Recursive Semantic Divergence for LLM Agent Consistency

Harshavardhan Abichandani
SAP SE
harshavardhan.abichandani@sap.com

Penny Chong
SAP SE
penny.chong@sap.com

Atin Ghosh
SAP SE
atin.ghosh@sap.com

Daniel Dahlmeier
SAP SE
d.dahlmeier@sap.com

Abstract

Large Language Model (LLM) agents with tool use can behave inconsistently across independent runs from identical states. They may call different tools, produce contradictory outputs, or follow divergent strategies. This variability compounds over time and undermines deployment reliability. Existing evaluation metrics primarily measure task correctness via completion-based scores and do not capture consistency in agent behavior across executions. Separately, agent behavior is analyzed either at the token level using probability distributions or at the single-response level using semantic clustering, but neither captures how inconsistency propagates across future turns of a trajectory. We introduce Recursive Semantic Divergence (RSD), a trajectory-level consistency metric adapted from bisimulation metrics in reinforcement learning. RSD measures the expected cumulative semantic disagreement between two independent rollouts of the same agent from the same state, using natural language inference to detect logical inconsistency. It is defined recursively over future states and approximated by a neural network trained on independent rollouts. The learned metric estimates future divergence from the current state, providing a consistency score at each turn without requiring the full trajectory to complete. When used as a reward signal for fine-tuning, RSD reduces AUC coefficient of variation (CV) by up to 30.9% on τ^2-bench and 50.5% on ToolSandbox. Across Qwen-2.5 and Qwen-3.5 models, RSD reduces AUC CV in seven out of eight model-benchmark settings, while improving the final task progress in all eight settings.[1]

1 Introduction

Recent advances have enabled large language model (LLM) agents to execute complex workflows and multi-step tasks via tool calling functions [Yao et al., 2023, Jang et al., 2025, Liu et al., 2023]. These agentic systems are increasingly deployed as autonomous agents that interact with users and tools over multiple turns. For long-horizon tasks such as code generation agents [Shinn et al., 2023, Yang et al., 2024], as well as agents built for critical applications such as healthcare systems [Tu et al., 2025, Kim et al., 2024], both correctness and reliability are important but manifest in different ways. In long- horizon tasks, like code generation, small errors in the beginning can accumulate over time, leading to compounding errors that deviate significantly from the expected solution trajectory. In critical applications such as healthcare systems, reliability is further challenged by the need for consistent agent behavior in repeated executions. However, in reality, we observe that given the same

[1] Source code is publicly available in the repository https://github.com/SAP-samples/agent-consistency-rsd

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

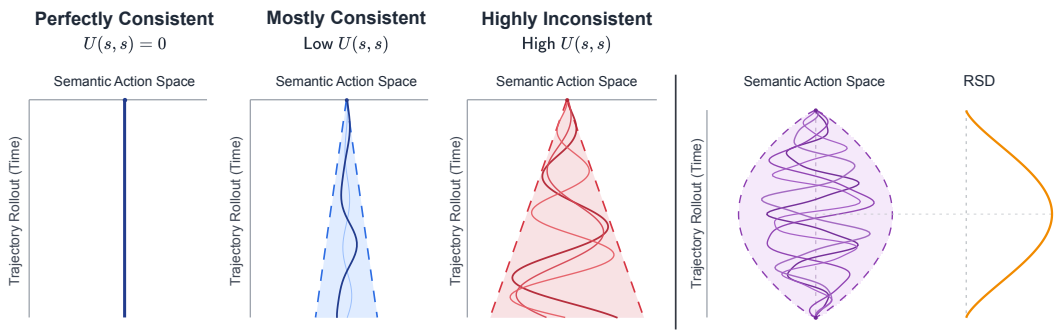


Figure 1: Diagram illustrating Recursive Semantic Divergence (RSD) in LLM agents. The first three panels display trajectory spreads for perfectly consistent, mostly consistent, and highly inconsistent agents over time. The final panel maps an agent's expanding and converging semantic action space alongside its corresponding RSD score, demonstrating how the metric tracks the variance of semantic divergence at each turn.

task to solve, the agent may still yield different decision paths across executions, such as calling entirely different tools, providing contradictory information to users, or following divergent strategies that lead to different outcomes.

While LLM decoding strategies such as setting the temperature to zero, or using greedy decoding reduce stochasticity at the token level, they do not necessarily ensure consistency at the agent-level. For a multi-step or multi-turn LLM agent, the behavior is determined by sequences of decisions conditioned on evolving trajectories, where small differences in the trajectories due to stochastic tool output responses or environmental feedback can lead to divergent agent trajectories even under deterministic decoding strategies. Additionally, even when the underlying LLM is the same, the decoding strategy is deterministic, and the agent interaction history is fixed, there are additional sources of non-determinism such as floating-point precision and hardware-level variability which can lead to inconsistent agent behavior.

This highlights that the agent behaviour, which is trajectory-dependent, may vary across executions under identical initial conditions. Existing evaluation metrics [Chang et al., 2024, Yao et al., 2024, Lu et al., 2024] primarily report task correctness using full or partial task completion metrics, but often do not capture consistency in agent performance. The $pass\hat{}k$ metric [Yao et al., 2024, Barres et al., 2025], which measures the chance that all k i.i.d. task trials are successful, is the closest existing metric to capturing agent consistency, but it remains binary and is used for reporting rather than enforcing consistency in the agent behavior. Another line of work considers either token-level distributions [Malinin and Gales, 2020] or single-response semantic similarity [Kuhn et al., 2023], but these approaches do not capture how inconsistencies propagate across future turns in a trajectory. Notably, agent consistency is independent of correctness, as agents may exhibit stable behavior patterns that are either consistently successful or consistently failing across executions.

Motivated by the need for a reliable and consistent agent behavior in long-horizon tasks and critical applications, and the trajectory-dependent variability of LLM agents that is not captured by existing evaluation metrics, we propose a trajectory-level consistency metric to enforce agent consistency conditioned on identical interaction-history states. We introduce Recursive Semantic Divergence (RSD), an unsupervised trajectory-level consistency metric adapted from bisimulation metrics [Ferns et al., 2004] in reinforcement learning, which define state similarity in terms of behavioral equivalence. We extend this framework to interaction-history states and replace reward differences with a semantic discrepancy based on Natural Language Inference (NLI) [Bowman et al., 2015], adapted to LLM agents' actions. We then define a recursive bisimulation-style objective over this semantic distance, yielding a fixed-point formulation of behavioral consistency, which is approximated using the Matching under Independent Coupling (MICo) method [Castro et al., 2021]. We show that training with RSD metric as a reward signal improves agent consistency.

We summarize our contribution as follows:

1. We introduce Recursive Semantic Divergence (RSD), an unsupervised trajectory-level consistency metric that adapts bisimulation from reinforcement learning to evaluate LLM agents.
2. We empirically demonstrate that using the learned RSD metric as a reward signal for fine-tuning significantly reduces behavioral variation and improves consistency across benchmarks and model scales.
3. We provide a theoretical analysis showing that the RSD fixed-point formulation converges and yields meaningful guarantees about agent behavior. In particular, we derive bounds on the probability of total trajectory disagreement and prove the existence of a dominant action class when the agent exhibits low RSD.

2 Related Work

Agent evaluation. Existing agent evaluation methods primarily focus on task correctness or completion rates. Wang et al. [2023], Xie et al. [2024] evaluate coarse-grained agent performance using success-based metrics, while Chang et al. [2024], Chong et al. [2026] report fine-grained metrics that capture intermediate progress and turn-level efficiency. Representing the tool call dependencies of an agent with a directed acyclic graph (DAG) [Lu et al., 2024], is also used to evaluate agent performance but it does not assess agent consistency across multiple executions. Although some prior works partially account for agent stochasticity when reporting final performance metrics, they do not explicitly measure consistency in agent behavior or propose methods to enforce consistent execution of agent trajectories. For instance, Yao et al. [2024], Barres et al. [2025], measure agent reliability with the $pass\hat{}k$ metric which quantifies the chance that all k i.i.d. task trials are successful out of the n total trials, but the metric remains limited for binary task outcomes only and not for evaluating the consistency of intermediate progress. Similarly, Chong et al. [2026], analyse the agent consistency across the different executions by visualising expected agent progress and variance in a plot, but does not provide method to improve agent consistency, which is the primary goal of our work.

