg| \tag{25}$$

Step 2: Simplification Simplifying the expression:

$$= \left| r_j(s,a) - r_i(s,a) + \gamma \sum_{s'} p(s' \mid s,a) \left(Q_j^{\pi_j,\mathrm{RN}^*}(s', \pi_j^*(s')) - Q_i^{\pi_j,\mathrm{RN}^*}(s', \pi_j^*(s')) \right) \right| \tag{26}$$

Step 3: Triangle Inequality Applying the triangle inequality to further simplify:

$$\leq |r_i(s,a) - r_j(s,a)| + \gamma \sum_{s'} p(s' \mid s,a) \left| Q_j^{\pi_j,\mathrm{RN}^*}(s', \pi_j^*(s')) - Q_i^{\pi_j,\mathrm{RN}^*}(s', \pi_j^*(s')) \right| \tag{27}$$

Step 4: Definition of Δ' By the definition of Δ'_{ij}:

$$\leq \|r_i - r_j\|_\infty + \gamma \Delta'_{ij} \tag{28}$$

Step 5: Substituting Δ' Since the inequality holds $\forall s \in \mathcal{S}, \forall a \in \mathcal{A}$, it applies particularly for Δ'_{ij}, allowing the substitution:

$$\Delta'_{ij} \leq \|r_i - r_j\|_\infty + \gamma \Delta'_{ij} \tag{29}$$

□

Lemma C.3. *Let d^{π_1} and d^{π_2} be two discrete probability distributions. Then,*

$$\|d^{\pi_1} - d^{\pi_2}\|_1 \leq 2. \tag{30}$$

Proof. **Step 1: Define the L1-norm.** The L1-norm of $d^{\pi_1} - d^{\pi_2}$ is defined as:

$$\|d^{\pi_1} - d^{\pi_2}\|_1 = \sum_x |d^{\pi_1}(x) - d^{\pi_2}(x)| \tag{31}$$

Step 2: Define the total variation norm (TV) for distance between probability distributions. The total variation distance between d^{π_1} and d^{π_2} is given by:

$$\mathrm{TV}(d^{\pi_1}, d^{\pi_2}) = \frac{1}{2} \sum_x |d^{\pi_1}(x) - d^{\pi_2}(x)| \tag{32}$$

Step 3: Bounded total variation norm. According to (36), the total variation distance between two probability distributions is always bounded by 1:

$$\mathrm{TV}(d^{\pi_1}, d^{\pi_2}) \leq 1 \tag{33}$$

Step 4: L1-norm for distributions is bounded. From the definition of the total variation norm, the L1-norm can be expressed as twice the total variation distance:

$$\|d^{\pi_1} - d^{\pi_2}\|_1 = 2 \cdot \mathrm{TV}(d^{\pi_1}, d^{\pi_2}) \leq 2 \tag{34}$$

Thus, the L1-norm of the difference between the two probability distributions is bounded by 2. □

Lemma C.4. *If ρ is L-Lipschitz, then:*

$$\left|\rho(d^{\pi_i,RA^*}) - \rho(d^{\pi_j,RN^*})\right| \leq 2 \cdot L \tag{35}$$

Proof. **Step 1:**

Lipschitz Continuity. Since ρ is L-Lipschitz, we have:

$$|\rho(d^{\pi_i,\mathrm{RA}^*}) - \rho(d^{\pi_j,\mathrm{RN}^*})| \leq L\|d^{\pi_j,\mathrm{RN}^*} - d^{\pi_i,\mathrm{RN}^*}\|_1, \tag{36}$$

where $\|d^{\pi_j,\mathrm{RN}^*} - d^{\pi_i,\mathrm{RN}^*}\|_1$ is the L1-norm of the difference between the probability distributions d^{π_j,RN^*} and d^{π_i,RN^*}.

Step 2:

Bounding the L1-norm. From Lemma C.3 we know that:

$$\|d^{\pi_j,\mathrm{RN}^*} - d^{\pi_i,\mathrm{RN}^*}\|_1 \leq 2 \tag{37}$$

Step 3:

Combining the Equations By combining the two equations above, we obtain:

$$|\rho(d^{\pi_i,\mathrm{RA}^*}) - \rho(d^{\pi_j,\mathrm{RN}^*})| \leq 2L \tag{38}$$

□

Lemma C.5. *If ρ is L-Lipschitz. Then,*

$$|Q_i^{\pi_i,RA^*}(s,a) - Q_i^{\pi,RN^*}(s,a)| \leq 2 \cdot \gamma \cdot L \cdot c. \tag{39}$$