Uncertainty estimation. Quantifying uncertainty in LLM outputs has been approached from multiple angles. Malinin and Gales [2020] measure token-level entropy, but such methods require access to model logits or weights and fail to account for the semantic content of the output. Kuhn et al. [2023] introduced semantic entropy, in which sampled responses are semantically clustered before entropy is computed. While this addresses the limitations of token-level measures, it operates at the level of single-query responses and does not capture how disagreement at one turn propagates across a multi-turn trajectory.

For multi-step agents specifically, Kirchhof et al. [2025] argue that uncertainty quantification methods require fundamental reassessment due to underspecification and environmental effects introduced during agent interaction. Duan et al. [2025] propose an information-theoretic framework that decomposes uncertainty propagation across agent steps into intrinsic and extrinsic components. However, their approach is primarily diagnostic, requires the agent to complete the full trajectory before analysis, and does not produce a single scalar score that can be directly used as a reward signal. In contrast, RSD is a lookahead, unsupervised metric that captures how semantic inconsistency propagates across future agent turns.

3 Preliminaries

3.1 Bisimulation Metrics

Bisimulation metrics [Larsen and Skou, 1991, Ferns et al., 2004] were introduced in reinforcement learning to measure the behavioral distance between MDP states. Two states are bisimilar if and only if they receive the same immediate reward and they transition to states that are themselves bisimilar. This recursive definition yields a fixed-point equation. Following Ferns et al. [2004], for an MDP with reward R, transition kernel P, and discount factor γ, the bisimulation distance d^* is the unique fixed point of the operator $\mathcal{T}$ acting on the space of all state-pair distance functions $\mathcal{D}$. We define $\mathcal{T} : \mathcal{D} \to \mathcal{D}$, $d \mapsto \mathcal{T}d$, with pointwise form

$$(\mathcal{T}d)(s^{(i)}, s^{(j)}) = |R(s^{(i)}) - R(s^{(j)})| + \gamma \cdot W(d)\big(P(\cdot \mid s^{(i)}, \pi),\ P(\cdot \mid s^{(j)}, \pi)\big), \tag{1}$$

and the bisimulation distance is given by the fixed point $d^* = \mathcal{T}d^*$, where $W(d)(\cdot,\cdot)$ denotes the Kantorovich distance between the two successor-state distributions $P(\cdot \mid s^{(i)}, \pi)$ and $P(\cdot \mid s^{(j)}, \pi)$, under the common policy π, using the ground metric d. States that are close under d^* produce similar reward sequences and similar future trajectories, regardless of how different they appear in observation space.

3.2 MICo: Sampling-Based Bisimulation

Computing the exact fixed point of Equation (1) requires evaluating the Kantorovich distance over successor state distributions, which is intractable when these distributions are accessible only via samples. The Matching under Independent Coupling (MICo) method [Castro et al., 2021] addresses this by replacing the intractable computation with sample-based coupling: rather than computing $W(d)(P(\cdot|s^{(i)}, \pi), P(\cdot|s^{(j)}, \pi))$ over the full distribution, it pairs individual successor samples $(s'^{(i)}, s'^{(j)})$ drawn independently from their respective transition distributions. The key insight from Castro et al. [2021] is that training on independently coupled samples still converges to a distance function that reflects behavioral similarity.

MICo parameterizes the distance through a learned embedding $\phi_\theta : \mathbb{R}^d \to \mathbb{R}^m$ that maps state representations to a latent space. The distance between two states combines a norm component (capturing individual state "spread") and an angular component (capturing relative orientation):

$$\hat{U}_\theta(s^{(i)}, s^{(j)}) = \alpha\big(\|\phi_\theta(z^{s^{(i)}})\|^2 + \|\phi_\theta(z^{s^{(j)}})\|^2\big) + \beta \cdot d_\angle\big(\phi_\theta(z^{s^{(i)}}),\ \phi_\theta(z^{s^{(j)}})\big), \tag{2}$$

where $z^s \in \mathbb{R}^d$ is a pretrained text embedding of state s, $d_\angle$ is the angular distance (normalized arccos of cosine similarity), and $\alpha, \beta > 0$ are fixed coefficients.

4 Methodology

4.1 Problem Formulation

We follow existing methods [Yao et al., 2023, 2024, Huang et al., 2025] of formulating the LLM agent environment as a Partially Observable Markov Decision Process (POMDP) [Kaelbling et al., 1998]:

$$\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{O}, T, R \rangle, \tag{3}$$

where $\mathcal{S}$ is the set of true environment states such as database contents and user preferences, $\mathcal{A}$ is the agent's action space which includes the agent responses and tool calls, and $\mathcal{O}$ is the observation space which includes all feedback the agent receives from the environment such as user utterance and tool output. The transition function $T : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ governs how the environment changes after the agent takes an action. The reward function $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ defines the reward the agent receives for executing a particular action in that state. This reward may be difficult to obtain without ground truth or human feedback.

In a standard setting, LLM agents never observe state $s \in \mathcal{S}$ directly, due to partial observability, context limitations, and potential access restrictions. Instead, the agent conditions on the interaction history, which includes the complete sequence of observations received $o_0, o_1, ... \in \mathcal{O}$ and actions taken $a_0, a_1, ... \in \mathcal{A}$:

$$h_t = (o_0, a_0, o_1, a_1, \ldots, o_{t-1}, a_{t-1}, o_t). \tag{4}$$

We reduce this POMDP to an MDP over belief states. For LLM agents the model's policy is already a function of the conversation history. Therefore, the history h_t serves as the belief state, and we define the induced MDP $\mathcal{M}' = \langle \mathcal{H}, \mathcal{A}, T' \rangle$ where $\mathcal{H}$ is the set of all reachable histories and the transitions T' are deterministic concatenation of the observations: $h_{t+1} = h_t \oplus a_t \oplus o_{t+1}$. We denote a state in this induced MDP as $s_t \equiv h_t$.

The underlying LLM induces a stochastic policy $\pi(a \mid s)$ over this MDP, which results in a distribution over future trajectories $\tau \sim \Pi(\cdot \mid s)$. This means that two independent samples from this distribution may produce semantically different sequences. We refer to this semantic spread of sampled trajectories from the same state as inconsistency. Lower spread indicates that the policy is more deterministic.

4.2 Recursive Semantic Divergence

The bisimulation framework (Section 3.1) supplies a recursive, discounted measure of behavioral similarity that captures how present differences propagate into the future. This notion maps directly onto agent consistency: if two independent runs from the same state yield similar behavioral futures, the agent is consistent; if they yield divergent futures, it is not.