Proof. **Step 1: Defining the action-value functions** We begin by expressing the action-value functions in terms of the dual variable d, which represents the policy-specific adjustments:

$$Q_i^{\pi_i,\mathrm{RA}^*}(s,a) = r_i(s,a) + \gamma \sum_{s'} p(s'|s,a)\langle d^{\pi_i,\mathrm{RA}^*}, r_i\rangle, \tag{40}$$

$$Q_i^{\pi_i,\mathrm{RN}^*}(s,a) = r_i(s,a) + \gamma \sum_{s'} p(s'|s,a)\langle d^{\pi_i,\mathrm{RN}^*}, r_i\rangle. \tag{41}$$

Step 2: Calculating the difference The difference in the action-value functions is then given by:

$$|Q_i^{\pi_i,\mathrm{RA}^*}(s,a) - Q_i^{\pi_i,\mathrm{RN}^*}(s,a)| = \gamma \sum_{s'} p(s'|s,a)|\langle d^{\pi_i,\mathrm{RA}^*}, r_i\rangle - \langle d^{\pi_i,\mathrm{RN}^*}, r_i\rangle|. \tag{42}$$

Step 3: Bounding difference in return in the dual form Given (15), d^{π_i,RA^*} maximizes $\langle d, r_i\rangle - c\rho(d)$, across feasible occupancy measures d, then:

$$\langle d^{\pi_i,\mathrm{RA}^*}, r_i\rangle - c\rho(d^{\pi_i,\mathrm{RA}^*}) \geq \langle d^{\pi_i,\mathrm{RN}^*}, r_i\rangle - c\rho(d^{\pi_i,\mathrm{RN}^*}), \tag{43}$$

$$\Leftrightarrow \langle d^{\pi_i,\mathrm{RN}^*}, r_i\rangle - \langle d^{\pi_i,\mathrm{RA}^*}, r_i\rangle \leq c(\rho(d^{\pi_i,\mathrm{RA}^*}) - \rho(d^{\pi_i,\mathrm{RN}^*})), \tag{44}$$

$$\Leftrightarrow |\langle d^{\pi_i,\mathrm{RA}^*}, r_i\rangle - \langle d^{\pi_i,\mathrm{RN}^*}, r_i\rangle| \leq c|\rho(d^{\pi_i,\mathrm{RA}^*}) - \rho(d^{\pi_i,\mathrm{RN}^*})|. \tag{45}$$

Step 4: Using Lipschitz continuity Assuming ρ is Lipschitz continuous with constant L and analyzing the optimization criteria:

$$|\langle d^{\pi_i,\mathrm{RA}^*}, r_i\rangle - \langle d^{\pi_i,\mathrm{RN}^*}, r_i\rangle| \leq c|\rho(d^{\pi_i,\mathrm{RA}^*}) - \rho(d^{\pi_i,\mathrm{RN}^*})|, \tag{46}$$

$$|\rho(d^{\pi_i,\mathrm{RA}^*}) - \rho(d^{\pi_i,\mathrm{RN}^*})| \leq L\|d^{\pi_i,\mathrm{RA}^*} - d^{\pi_i,\mathrm{RN}^*}\|_1, \tag{47}$$

$$|\langle d^{\pi_i,\mathrm{RN}^*}, r_i\rangle - \langle d^{\pi_i,\mathrm{RA}^*}, r_i\rangle| \leq L \cdot c \cdot \|d^{\pi_i,\mathrm{RA}^*} - d^{\pi_i,\mathrm{RN}^*}\|_1. \tag{48}$$

Step 5: Bounding the L1-norm Following C.3

$$\|d^{\pi_i,\text{RA}^*} - d^{\pi_i,\text{RN}^*}\|_1 \leq 2 \tag{49}$$

Step 5: Final bound on the action-value function difference Integrating these observations into (42):

$$|Q_i^{\pi_i,\text{RA}^*}(s,a) - Q_i^{\pi,\text{RN}}(s,a)| \leq 2 \cdot \gamma \cdot L \cdot c. \tag{50}$$

□

Lemma C.6.

$$|\tilde{Q}_i^{\pi_i,RA^*}(s,a) - \tilde{Q}_i^{\pi_j,RN^*}(s,a)| \leq 2\left(\frac{1}{1-\gamma}\|r_i - r_j\|_\infty + 2 \cdot L \cdot c\right). \tag{51}$$

Proof. We aim to establish the bound on the difference between the modified action-value functions $\tilde{Q}_i^{\pi_i,\text{RA}^*}$ and $\tilde{Q}_i^{\pi_j,\text{RN}^*}$ for state-action pairs (s,a).

Step 1: Applying definition of $\tilde{Q}$ and triangle inequality We start by applying the definition of the modified action-value functions and the triangle inequality:

$$\begin{aligned}|\tilde{Q}_i^{\pi_i,\text{RA}^*}(s,a) - \tilde{Q}_i^{\pi_j,\text{RN}^*}(s,a)| &= |Q_i^{\pi_i,\text{RA}^*}(s,a) - c\rho(d^{\pi_i,\text{RA}^*}) - Q_i^{\pi_j,\text{RN}^*}(s,a) + c\rho(d^{\pi_j,\text{RN}^*})| \\ &\leq |Q_i^{\pi_i,\text{RA}^*}(s,a) - Q_i^{\pi_j,\text{RN}^*}(s,a)| + c|\rho(d^{\pi_i,\text{RA}^*}) - \rho(d^{\pi_j,\text{RN}^*})|.\end{aligned}$$

Step 2: Applying the triangle inequality to the first term Adding and subtracting $Q_i^{\pi_i,\text{RN}^*}(s,a)$ and $Q_j^{\pi_j,\text{RN}^*}(s,a)$ to decompose the term:

$$\begin{aligned}|Q_i^{\pi_i,\text{RA}^*}(s,a) - Q_i^{\pi_j,\text{RN}^*}(s,a)| \leq{} & |Q_i^{\pi_i,\text{RA}^*}(s,a) - Q_i^{\pi_i,\text{RN}^*}(s,a)| + |Q_i^{\pi_i,\text{RN}^*}(s,a) - Q_j^{\pi_j,\text{RN}}(s,a)| \\ & + |Q_j^{\pi_j,\text{RN}^*}(s,a) - Q_i^{\pi_j,\text{RN}^*}(s,a)|.\end{aligned}$$

Step 3: Bounding the terms using referenced lemmas From Lemmas C.1, C.2, C.5:

$$|Q_i^{\pi_i,\text{RA}^*}(s,a) - Q_i^{\pi_j,\text{RN}^*}(s,a)| \leq \frac{2}{1-\gamma}\|\text{r}_i - \text{r}_j\|_\infty + 2 \cdot \gamma \cdot L \cdot c.$$

Step 4: Bounding the second term using Lemma C.4 From the established bound on the difference in risk-aware and risk-neutral policies:

$$c|\rho(d^{\pi_i,\text{RA}^*}) - \rho(d^{\pi_j,\text{RN}^*})| \leq 2 \cdot L \cdot c.$$

By summing up these inequalities, we derive the final result:

$$|\tilde{Q}_i^{\pi_i,\text{RA}^*}(s,a) - \tilde{Q}_i^{\pi_j,\text{RN}^*}(s,a)| \leq \frac{2}{1-\gamma}\|\text{r}_i - \text{r}_j\|_\infty + 4 \cdot L \cdot c,$$

thereby concluding the proof. □

Definition C.7 (Bellman Operator of Policy π)**.** Let Q be a (possibly inaccurate) state-action value function and π a policy. The Bellman operator applied to Q under policy π, denoted by T^π, is defined as:

$$T^\pi Q(s,a) = r(s,a) + \gamma \sum_{s'} p(s'|s,a)Q(s',\pi(s')). \tag{52}$$

Properties:

- Given $Q(s,a)$, $(T^\pi)^2 Q(s,a) = T^\pi(T^\pi(Q(s,a)))$.

- $(T^\pi)^\infty Q(s,a) = Q^\pi(s,a)$, where $Q^\pi(s,a)$ is the state-action value function under policy π.
- $Q^\pi(s,a)$ is the fixed point under the Bellman operator of π: $T^\pi(Q^\pi(s,a)) = Q^\pi(s,a)$.
- Monotonicity of the Bellman operator: if $Q_1(s,a) \geq Q_2(s,a)$, then $T^\pi Q_1(s,a) \geq T^\pi Q_2(s,a)$.
- It follows that if $T^\pi(Q^\pi(s,a)) \geq Q^\pi(s,a)$, then $(T^\pi)^2(Q^\pi(s,a)) \geq (T^\pi)Q^\pi(s,a)$.

Policy definition The policy π_i:

$$\pi_i(a|s) = \begin{cases} 1 & \text{if } a = \arg\max_b \max_{j=1,2,\ldots,n} \tilde{Q}_i^{\pi_j^*}(s,b) \\ 0 & \text{otherwise} \end{cases} \tag{53}$$

for simplicity,

$$\pi_i(s) \in undersetb \arg\max \max_{j=1,2,\ldots,n} \tilde{Q}_i^{\pi_j^*}(s,b). \tag{54}$$

$\mathbf{Q}_{\max}$ definition let $(Q_{\max}(s,a)$ be defined as:

$$Q_{\max}(s,a) = \max_j \left(Q^{\pi_j}(s,a) - c\rho(d^{\pi_j}) \right), \tag{55}$$