In the absence of human feedback or ground truth labels for LLM agents, we replace the immediate reward difference in a standard MDP, $|R(s^{(i)}) - R(s^{(j)})|$ from Equation (1) with a Natural Language Inference (NLI)-based distance function. We adopt NLI [Bowman et al., 2015, Williams et al., 2018], which classifies sentence pairs as entailment, contradiction, or neutral and use it as an unsupervised measure of semantic agreement between agent actions, where bidirectional entailment is interpreted as equivalence, contradiction as disagreement, and neutrality as partial overlap. We define the following NLI distance function:

Definition 1 (NLI Distance Function)**.** *For agent actions $a^{(i)}, a^{(j)} \in \mathcal{A}$:*

$$d_{NLI}(a^{(i)}, a^{(j)}) = \begin{cases} 0 & \textit{if bidirectional entailment.} \\ 1 & \textit{if contradiction in either direction.} \\ 0.5 & \textit{otherwise (neutral).} \end{cases} \tag{5}$$

This function is symmetric, identity-preserving ($d_{\text{NLI}}(a, a) = 0$), and bounded in $[0, 1]$.For tool-calling actions, entailment indicates functionally equivalent actions, such as calling the same tool with equivalent arguments, while contradiction indicates incompatible actions, such as calling the same tool with different arguments that lead to different outcomes. Neutral captures actions whose functional equivalence is ambiguous. Considering two independently sampled trajectories, where t denotes the transition depth along each trajectory, and substituting d_{NLI} for the immediate reward difference in Equation (1), we obtain the following recursive distance:

$$U(s_t^{(i)}, s_t^{(j)}) = \mathbb{E}_{a_t^{(i)} \sim \pi(\cdot|s_t^{(i)}),\ a_t^{(j)} \sim \pi(\cdot|s_t^{(j)})} \big[d_{\text{NLI}}(a_t^{(i)}, a_t^{(j)}) \big] + \gamma \cdot \mathbb{E}\big[U(s_{t+1}^{(i)}, s_{t+1}^{(j)}) \big], \tag{6}$$

where $s_{t+1}^{(i)} \sim P(\cdot|s_t^{(i)}, a_t^{(i)})$ and $s_{t+1}^{(j)} \sim P(\cdot|s_t^{(j)}, a_t^{(j)})$ are successor states.

4.3 Learned Approximation

Computing the exact fixed point of Equation (6) is intractable for two reasons. First, we treat the LLM as a black box and do not have access to the internal transition probabilities $P(s_{t+1} \mid s_t, a_t)$. Second, even with access, the state space (all possible interaction histories) is computationally large.

To address this, we approximate the fixed point of Equation (6) using the MICo method (Section 3.2) with temporal-difference (TD) learning using the rollouts from the LLM agent. Given paired trajectory rollouts $\tau^{(i)}$ and $\tau^{(j)}$ from the same task, we pair the state-action transitions $(s_t^{(i)}, a_t^{(i)})$ and $(s_t^{(j)}, a_t^{(j)})$ at the same rollout depth t. The shared index denotes the transition depth along each rollout. After divergence, each trajectory independently follows its own successor states. The n-step TD target y_t is constructed by unrolling both trajectories for the next n steps:

$$y_t = \sum_{k=0}^{n-1} \gamma^k d_{\text{NLI}}(a_{t+k}^{(i)}, a_{t+k}^{(j)}) + \gamma^n \hat{U}_{\bar{\theta}}(s_{t+n}^{(i)}, s_{t+n}^{(j)}), \tag{7}$$

where $\bar{\theta}$ is a copy of the network parameters that is soft-updated at each step with θ and γ is the discount factor. If either trajectory terminates within the n-step window, the TD target accumulates disagreement only over the valid paired transitions, and the bootstrap term is omitted. No padding or actions beyond termination are introduced. The target corresponds to an n-step unrolling of the Bellman equation, where discrepancies are accumulated and exponentially discounted with the time step. The network is trained by minimizing the following Bellman residual:

$$\mathcal{L}(\theta) = \mathbb{E}\left[\left(\hat{U}_\theta(s_t^{(i)}, s_t^{(j)}) - y_t \right)^2 \right]. \tag{8}$$

4.4 Self-Distance as a Measure of Consistency

The MICo distance function from Equation (2) produces what Castro et al. [2021] term a *diffuse* distance—the self-distance $U(s, s)$ can be non-zero. We use this property to measure how stochastic the process is from a given state. If the transition dynamics from state s have many possible successor states, then $U(s, s) > 0$. For deterministic states, $U(s, s) = 0$. Intuitively, it captures how wide the tree of possibilities is when unrolling the process from that state.

For the self-distance $\hat{U}_\theta(s, s)$, the angular term vanishes (a vector has zero angle with itself), leaving:

$$\hat{U}_\theta(s, s) = 2\alpha\|\phi_\theta(z^s)\|^2. \tag{9}$$

In our formulation, stochasticity arises from the LLM policy $\pi(\cdot \mid s)$, which may sample different actions across runs. Since we replace the reward difference with NLI-based semantic disagreement, the self-distance $U(s, s)$ captures the expected cumulative *semantic* divergence between two independent trajectories originating from the same conversational state. It is zero when the agent always produces semantically equivalent actions (regardless of surface variation in phrasing), and positive when the agent exhibits semantic disagreement across independent runs.

Definition 2 (Recursive Semantic Divergence)**.** *The Recursive Semantic Divergence at state s is defined as the infinite-horizon discounted sum of stepwise discrepancies obtained by expanding Equation* (6)*:*

$$U(s, s) = \mathbb{E}_{\tau,\tau' \sim \Pi(\cdot|s)} \left[\sum_{t=0}^{\infty} \gamma^t \cdot d_{NLI}(a_t, a'_t) \right], \tag{10}$$

where τ and τ' are independent trajectory rollouts from the distribution of trajectories $\Pi(\cdot|s)$, approximated in practice by $\hat{U}_\theta(s, s) = 2\alpha\|\phi_\theta(z_s)\|^2$.

RSD provides a notion of how consistent an LLM agent is. Consider a state s from which the agent's policy $\pi(\cdot|s)$ places all probability mass on a single semantic equivalence class of actions (e.g., it always calls the same tool with the same arguments, possibly with different phrasings). In this case, $d_{\text{NLI}}(a, a') = 0$ for any two independent samples, and by the Bellman structure, if all reachable successor states also exhibit this concentration, then $U(s, s) = 0$. The agent is perfectly consistent from state s onward.

The complete RSD training algorithm is detailed in Algorithm 1 in Appendix A.1. For inference, the computation of the consistency score requires only a single forward pass, with no additional rollouts, NLI evaluations, or sampling. The Appendix A.1 also details the algorithm for RSD inference in Algorithm 2.

5 Theoretical Analysis

Our formulation replaces the reward signal with NLI classifications and operates on compressed state embeddings rather than full conversation histories. We need to establish two things: first, that the recursive equation converges to a well-defined quantity, and second, that this quantity provides meaningful guarantees about the agent's actual behavior.

The Bellman operator in Equation (6) is a γ-contraction in supremum norm, so a unique fixed point U^* exists with $U^* \in [0, 1/(1-\gamma)]$. Refer to Section A.2

5.1 Disagreement Probability

In this section we interpret the $U(s, s)$ score and make conclusions about the probability that two independent runs of the agent disagree. We first show a bound on the total trajectory level disagreement and a bound on the existence of a dominant behavioral mode.

Theorem 1 (Trajectory Agreement)**.** *Let $D(\tau^{(i)}, \tau^{(j)}) = \sum_{t=0}^{\infty} \gamma^t d_{NLI}(a_t^{(i)}, a_t^{(j)})$ be the total discounted disagreement between two independent rollouts from state s. For any threshold $\delta > 0$:*

$$\Pr\left(D(\tau^{(i)}, \tau^{(j)}) \geq \delta\right) \leq \frac{U(s, s)}{\delta}. \tag{11}$$

Theorem 2 (Dominant Action Class)**.** *If $U(s,s) < 1/4$, there exists a semantic equivalence class $[C^*]$ such that:*

$$\Pr(\pi(s) \in [C^*]) \geq 1 - 2U(s,s). \tag{12}$$

Since all disagreement terms in Definition 2 are non-negative, the expected immediate disagreement $D_0(s) = \mathbb{E}_{a,a' \sim \pi(\cdot|s)}[d_{\text{NLI}}(a,a')]$ satisfies $D_0(s) \leq U(s,s)$. Thus, a low trajectory-level RSD also bounds immediate action disagreement. This guarantee assumes that bidirectional entailment induces well-defined semantic equivalence classes. In practice, imperfect NLI classifications may violate this assumption, and we analyze the effect of approximation errors in Section A.2.