Proposition C.8. *Let $Q_{max}(s,a)$ be an initial estimate of the action-value function. Then, the application of the infinite Bellman operator $(T^{\pi_i})^\infty$ to $Q_{max}(s,a)$ converges to $Q^{\pi_i}(s,a)$, the true action-value function under policy π_i.*

$$(T^{\pi_i})^\infty Q_{max}(s,a) = Q^{\pi_i}(s,a) \tag{56}$$

Proof. This result follows from the definition of the Bellman operator (see Definition C.7), asserting that the iterative application of T^{π_i} to any initial function eventually converges to the fixed point of T^{π_i}, which is $Q^{\pi_i}(s,a)$. $\square$

Lemma C.9. *Assuming $|\rho(d_s^\pi) - \rho(d_{s'}^\pi)| \approx 0$ where s' follows s in the trajectory, it holds that:*

$$T^{\pi_i} Q_{max}(s,a) \geq Q_{max}(s,a).$$

Proof. **Step 1: Definition of the Bellman operator** Refer to Equation (52):

$$T^{\pi_i} Q_{\max}(s,a) = r(s,a) + \gamma \sum_{s'} p(s'|s,a) \left[Q_{\max}(s', \pi_i(s')) \right]. \tag{57}$$

Step 2: Definition of π_i From Equation (54):

$$T^{\pi_i} Q_{\max}(s,a) = r(s,a) + \gamma \sum_{s'} p(s'|s,a) \left[\max_b Q_{\max}(s',b) \right]. \tag{58}$$

Step 3: Property of max, for a particular case of π_j^{**}:

$$T^{\pi_i} Q_{\max}(s,a) \geq r(s,a) + \gamma \sum_{s'} p(s'|s,a) \left[Q_{\max}(s', \pi_j, \mathrm{RN}^*(s')) \right]. \tag{59}$$

Step 4: Definition of $Q_{\max}$:

$$T^{\pi_i} Q_{\max}(s,a) \geq r(s,a) + \gamma \sum_{s'} p(s'|s,a) \left[\max_k Q^{\pi_k, \mathrm{RN}^*}(s', \pi_j, \mathrm{RN}^*(s')) - c\rho(d_{s'}^{\pi_k, \mathrm{RN}^*}) \right]. \tag{60}$$

Step 5: Property of max, for a particular case of π_j^{**}:

$$T^{\pi_i} Q_{\max}(s,a) \geq r(s,a) + \gamma \sum_{s'} p(s'|s,a) \left[Q^{\pi_j^*}(s', \pi_j, \text{RN}^*(s')) - c\rho(d_{s'}^{\pi_j, \text{RN}^*}) \right]. \quad (61)$$

Step 6: Expanding the expression:

$$T^{\pi_i} Q_{\max}(s,a) \geq r(s,a) + \gamma \sum_{s'} p(s'|s,a) \left[Q^{\pi_j, \text{RN}^*}(s', \pi_j, \text{RN}^*(s')) \right] - \gamma \sum_{s'} p(s'|s,a) c\rho(d_{s'}^{\pi_j, \text{RN}^*}). \quad (62)$$

Step 7: Applying the practical assumption that $|\rho(d_s^\pi) - \rho(d_{s'}^\pi)| \approx 0$:

$$T^{\pi_i} Q_{\max}(s,a) \geq r(s,a) + \gamma \sum_{s'} p(s'|s,a) \left[Q^{\pi_j, \text{RN}^*}(s', \pi_j, \text{RN}^*(s')) \right] - \gamma c\rho(d_s^{\pi_j, \text{RN}^*}). \quad (63)$$

since $\sum_{s'} p(s'|s,a) = 1$

Step 8: Bellman operator definition for π_j, RN*:

$$r(s,a) + \gamma \sum_{s'} p(s'|s,a) \left[Q_{\text{RN}}^{\pi_j, \text{RN}^*}(s', \pi_j, \text{RN}^*(s')) \right] = T^{\pi_j, \text{RN}^*} Q^{\pi_j, \text{RN}^*}(s,a). \quad (64)$$

Step 9: Bellman operator property fixed point for π_j, RN*:

$$T^{\pi_j, \text{RN}^*}(Q^{\pi_j, \text{RN}^*}(s,a)) = Q^{\pi_j, \text{RN}^*}(s,a). \quad (65)$$

Step 10: Substituting:

$$r(s,a) + \gamma \sum_{s'} p(s'|s,a) \left[Q_{\text{RN}}^{\pi_j, \text{RN}^*}(s', \pi_j, \text{RN}^*(s')) \right] = Q^{\pi_j, \text{RN}^*}(s,a). \quad (66)$$

Step 11: Concluding:

$$T^{\pi_i} Q_{\max}(s,a) \geq Q^{\pi_j, \text{RN}^*}(s,a) - \gamma c\rho(d_s^{\pi_j, \text{RN}^*}). \quad (67)$$

Step 12: Removing γ: Since $\gamma \leq 1$

$$T^{\pi_i} Q_{\max}(s,a) \geq Q^{\pi_j, \text{RN}^*}(s,a) - c\rho(d_s^{\pi_j, \text{RN}^*}). \quad (68)$$

Step 13: Applying the definition of $Q_{\max}$: Since the above holds $\forall j$, then it holds for the maximum

$$T^{\pi_i} Q_{\max}(s,a) \geq Q_{\max}(s,a). \quad (69)$$

$\square$

Lemma C.10. *The true action-value function Q^{π_i} under policy π_i satisfies the following inequality for all j:*

$$Q^{\pi_i}(s,a) \geq Q^{\pi_j, RN^*}(s,a) - c\rho(d^{\pi_j, RN^*}). \quad (70)$$

Proof. **Step 1:** From Lemma C.9, we have that the Bellman operator applied to $Q_{\max}$ satisfies:

$$T^{\pi_i} Q_{\max}(s,a) \geq Q_{\max}(s,a). \quad (71)$$

Step 2: From the monotonicity property of the Bellman operator:

$$(T^{\pi_i})^2 Q_{\max}(s,a) \geq (T^{\pi_i}) Q_{\max}(s,a) \geq Q_{\max}(s,a). \quad (72)$$

Step 3: Applying the Bellman operator infinitely many times:

$$(T^{\pi_i})^\infty Q_{\max}(s,a) \geq Q_{\max}(s,a). \quad (73)$$

Step 4: Applying Proposition C.8 that states the convergence to $Q^{\pi_i}(s,a)$:

$$Q^{\pi_i}(s,a) \geq Q_{\max}(s,a). \tag{74}$$

Step 5: Applying the property of max in $Q_{\max}$:

$$Q^{\pi_i}(s,a) \geq Q^{\pi_j,\text{RN}^*}(s,a) - c\rho(d^{\pi_j,\text{RN}^*}). \tag{75}$$

□

Theorem C.1. *Let $M_i \in \mathcal{M}$ and let $Q_i^{\pi_j^*}$ be the action-value function of an optimal (risk-aware or risk-neutral) policy π_j^* of $M_j \in \mathcal{M}$ when evaluated in M_i, and let $\rho_i(d^{\pi_j^*})$ be an L-Lipschitz caution factor of π_j^* in M_i, bounded by a constant K, i.e, $|\rho_i(d)| \leq K$. Further, let $\tilde{Q}_i^{\pi_j^*}(s,a) = Q_i^{\pi_j^*}(s,a) - c \cdot \rho_i(d^{\pi_j^*})$.*

$$\textit{Let } \pi_i(a|s) = \begin{cases} 1 & \textit{if } a = \arg\max_b \max_{j=1,2,\ldots,n} \tilde{Q}_i^{\pi_j^*}(s,a) \\ 0 & \textit{otherwise}. \end{cases} \tag{76}$$

$$\tilde{Q}_i^{\pi_i^*}(s,a) - \tilde{Q}_i^{\pi_i}(s,a) \leq min_j \left(\frac{2}{1-\gamma} \|r_i - r_j\|_\infty + (4 \cdot L + K) \cdot c \right). \tag{77}$$

Proof. **Step 1: Notation.** Define the modified action-value function for the optimal policy as:

$$\tilde{Q}_i^{\pi_i^*}(s,a) = \tilde{Q}_i^{\pi_i,\text{RA}^*}(s,a),$$

and then, the difference between the optimal and another policy is:

$$\tilde{Q}_i^{\pi_i^*}(s,a) - \tilde{Q}_i^{\pi_i}(s,a) = \tilde{Q}_i^{\pi_i,\text{RA}^*}(s,a) - \tilde{Q}_i^{\pi_i}(s,a).$$

Step 2: Appyling Lemma C.10.

$$Q^{\pi_i}(s,a) \geq \max_j \left[Q_i^{\pi_j,\text{RN}^*}(s,a) - c\rho(d^{\pi_j,\text{RN}^*}) \right],$$

which equivalently means:

$$-Q_i^{\pi_i}(s,a) \leq -\min_j \tilde{Q}_i^{\pi_j,\text{RN}^*}(s,a) \quad \forall M_j.$$

Step 3: Substituting into inequality. From the definitions above, the difference can be rewritten and bounded as:

$$\tilde{Q}_i^{\pi_i,\text{RA}^*}(s,a) - \tilde{Q}_i^{\pi_i}(s,a) \leq \min_j \left(\tilde{Q}_i^{\pi_i,\text{RA}^*}(s,a) - \tilde{Q}_i^{\pi_j,\text{RN}^*}(s,a) \right) + c \cdot \rho(d^{\pi_i^*}) \quad \forall M_j.$$

Step 4: Using Lemma C.6.