Together, we can interpret the results as, if $U(s,s) = 0.1$, the probability that two independent actions sampled from state s contradict each other is at most 10%, and at least 80% of the policy's mass is concentrated on a single semantic action class.

The proofs for Theorem 1 and Theorem 2, and other theorems are detailed in Appendix A.2.

Table 1: Performance and consistency, measured by the coefficient of variation (CV), across different fine-tuning strategies. The upwards and downwards arrows ↑↓ show relative change from the baseline. The blue downwards arrow ↓ indicates reduction in variation, while the red upwards arrow ↑ indicates increase in variation.

Model	**AUC**	**AUC CV**	**Progress**	**Progress CV**	$pass@1$	$pass\hat{}3$
ToolSandbox Qwen-2.5-3B						
Base Qwen (baseline)	0.271	0.90	0.293	0.86	14.1%	20.0%
Token Entropy	0.302	0.50↓43.8%	0.317	0.53↓37.8%	15.6%	20.0%
Semantic Entropy	0.326	0.83↓7.0%	0.354	0.82↓4.0%	17.2%	20.0%
SFT	0.311	0.82↓8.9%	0.327	0.81↓5.6%	17.2%	20.0%
RSD	**0.340**	**0.44**↓50.5%	0.362	**0.43**↓49.3%	18.8%	21.4%
ToolSandbox Qwen-2.5-7B						
Base Qwen (baseline)	0.539	0.38	0.557	0.36	29.7%	28.6%
Token Entropy	0.510	0.50↑32.4%	0.523	0.50↑40.5%	21.9%	19.4%
Semantic Entropy	0.504	0.36↓6.2%	0.512	0.34↓6.1%	23.4%	23.2%
SFT	0.518	0.45↑17.9%	0.530	0.43↑20.6%	31.2%	28.8%
RSD	**0.555**	**0.28**↓26.8%	0.562	**0.27**↓23.8%	25.0%	28.6%
τ^2-bench Qwen-2.5-3B						
Base Qwen (baseline)	0.200	1.32	0.240	1.25	9.4%	0.3%
Token Entropy	0.204	1.38↑3.9%	0.247	1.32↑6.1%	9.4%	0.3%
Semantic Entropy	0.180	1.11↓15.9%	0.239	1.00↓20.0%	6.2%	1.0%
SFT	**0.289**	0.95↓28.0%	0.353	**0.90**↓27.8%	15.6%	5.4%
RSD	0.274	**0.91**↓30.9%	0.332	0.92↓26.1%	15.6%	3.6%
τ^2-bench Qwen-2.5-7B						
Base Qwen (baseline)	0.249	0.78	0.309	0.74	11.2%	0.6%
Token Entropy	0.304	0.75↓3.7%	0.387	**0.70**↓4.6%	15.6%	1.2%
Semantic Entropy	0.242	0.78↑0.5%	0.296	0.74±0%	7.8%	0.0%
SFT	**0.366**	**0.73**↓6.5%	0.462	0.72↓1.9%	25.0%	7.1%
RSD	0.278	0.89↑14.2%	0.349	0.85↑15.1%	12.5%	3.0%

6 Experimental Setup

We evaluate RSD on τ^2-bench [Barres et al., 2025], and ToolSandbox [Lu et al., 2024] to measure whether the learned metric captures trajectory-level consistency and if it can serve as a reward signal for improving consistency during fine-tuning. For each benchmark, we run 8 independent trials across 8 test samples (64 trajectories) in total for Qwen-2.5 models (3B and 7B parameters) [Team, 2024], where each trajectory consists of interleaved user messages, agent actions (including tool calls), and tool outputs. We construct the training set by generating multiple rollouts using temperature set to 1.0. We compare RSD against the Qwen baseline and three other methods, i.e., token-level entropy [Malinin and Gales, 2020], semantic entropy [Kuhn et al., 2023], and supervised

fine-tuning (SFT). For token entropy, semantic entropy, and RSD experiments, we fine-tune the base models using Group Relative Policy Optimization (GRPO) [Shao et al., 2024], using each metric as a reward signal. We minimize token and semantic entropy via negative entropy rewards, and minimize RSD self-distance $-\hat{U}_\theta(s, s)$. We also evaluate RSD on Qwen-3.5 [Qwen Team, 2026] across both benchmarks, comparing RSD fine-tuning against their corresponding base models. Refer to Section A.4 for results.

For the evaluation metric, we adopt the evaluation framework from Chong et al. [2026] to evaluate the agent behaviour. We report two task performance metrics, area under the progress curve (AUC) which measures how quickly the agent makes progress, and the final progress rate. To measure behavioral consistency, we compute the coefficient of variation (CV = σ/μ) of performance metrics across 8 independent trials per task, where a lower CV indicates more consistent behavior. To ensure a fair comparison across models, we restrict evaluation to the intersection of samples where all models within a size group achieve non-zero mean AUC. For inference, we set the temperature to 0. For more details, refer to Appendix A.3.

7 Results and Discussion

7.1 Main Results

Table 1 summarizes the overall performance and consistency changes across two benchmarks and two base models. In most cases, we observe that RSD achieves a substantial reduction in CV. Smaller models show the largest reduction in CV compared to larger models. On τ^2-bench with Qwen 2.5 3B, RSD reduces AUC CV by 30.9% and progress CV by 26.1% compared to the base model. On ToolSandbox 3B, we observe a 50.5% reduction in AUC CV and a 49.3% reduction in progress CV. These results suggest that smaller models exhibit higher intrinsic stochasticity in their decision-making processes. In contrast, 7B models exhibit lower baseline variance (e.g., 0.38 CV for ToolSandbox 7B compared to 0.90 for 3B), indicating more stable internal representations. For these larger models, the absolute gains from RSD are smaller because there is less variance to eliminate. On τ^2-bench with Qwen-2.5-7B, however, RSD increases both AUC CV and progress CV. A closer analysis of the individual test samples reveals one case where RSD fine-tuning leads to more frequent failures. For this sample, the mean progress decreases from 0.375 to 0.125, while the standard deviation also decreases from 0.518 to 0.354. However, the larger reduction in mean progress causes the CV to increase, despite the lower standard deviation. This illustrates that RSD may reinforce unsuccessful behavior, highlighting that consistency does not necessarily imply correctness.

We observe that token entropy and semantic entropy produce inconsistent results, with token entropy increasing CV in multiple cases. Minimizing token entropy encourages the model to make confident predictions at each decoding step. However, this token-level confidence does not translate to trajectory-level consistency. This occurs because token entropy measures local uncertainty at the vocabulary level without considering semantic meaning. Semantic entropy partially addresses this issue by clustering similar responses using NLI, but using this signal as a reward still produces inconsistent behavior on τ^2-bench (-15.9% on 3B and $+0.5\%$ on 7B) and achieves only modest reductions on ToolSandbox (-7.0% on 3B and -6.0% on 7B). The fundamental limitation of both methods is that they operate myopically at the level of individual turns. Agents can therefore be locally consistent while remaining globally inconsistent. RSD directly addresses this issue through its trajectory-level formulation (Definition 2) and empirically improves consistency across benchmarks.

We compare RSD with SFT to investigate whether access to ground-truth labels and expert demonstrations leads to more consistent behavior. We observe that on τ^2-bench, SFT reduces CV (-28.0% for 3B and -6.5% for 7B), but on ToolSandbox it increases CV ($+17.9\%$ for 7B).

Our training procedure estimates RSD from rollouts generated by a strong base agent that already makes non-trivial progress on the target benchmarks. As a result, the RSD network is trained only on trajectories sampled from a competent policy, and GRPO updates move the policy locally within this region of the behavior space. In particular, degenerate policies that always output a fixed response and achieve near-zero progress do not appear in the rollout distribution and are therefore not explicitly assigned low RSD. This helps explain why optimizing RSD in our experiments does not collapse to trivial "always fail consistently" solutions, even though such policies would theoretically achieve low RSD. Exploring from a randomly initialized or extremely weak policy could expose different failure

modes, and avoiding trivial but consistent solutions in that regime would likely require combining RSD with correctness-aligned rewards. For more information on CV calculation and sample count, refer to Section A.3.4.