$$\tilde{Q}_i^{\pi_i,\text{RA}^*}(s,a) - \tilde{Q}_i^{\pi_i}(s,a) \leq 2 \cdot \min_j \left(\frac{1}{1-\gamma} \|\mathrm{r}_i - \mathrm{r}_j\|_\infty + 2 \cdot L \cdot c \right) + c \cdot \rho(d^{\pi_i^*}) \quad \forall M_j.$$

Step 5: Using the boundedness of $\rho(d)$.

$$\tilde{Q}_i^{\pi_i,\text{RA}^*}(s,a) - \tilde{Q}_i^{\pi_i}(s,a) \leq 2 \cdot \min_j \left(\frac{1}{1-\gamma} \|\mathrm{r}_i - \mathrm{r}_j\|_\infty + 2 \cdot L \cdot c \right) + c \cdot K \quad \forall M_j.$$

□

Corollary C.2. *Under the same assumptions as Theorem C.1, and letting $\phi_{\max} = \max_{s,a,s'} \|\phi(s,a,s')\|$, we have that:*

$$\tilde{Q}_i^{\pi_i^*}(s,a) - \tilde{Q}_i^{\pi_i}(s,a) \leq \min_j \left(\frac{2}{1-\gamma} \phi_{\max} \|\mathbf{w}_i - \mathbf{w}_j\| + (4 \cdot L + K) \cdot c \right) \tag{78}$$

Proof. Using the reward decomposition and the Cauchy-Schwarz inequality, we establish that:

$$\|r_i - r_j\| \leq \phi_{\max} \|\mathbf{w}_i - \mathbf{w}_j\| \tag{79}$$

□

D Limitations of Risk-Aware Test-Time Adaptation

We argue that the current state-of-the-art on risk-aware test-time adaptation (2) is limited in **three** different aspects. We focus on a Gridworld to highlight the limitations as follows.

L1: Risk-aware source policies are conservative. A critical assumption in (2) is that the source policies are trained with a variance term in the objective, i.e., the objective is a weighted sum of return and variance. We claim that adding the variance term limits the space of possible test-time policies realizable from the source policies. We illustrate this with two scenarios:

Scenario 1: The source policies are variance-aware, as in (2). In Fig. 6, the states in the upper path provide *stochastic* rewards (shown as gifts) that could be positive or negative. Both variance-aware policies avoid the upper path, resulting in two *identical* source policies.

Scenario 2: The source policies are risk-agnostic, i.e., trained to maximize expected return only. In the lower part of Fig. 6, each source task prefers a different path—one upper, one lower—based solely on return, thus yielding *diverse* policies.

Results: In the target task, the optimal path has high return and low variance. Scenario 2 leads to a better test-time policy because the diversity among risk-agnostic sources enables identifying this path. By contrast, the identical, conservative source policies in Scenario 1 fail to exploit it.

Takeaway: Training source policies in a risk-agnostic fashion, as in standard RL (32), enables better coverage and flexibility at test time.

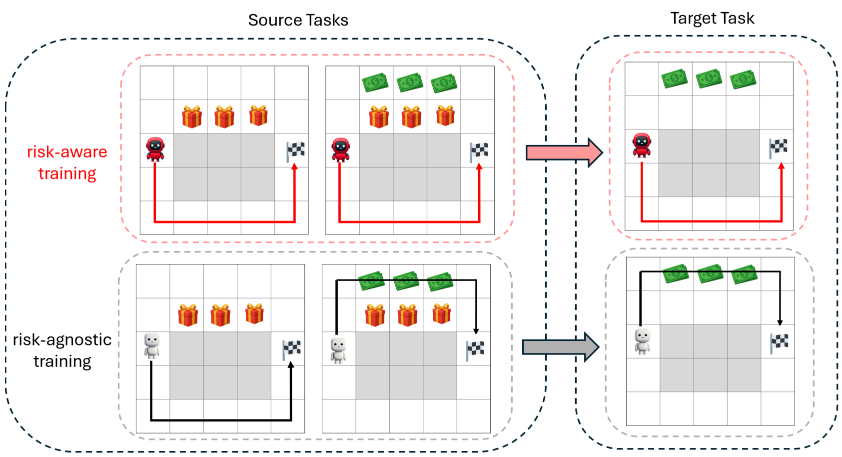

Figure 6: Illustration of adaptation where two source policies (left) are adapted to a single target task (right). **Top:** *Risk-aware training* (as in (2)) leads to identical conservative behaviors avoiding high-variance regions, resulting in poor target adaptation. **Bottom:** *Risk-agnostic training* produces diverse source policies. The adapted target policy successfully identifies the optimal upper path.

L2: The variance of the return fails in deterministic settings. Return variance is defined as:

$$\tilde{Q}^{\pi}(s,a) = \mathrm{Var}[G_t \mid S_t = s, A_t = a] = \mathbb{E}\left[(G_t - \mathbb{E}[G_t])^2 \mid S_t = s, A_t = a\right]$$

This quantity measures variability across full-trajectory rollouts. But if both the transition dynamics and policy are deterministic (37; 38), then all rollouts are identical, making return variance *zero*. In this case, the method in (2) behaves as if it were risk-agnostic and collapses to earlier adaptation frameworks like (25; 26; 29).

However, risk may still be present in other forms—e.g., **per-step reward variance**. Consider two deterministic rollouts: - (1, 3, -4, 2, -3) $\rightarrow$ High per-step variability - (2, 3, 2, 3, 2) $\rightarrow$ Low per-step variability Even though both are fixed trajectories, the first is intuitively riskier. Return variance fails to capture this.

A practical example is shown in Fig. 7. The source policies (left) are deterministic. In the target task (middle and right), the lower path either (i) has high per-step reward variance (middle) or (ii) minimizes return variance but passes through a hazardous barrier (right). In both cases, the adaptation from (2) fails to account for real risk.

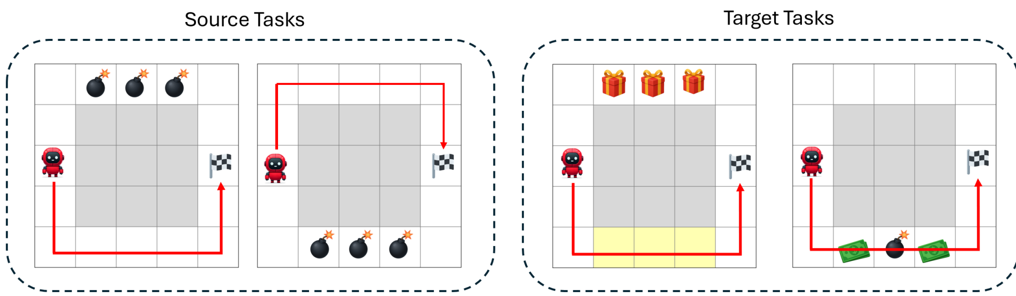

Figure 7: Unified illustration of the limitations of using return variance as a risk measure. **Left:** Source policies are deterministic and follow fixed reward sequences. **Middle:** Target task with high per-step reward variance in the lower path—ignored by return-based risk. **Right:** Target task with a hazardous barrier (yellow) on the low-variance path—again selected by the agent in (2), despite higher real-world risk.

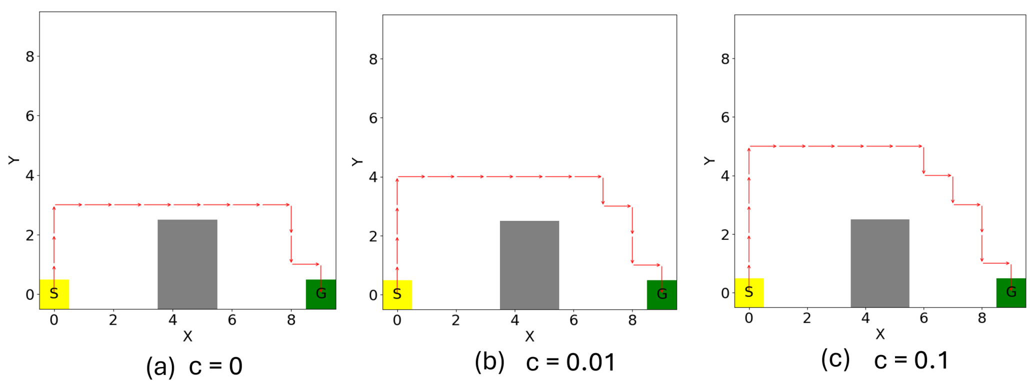

Figure 8: The performance of CAT as a function of the parameter c. As c increases, the transferred policy becomes safer. However, the discounted cumulative reward decreases, as the agent takes more steps to reach the goal.

L3: Variance is not representative of all forms of risk. Beyond per-step variability, other types of risk exist—e.g., *barrier risk*, where specific states must be avoided altogether (17). In Fig. 7 (right), the agent must choose between: - Lower path: low return variance, but passes through a danger zone (yellow) - Upper path: avoids the barrier, but has higher return variance

The method of (2) chooses the lower path, due to its narrow focus on return variance. In this case, return variance is misaligned with the true risk objective: avoiding danger.

Summary: The state of the art in risk-aware test-time adaptation is limited in three ways: (i) it assumes risk-aware source policies, which reduces diversity, (ii) it only accommodates return variance as a risk signal, and (iii) even this variance formulation fails in deterministic or structured environments. Our proposed framework, **TRAM**, addresses all three challenges, as we describe next.