7.2 Hybrid Training Ablation

Table 2 investigates whether combining RSD with task-specific progress rewards can improve upon pure RSD. We combine the reward from RSD (weight λ) and progress delta at each step (weight $1-\lambda$) on ToolSandbox for Qwen-2.5-7b model. We observe a counterintuitive result where RSD ($\lambda = 1$) achieves both the lowest CV and the best performance. As we decrease λ to incorporate progress rewards, both consistency and performance degrade monotonically. We suspect this is happening because of progress delta being a very sparse reward. Agent sometimes makes the most amount of progress in the final turns which causes previous turns to receive near 0 rewards, causing the signal to be noisy.

Table 2: Hybrid ablation on ToolSandbox Qwen-2.5-7B. The RSD weight λ controls the balance between RSD consistency reward (λ) and progress delta reward ($1-\lambda$). $\uparrow\downarrow$ shows relative change from pure RSD ($\lambda = 1.0$). The red upwards arrow $\uparrow$ indicates increased in variation.

Model	**AUC**	**AUC CV**	**Progress**	**Prog CV**	$pass@1$	$pass\hat{}3$
RSD (λ=1.0)	**0.555**	**0.28**	0.562	**0.27**	25.0%	28.6%
Hybrid (λ=0.8)	0.533	0.36$\uparrow$29.3%	0.541	0.33$\uparrow$21.1%	26.6%	28.6%
Hybrid (λ=0.5)	0.532	0.37$\uparrow$33.6%	0.539	0.35$\uparrow$28.2%	29.7%	21.4%
Hybrid (λ=0.2)	0.518	0.42$\uparrow$51.0%	0.536	0.35$\uparrow$27.0%	23.4%	21.4%

7.3 Evaluating finetuned models using RSD

To validate that finetuning with RSD as a reward signal reduces the actual inter-run semantic inconsistency captured by our metric, we evaluate the RSD score directly on the outputs of each finetuned model. For each test sample, the agent is run 8 times independently, and pairwise NLI (gpt-4.1) is used to classify whether outputs at each turn are functionally equivalent (entailment), contradictory, or neutral.

Table 3 reports the overall RSD scores across all methods. On ToolSandbox, RSD finetuning reduces the inter-run inconsistency by 52.8% (from 0.672 to 0.317), the largest reduction among all methods. Token entropy and semantic entropy also reduce inconsistency (-40.5% and -34.4% respectively), confirming that all uncertainty-based rewards can improve behavioral consistency, though RSD is most effective because it directly optimizes the trajectory-level consistency objective. Notably, SFT shows negligible improvement (-1.1%) on ToolSandbox despite memorizing successful trajectories, suggesting that supervised learning alone does not enforce output consistency.

On τ^2-bench, the baseline RSD is substantially higher (0.873), reflecting the compounding nature of multi-turn airline service tasks where small early differences cascade into divergent trajectories. All methods show limited improvement, with SFT achieving the largest reduction (-8.3%) by anchoring the model to fixed expert demonstrations.

Table 3: Direct RSD evaluation (lower = more consistent) across 8 independent trials per sample, measured via gpt-4.1.

	ToolSandbox 7B		*τ^2-bench 7B*	
Model	**RSD**	**Reduction**	**RSD**	**Reduction**
Base Qwen	0.672	—	0.873	—
Token Entropy	0.400	−40.5%	0.877	+0.5%
Semantic Entropy	0.440	−34.4%	0.860	−1.5%
SFT	0.664	−1.1%	**0.800**	−8.3%
Hybrid (λ=0.8)	0.387	−42.5%	0.867	−0.7%
RSD (ours)	**0.317**	−52.8%	0.875	+0.2%

8 Conclusion and Limitation

We introduced Recursive Semantic Divergence (RSD), an unsupervised trajectory-level consistency metric for LLM agents that captures how behavioral inconsistencies propagate across future interaction turns. Unlike existing correctness-based or token-level approaches, RSD provides a recursive estimate of semantic divergence between independent executions from the same interaction-history state, and can additionally be used as a training signal to reinforce consistent agent behavior. Experiments across multiple benchmarks demonstrate that optimizing RSD reduces trajectory variability.

Nevertheless, our approach has several limitations. RSD may require retraining or adaptation when transferred to new domains or environments. Moreover, consistency does not necessarily imply correctness, and therefore the metric cannot distinguish consistently failing behaviors from consistently successful ones. Finally, the reliability of RSD depends heavily on the accuracy and calibration of the underlying NLI model used to estimate semantic disagreement.

We use LLMs to polish the paper, write portions of the training and inference code, and verify the proofs presented in the paper.

Acknowledgments and Disclosure of Funding

This work received no external funding and the authors declare no competing interests beyond their disclosed affiliations.

References

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-bench: Evaluating conversational agents in a dual-control environment. *arXiv preprint arXiv:2506.07982*, 2025.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2015.

Pablo Samuel Castro, Tyler Kastner, Prakash Panangaden, and Mark Rowland. Mico: Improved representations via sampling-based state similarity for markov decision processes. In *Advances in Neural Information Processing Systems*, 2021.

Ma Chang, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. Agentboard: An analytical evaluation board of multi-turn llm agents. *Advances in neural information processing systems*, 37:74325–74362, 2024.

Penny Chong, Harshavardhan Abichandani, Jiyuan Shen, Atin Ghosh, Min Pyae Moe, Yifan Mai, and Daniel Dahlmeier. Talk, evaluate, diagnose: User-aware agent evaluation with automated error analysis. In *The Fourteenth International Conference on Learning Representations*, 2026. URL `https://openreview.net/forum?id=fHsVNklKOc`.

Jinhao Duan, James Diffenderfer, Sandeep Madireddy, Tianlong Chen, Bhavya Kailkhura, and Kaidi Xu. Uprop: Investigating the uncertainty propagation of llms in multi-step agentic decision-making, 2025. URL `https://arxiv.org/abs/2506.17419`.

Norm Ferns, Prakash Panangaden, and Doina Precup. Metrics for finite markov decision processes. In *Uncertainty in Artificial Intelligence*, 2004.

Kung-Hsiang Huang, Akshara Prabhakar, Onkar Thorat, Divyansh Agarwal, Prafulla Kumar Choubey, Yixin Mao, Silvio Savarese, Caiming Xiong, and Chien-Sheng Wu. Crmarena-pro: Holistic assessment of llm agents across diverse business scenarios and interactions. *arXiv preprint arXiv:2505.18878*, 2025.

Kyochul Jang, Donghyeon Lee, Kyusik Kim, Dongseok Heo, Taewhoo Lee, Woojeong Kim, and Bongwon Suh. Dice-bench: Evaluating the tool-use capabilities of large language models in multi-round, multi-party dialogues. *arXiv preprint arXiv:2506.22853*, 2025.

Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101:99–134, 1998.

Yubin Kim, Chanwoo Park, Hyewon Jeong, Yik Siu Chan, Xuhai Xu, Daniel McDuff, Hyeonhoon Lee, Marzyeh Ghassemi, Cynthia Breazeal, and Hae Won Park. MDAgents: An adaptive collaboration of LLMs for medical decision-making. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=EKdk4vxKO4.

Michael Kirchhof, Gjergji Kasneci, and Enkelejda Kasneci. Position: Uncertainty quantification needs reassessment for large-language model agents, 2025. URL https://arxiv.org/abs/2505.22655.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *International Conference on Learning Representations*, 2023.