E The effect of the hyperparameter c

Figure 8 shows the policy obtained for the different values of the coefficient c. The larger the c, the lower the risk, but the larger the amount of steps to reach the goal.

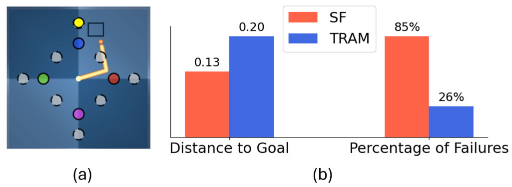

Figure 9: The Reacher domain (adapted from (25), also used in (2; 29)). (a) Setup: The two-joint robotic arm must reach one of the circled goals. Four source policies exist (blue, green, red and purple). During training, an optimal risk-neutral policy is found for each of the training tasks, by maximizing the expected return. One test task (yellow) is considered. A danger region is introduced at this test task (light blue rectangle). A failure is considered once the tip of the joint enters the danger region. The testing experiment was repeated 100 times. (b) A bar graph of the results. The failure for `CAT` is substantially less than risk-neutral transfer (25) On the other hand, the mean distance to the goal is slightly higher, as the `CAT` agent must balance between return and caution.

F Reacher

In this section, we test our algorithm on a continuous state-space transfer RL benchmark. In this process, we attempt to answer the following questions

(EQ3) Can `CAT` scale up to complex continuous domains?

(EQ4) Can `CAT` be used in conjunction with function approximation?

The Reacher domain, shown in Figure 9 (a), is a set of control tasks defined in the MuJoCo physics engine (31). Each task requires moving a two-joint torque-controlled simulated robot arm to a given target location. The Reacher domain experiment is the standard experiment in transfer reinforcement learning via successor features (25; 2; 29).

MDP modeling The experiment involves a 4-dimensional continuous state-space and 9 possible values for the action (corresponding to maximum, minimum and zero value of torque for each of the 3 dimensions). The reward is a function of the distance to the goal, and the dynamics are governed by the simulator.

Tasks Description Four source tasks have been considered, and instead of training Deep Q-Networks (DQNs) to choose the optimal action, to allow for instantaneous evaluation in the test tasks, we train Successor Feature Deep Q-Networks (SFDQNs). Therefore, each of the 4 source tasks has an associated SFDQN that carries its optimal policy. The SFDQN returns the successor feature vector for a given state and action, and if we wish to evaluate this policy in a new task, we simply perform the dot product of this SF vector with the weight vector of that task.

Training and Testing details Since the source policies are risk-neutral and have no knowledge of caution, we use the same parameters for training as in the original SF paper (25). We consider one of the test tasks in the Reacher domain and a barrier risk function with a parameter $c = 5$. Failure is defined as entering the particular barrier region in space.

Results The bar graph of Figure 9 shows the performance of `CAT` based on 100 samples, as opposed to the standard risk-neutral transfer scheme used in (25). The percentage of failure is substantially less with CAT. On the other hand, the mean distance to the goal is larger, which is expected, as `CAT` sacrifices the cumulative reward in exchange for a much higher level of safety.

G Base Code

The base code for the continuous example is taken from https://github.com/mike-gimelfarb/deep-successor-features-for-transfer/tree/main.

H CPU Resources

The specifications of the machine used are shown in Table 2.

Table 2: CPU Information Summary

Attribute	**Details**
Processor	Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz
Base Speed	1.99 GHz
Current Speed	1.57 GHz (can vary)
Cores	4
Logical Processors	8
L1 Cache	256 KB
L2 Cache	1.0 MB
L3 Cache	8.0 MB
Virtualization	Disabled
Hyper-V Support	Yes

We used large language models to write more efficient code and construct tables and some LaTeX commands, but not to write the paper.

Broader Impact

This work focuses on risk-aware inference-time transfer in reinforcement learning (RL), with the goal of improving the adaptability and safety of RL models in real-world scenarios. By incorporating a generalized notion of caution into the transfer process, this research contributes to the development of safer policies for deployment tasks, particularly in settings where direct fine-tuning is not feasible.

Potential societal benefits include improved safety and robustness in autonomous systems, such as robotics and decision-making agents, where ensuring risk-aware behavior is crucial. This work may also enhance the efficiency of RL applications in domains where unexpected risks can arise, such as healthcare, finance, and transportation.

However, as with any machine learning framework, there are potential ethical considerations. The reliance on predefined risk measures could introduce biases or limitations in identifying all possible risks in dynamic environments. Additionally, deploying RL models in safety-critical applications requires careful validation to ensure that the policies do not produce unintended harmful behaviors.

In summary, this work advances the field of reinforcement learning by enabling safer policy transfer, with broad applicability in various industries. While the societal implications are largely beneficial, careful deployment and validation remain essential to mitigating any unintended consequences.