Kim Guldstrand Larsen and Arne Skou. Bisimulation through probabilistic testing. *Inf. Comput.*, 94: 1–28, 1991. URL https://api.semanticscholar.org/CorpusID:45053958.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. Agentbench: Evaluating llms as agents. *arXiv preprint arXiv:2308.03688*, 2023.

Jiarui Lu, Thomas Holleis, Yizhe Zhang, Bernhard Aumayer, Feng Nan, Felix Bai, Shuang Ma, Shen Ma, Mengyu Li, Guoli Yin, et al. Toolsandbox: A stateful, conversational, interactive evaluation benchmark for llm tool use capabilities. *arXiv preprint arXiv:2408.04682*, 2024.

Andrey Malinin and Mark Gales. Uncertainty estimation in autoregressive structured prediction. *arXiv preprint arXiv:2002.07650*, 2020.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y.K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik R Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=vAElhFcKW6.

Qwen Team. Qwen2.5: A party of foundation models, September 2024. URL https://qwenlm.github.io/blog/qwen2.5/.

Tao Tu, Mike Schaekermann, Anil Palepu, Khaled Saab, Jan Freyberg, Ryutaro Tanno, Amy Wang, Brenna Li, Mohamed Amin, Yong Cheng, et al. Towards conversational diagnostic artificial intelligence. *Nature*, 642(8067):442–450, 2025.

Xingyao Wang, Zihan Wang, Jiateng Liu, Yangyi Chen, Lifan Yuan, Hao Peng, and Heng Ji. Mint: Evaluating llms in multi-turn interaction with tools and language feedback. *arXiv preprint arXiv:2309.10691*, 2023.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of NAACL-HLT*, 2018.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh J Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, et al. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. *Advances in Neural Information Processing Systems*, 37:52040–52094, 2024.

John Yang, Carlos E Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik R Narasimhan, and Ofir Press. SWE-agent: Agent-computer interfaces enable automated software engineering. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=mXpq6ut8J3.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains, 2024. *URL https://arxiv. org/abs/2406.12045*, 2024.

Appendix Contents

A.1 Algorithm for RSD training and inference 14

A.2 Theoretical Analysis 14

A.2.1 Convergence 14

A.2.2 Disagreement Probability 15

A.2.3 Approximation Error 17

A.3 Experimental Setup 18

A.3.1 RSD Training 18

A.3.2 Trajectory Divergence and Terminal States 18

A.3.3 Baselines 18

A.3.4 Evaluation Metrics 19

A.4 Evaluation on Qwen-3.5 Models 19

A.5 Validation of NLI Distance 20

A.5.1 Three-Class NLI Discretization 20

A.5.2 NLI Validation for Tool-Calling Actions 21

A.5.3 Cross-Model NLI Agreement 21

A Appendix

A.1 Algorithm for RSD training and inference

The following pseudocode details the algorithm for RSD training.

Algorithm 1 RSD Training: Learning the Consistency Metric

Require: Target agent policy π, M rollouts per task, NLI classifier, pretrained text encoder Enc
Require: Hyperparameters: γ, α, β, n (TD steps), learning rate η

Data collection: For each task, collect M independent rollouts $\{\tau^{(1)}, \ldots, \tau^{(M)}\}$ from π
NLI labeling: For all trajectory pairs (i, j) within the task, compute $d_{\text{NLI}}(a_t^{(i)}, a_t^{(j)})$ at each turn t
Initialize MICoNetwork parameters θ, target network $\bar{\theta} \leftarrow \theta$
for training step $= 1, \ldots, N$ **do**
  Sample a task and two trajectories $\tau^{(i)}, \tau^{(j)}$ for the same task
  Sample a turn t within the trajectory
  Compute embeddings: $z^{s_t^{(i)}} \leftarrow \text{Enc}(s_t^{(i)})$, $z^{s_t^{(j)}} \leftarrow \text{Enc}(s_t^{(j)})$
  Compute n-step TD target: $y_t \leftarrow \sum_{k=0}^{n-1} \gamma^k d_{\text{NLI}}(a_{t+k}^{(i)}, a_{t+k}^{(j)}) + \gamma^n \hat{U}_{\bar{\theta}}(s_{t+n}^{(i)}, s_{t+n}^{(j)})$
  Update θ to minimize $(\hat{U}_\theta(s_t^{(i)}, s_t^{(j)}) - y_t)^2$
  Soft update the target network
end for
return Trained MICoNetwork ϕ_θ

The following pseudocode details the algorithm for RSD inference.

Algorithm 2 RSD Inference: Per-State Consistency Scoring

Require: Trained MICoNetwork ϕ_θ, pretrained text encoder Enc, norm coefficient α
Require: Interaction history state s

$z^s \leftarrow \text{Enc}(s)$
$\phi_s \leftarrow \phi_\theta(z^s)$
$\hat{U}(s, s) \leftarrow 2\alpha \|\phi_s\|^2$
return $\hat{U}(s, s)$

A.2 Theoretical Analysis

We establish that the RSD fixed-point equation converges to a well-defined quantity with the NLI changes, and the quantity provides meaningful guarantees about agent behavior. Throughout, we let $\tau^{(i)}, \tau^{(j)} \sim \Pi(\cdot|s)$ denote two independent trajectory rollouts from state s, with $a_t^{(i)}, a_t^{(j)}$ the actions taken at step t in each rollout.

A.2.1 Convergence

We first verify that the recursive equation (Equation (6)) has a unique solution. Recall the RSD fixed-point equation:

$$U(s_t^{(i)}, s_t^{(j)}) = \mathbb{E}_{a_t^{(i)} \sim \pi(\cdot|s_t^{(i)}),\, a_t^{(j)} \sim \pi(\cdot|s_t^{(j)})} \big[d_{\text{NLI}}(a_t^{(i)}, a_t^{(j)}) \big] + \gamma \cdot \mathbb{E}\big[U(s_{t+1}^{(i)}, s_{t+1}^{(j)}) \big].$$

Define the Bellman operator $\mathcal{T}$ acting on bounded functions $U : \mathcal{H} \times \mathcal{H} \to \mathbb{R}_{\geq 0}$ by:

$$(\mathcal{T}U)(s_t^{(i)}, s_t^{(j)}) = \mathbb{E}_{a_t^{(i)} \sim \pi(\cdot|s_t^{(i)}),\, a_t^{(j)} \sim \pi(\cdot|s_t^{(j)})} \big[d_{\text{NLI}}(a_t^{(i)}, a_t^{(j)}) \big] + \gamma \cdot \mathbb{E}\big[U(s_{t+1}^{(i)}, s_{t+1}^{(j)}) \big],$$

where $s_{t+1}^{(i)} \sim P(\cdot|s^{(i)}, a^{(i)})$ and $s_{t+1}^{(j)} \sim P(\cdot|s^{(j)}, a^{(j)})$ are the respective successor states. Since $d_{\text{NLI}} \in [0, 1]$ is bounded and $\gamma < 1$, the operator $\mathcal{T}$ is a γ-contraction in supremum norm.

A.2.2 Disagreement Probability

Given that U^* is well-defined, we derive what the self-distance $U(s,s)$ implies about agent behaviour. We make use of the Definition 2, where self distance equals to the expected total discounted disagreement between two independent rollouts.

$$U(s,s) = \mathbb{E}_{\tau^{(i)},\tau^{(j)}\sim\Pi(\cdot|s)}\left[\sum_{t=0}^{\infty}\gamma^t\, d_{\text{NLI}}(a_t^{(i)}, a_t^{(j)})\right] = \mathbb{E}[D(\tau^{(i)},\tau^{(j)})], \tag{13}$$

where $D(\tau^{(i)},\tau^{(j)}) \coloneqq \sum_{t=0}^{\infty}\gamma^t\, d_{\text{NLI}}(a_t^{(i)}, a_t^{(j)}) \geq 0$ is the total discounted disagreement.

Theorem 1 (Trajectory Agreement)**.** *Let $D(\tau^{(i)},\tau^{(j)}) = \sum_{t=0}^{\infty}\gamma^t d_{NLI}(a_t^{(i)}, a_t^{(j)})$ be the total discounted disagreement between two independent rollouts from state s. For any threshold $\delta > 0$:*

$$\Pr\left(D(\tau^{(i)},\tau^{(j)}) \geq \delta\right) \leq \frac{U(s,s)}{\delta}. \tag{14}$$

Proof. Since $D(\tau^{(i)},\tau^{(j)}) \geq 0$ and $\mathbb{E}[D(\tau^{(i)},\tau^{(j)})] = U(s,s)$ by Equation (13), Markov's inequality directly gives:

$$\Pr\left(D(\tau^{(i)},\tau^{(j)}) \geq \delta\right) \leq \frac{\mathbb{E}[D(\tau^{(i)},\tau^{(j)})]}{\delta} = \frac{U(s,s)}{\delta}.$$

□

A useful instantiation is $\delta = \sqrt{U(s,s)}$, which gives $\Pr(D \geq \sqrt{U(s,s)}) \leq \sqrt{U(s,s)}$: both the disagreement threshold and the probability of exceeding it shrink as $U(s,s) \to 0$.

Corollary 2 (Immediate Contradiction Bound)**.** *The probability that two independent actions from state s contradict each other satisfies:*

$$\Pr\left(\textit{contradiction at step } 0\right) \leq U(s,s).$$

Proof. We first bounding the expected immediate disagreement by $U(s,s)$, then bounding the contradiction probability by the expected immediate disagreement.

Define $D_0(s) \coloneqq \mathbb{E}_{a_0^{(i)},a_0^{(j)}\sim\pi(\cdot|s)}[d_{\text{NLI}}(a_0^{(i)}, a_0^{(j)})]$ as the expected immediate disagreement. Expanding $D(\tau^{(i)},\tau^{(j)})$ into its first term and the remainder:

$$D(\tau^{(i)},\tau^{(j)}) = d_{\text{NLI}}(a_0^{(i)}, a_0^{(j)}) + \gamma\sum_{t=1}^{\infty}\gamma^{t-1}\, d_{\text{NLI}}(a_t^{(i)}, a_t^{(j)}).$$

Since $d_{\text{NLI}} \geq 0$, every term in the remainder is non-negative. Taking expectations of both sides:

$$U(s,s) = \mathbb{E}[D(\tau^{(i)},\tau^{(j)})] = D_0(s) + \underbrace{\gamma\mathbb{E}\left[\sum_{t=1}^{\infty}\gamma^{t-1}\, d_{\text{NLI}}(a_t^{(i)}, a_t^{(j)})\right]}_{\geq 0}.$$

Therefore $D_0(s) \leq U(s,s)$.

Then, let p_{entail}, p_{neutral}, and p_{contra} denote the probabilities that two independent samples $a_0^{(i)}, a_0^{(j)} \sim \pi(\cdot|s)$ yield bidirectional entailment, neutrality, or contradiction respectively, with $p_{\text{entail}} + p_{\text{neutral}} + p_{\text{contra}} = 1$. By Definition 1, d_{NLI} takes values 0, 0.5, and 1 for these three cases.

$$D_0(s) = 0 \cdot p_{\text{entail}} + 0.5 \cdot p_{\text{neutral}} + 1 \cdot p_{\text{contra}} = 0.5\, p_{\text{neutral}} + p_{\text{contra}}.$$

Since $0.5\, p_{\text{neutral}} \geq 0$, we have $D_0(s) \geq p_{\text{contra}}$.

This results in:

$$\Pr(\text{contradiction at step } 0) = p_{\text{contra}} \leq D_0(s) \leq U(s,s).$$

□

Theorem 3 (Dominant Action Class)**.** *If $U(s,s) < 1/4$, there exists a semantic equivalence class C^* such that:*

$$\Pr(\pi(s) \in C^*) \geq 1 - 2U(s,s). \tag{15}$$

Here $C^* = C_{k^*}$ where $k^* = \arg\max_k \Pr(\pi(s) \in C_k)$ is the most probable equivalence class under the policy.

Proof. Partition the action space into semantic equivalence classes $\{C_k\}_{k=1}^K$ defined by bidirectional NLI entailment: two actions $a^{(i)}, a^{(j)}$ belong to the same class C_k if and only if $d_{\text{NLI}}(a^{(i)}, a^{(j)}) = 0$ (i.e., they mutually entail each other). Let $p_k = \Pr(\pi(s) \in C_k)$ with $\sum_k p_k = 1$.

We first lower bound on collision probability. For two independent samples $a_0^{(i)}, a_0^{(j)} \sim \pi(\cdot|s)$, the probability they fall in the same class is:

$$\Pr(a_0^{(i)} \sim a_0^{(j)}) = \sum_{k=1}^K p_k^2.$$

When the samples fall in different classes, they are either neutral or contradictory. By Definition 1, $d_{\text{NLI}}(a_0^{(i)}, a_0^{(j)}) \geq 0.5$ in both cases. Therefore, we can lower-bound the expected immediate disagreement:

$$\begin{aligned}
D_0(s) &= \mathbb{E}[d_{\text{NLI}}(a_0^{(i)}, a_0^{(j)})] \\
&= \underbrace{\mathbb{E}[d_{\text{NLI}} \mid a_0^{(i)} \sim a_0^{(j)}]}_{=0} \Pr(a_0^{(i)} \sim a_0^{(j)}) \\
&\quad + \mathbb{E}[d_{\text{NLI}} \mid a_0^{(i)} \not\sim a_0^{(j)}] \Pr(a_0^{(i)} \not\sim a_0^{(j)}) \\
&\geq 0.5 \Pr(a_0^{(i)} \not\sim a_0^{(j)}) \\
&= 0.5 \left(1 - \sum_{k=1}^K p_k^2\right).
\end{aligned} \tag{16}$$

By Theorem 2, $D_0(s) \leq U(s,s)$. Substituting into the inequality from Equation (16):

$$0.5 \left(1 - \sum_{k=1}^K p_k^2\right) \leq U(s,s).$$

Rearranging:

$$\sum_{k=1}^K p_k^2 \geq 1 - 2U(s,s). \tag{17}$$

Now, we show the probability of the dominant class. For any probability distribution $(p_1, \ldots, p_K)$ with $\sum_k p_k = 1$, the following holds:

$$\max_k p_k \geq \sum_{k=1}^K p_k^2.$$

Combining with Equation (17):

$$\max_k p_k \geq \sum_k p_k^2 \geq 1 - 2U(s,s).$$

The class $C^* = C_{k^*}$ achieving this maximum satisfies $\Pr(\pi(s) \in C^*) \geq 1 - 2U(s,s)$. The condition $U(s,s) < 1/4$ ensures this bound exceeds $1/2$, meaning C^* contains a strict majority of the policy's probability mass. □

A.2.3 Approximation Error

The guarantees above hold for the true fixed point U^*. In practice, the learned approximation $\hat{U}_\theta$ differs from U^* due to two sources of error: (1) the state embedding ϕ loses information relative to the full interaction history, and (2) the NLI classifier $\hat{d}_{\text{NLI}}$ is imperfect. We quantify how these per-step errors propagate through the recursive structure.

Here, d^*_{NLI} denotes the ideal semantic disagreement function, while $\hat{d}_{\text{NLI}}$ denotes its classifier-based approximation.

Let ϵ_ϕ denote the worst-case per-step error from embedding compression:

$$\epsilon_\phi = \sup_{s^{(i)}, s^{(j)}} \left| \mathbb{E}[d^*_{\text{NLI}}(a^{(i)}, a^{(j)}) \mid s^{(i)}, s^{(j)}] - \mathbb{E}[d^*_{\text{NLI}}(a^{(i)}, a^{(j)}) \mid \phi(s^{(i)}), \phi(s^{(j)})] \right|$$

and let ϵ_{NLI} denote the worst-case NLI classification error:

$$\epsilon_{\text{NLI}} = \sup_{a^{(i)}, a^{(j)}} \left| \hat{d}_{\text{NLI}}(a^{(i)}, a^{(j)}) - d^*_{\text{NLI}}(a^{(i)}, a^{(j)}) \right|$$

For the following analysis, we assume that the state embedding preserves the policy-induced distribution of successor embeddings, such that states with identical embeddings have the same distribution over future embedded states. We further assume that the learned network $\hat{U}_\theta$ exactly represents the fixed point of the approximate Bellman operator. These assumptions allow us to isolate the effects of embedding compression and NLI classification errors.

Theorem 4 (Error Propagation)**.** *Under the assumptions above, the approximation error satisfies:*

$$\left| \hat{U}_\theta(s, s) - U^*(s, s) \right| \leq \frac{\epsilon_\phi + \epsilon_{NLI}}{1 - \gamma}. \tag{18}$$

Proof. Let $\tilde{\mathcal{T}}$ denote the approximate Bellman operator that uses $\hat{d}_{\text{NLI}}$ in place of d^*_{NLI} and conditions on embeddings $\phi(s)$ rather than full states.

We define both operators on full state pairs using the same policy-induced transition kernel. Under the embedding assumption, the fixed point of $\tilde{\mathcal{T}}$ can be represented using only the embedded states.

Since each modification introduces at most ϵ_{NLI} and ϵ_ϕ error respectively in the per-step cost, while the discounted recursive term is shared, we have for any function U and any state pair $(s^{(i)}, s^{(j)})$:

$$|(\mathcal{T}U)(s^{(i)}, s^{(j)}) - (\tilde{\mathcal{T}}U)(s^{(i)}, s^{(j)})| \leq \epsilon_\phi + \epsilon_{\text{NLI}}.$$

Both operators are γ-contractions (since only the per-step cost differs, not the discount structure). Let U^* and $\tilde{U}^*$ be their respective fixed points. Evaluating at U^*:

$$\tilde{\mathcal{T}}U^* = \tilde{\mathcal{T}}U^* - \mathcal{T}U^* + U^*,$$

so $\|\tilde{\mathcal{T}}U^* - U^*\|_\infty \leq \epsilon_\phi + \epsilon_{\text{NLI}}$. Applying $\tilde{\mathcal{T}}$ repeatedly and using contractivity:

$$\|\tilde{U}^* - U^*\|_\infty = \lim_{n \to \infty} \|\tilde{\mathcal{T}}^n U^* - U^*\|_\infty \leq \sum_{k=0}^{\infty} \gamma^k (\epsilon_\phi + \epsilon_{\text{NLI}}) = \frac{\epsilon_\phi + \epsilon_{\text{NLI}}}{1 - \gamma}.$$

□

For $\gamma = 0.8$, the amplification factor is $1/(1 - \gamma) = 5$, showing how per-step approximation errors accumulate through the recursive structure.

This bound holds under exact Bellman approximation. In practice, the learned network may introduce additional fitting error, which is not captured by the bound.

A.3 Experimental Setup

We evaluate RSD on τ^2-bench [Barres et al., 2025], and ToolSandbox [Lu et al., 2024] to measure whether the learned metric captures trajectory-level consistency and if it can serve as a reward signal for improving consistency during fine-tuning. τ^2-bench contains scenarios related to airline customer service agent, while ToolSandbox comprises general-purpose tasks spanning diverse domains, including calendar management and web search. For each benchmark, we run 8 independent trials across 8 test samples (64 trajectories) in total for Qwen-2.5 models (3B and 7B parameters) [Team, 2024], where each trajectory consists of interleaved user messages, agent actions (including tool calls), and tool outputs.

A.3.1 RSD Training

We construct the training set by generating multiple rollouts using temperature set to 1. For τ^2-bench we collect $M = 30$ independent rollouts from 13 task instances, and for ToolSandbox we collect $M = 8$ independent rollouts from 29 task instances. From the resulting trajectories, we extract state-action transition as described in Section 4.1. Each state s_t consists of the full interaction history up to timestep t, i.e, $(o_0, a_0, o_1, a_1, \ldots, o_{t-1}, a_{t-1}, o_t)$, where observations include user messages and tool outputs, and actions include agent responses and tool calls.

We adopt the MICo parameterization from Equation (2), where z^s is computed using a frozen Cohere embed-v4-0 and mapped through a 3-layer MLP into a 256-dimensional latent space. The network is trained to satisfy Equation (6) using $n = 3$ TD steps and $\gamma = 0.8$. To compute pairwise NLI relations between actions, we use gpt-4.1 to label each pair as entailment, contradiction or neutral, which are then mapped to scores as defined in Definition 1. The trained models are then used as reward signals to finetune different models. Refer to Algorithm 1 for more detail on the training algorithm.

A.3.2 Trajectory Divergence and Terminal States

As described in Section 4.3, two independently sampled trajectories from the same initial state are paired at the same transition depth. The shared index t does not indicate semantically corresponding decisions across the two rollouts. After divergence, each trajectory independently follows its own successor states, without attempting to realign the trajectories. Consider two independent rollouts from the same initial state:

$$\tau^{(i)} : \texttt{hotel_booking} \rightarrow \texttt{flight_booking} \rightarrow \texttt{confirmed},$$
$$\tau^{(j)} : \texttt{flight_booking} \rightarrow \texttt{hotel_booking} \rightarrow \texttt{confirmed}.$$

At the first two transition depths, the paired actions are classified as contradictions, while the final responses are classified as entailment. The TD target therefore accumulates the discounted semantic disagreement between the different execution paths rather than attempting to match semantically equivalent actions at different positions.

For variable-length trajectories, if either trajectory terminates before the n-step TD window, the target accumulates disagreement only over the valid paired transitions. Specifically, if only $m < n$ paired transitions are available before termination, the TD target in Equation (7) becomes

$$y_t = \sum_{k=0}^{m-1} \gamma^k d_{\text{NLI}}(a_{t+k}^{(i)}, a_{t+k}^{(j)}). \tag{19}$$

The bootstrap term is omitted because no paired successor state exists beyond the terminal transition. No padding or additional actions beyond termination are introduced.

A.3.3 Baselines

We compare RSD against three classes of methods:

Token-level entropy [Malinin and Gales, 2020]: We compute the average token-level entropy across all tokens in the agent's response at each turn. This provides a measure of uncertainty at the token generation level but does not capture semantic consistency across multiple turns.

Semantic entropy [Kuhn et al., 2023]: We cluster agent responses using bidirectional NLI entailment to identify semantic equivalence classes, then compute the entropy over these clusters. Unlike token entropy, this captures semantic variation, but only at single-response granularity without considering future trajectory divergence.

Supervised fine-tuning (SFT): We fine-tune the base Qwen-2.5 models on expert trajectories from both benchmarks using next-token prediction. This provides a baseline that improves task success through supervised learning but does not explicitly optimize for consistency.

For token entropy, semantic entropy, and RSD, we fine-tune the base models using Group Relative Policy Optimization (GRPO) [Shao et al., 2024], using each metric as a reward signal. We minimize token and semantic entropy via negative entropy rewards, and minimize RSD self-distance $-\hat{U}_\theta(s, s)$. All methods use the same base models (Qwen-2.5-3B and Qwen-2.5-7B), share GRPO hyperparameters, and are evaluated under identical trajectory collection settings with 8 trials per task for a fair comparison.

A.3.4 Evaluation Metrics

We adopt the evaluation framework from Chong et al. [2026] to evaluate the agent behaviour, as it captures both intermediate progress and final outcomes. We report two task performance metrics, area under the progress curve (AUC) which measures how quickly the agent makes progress, and the final progress rate.

To measure behavioral consistency, we compute the coefficient of variation (CV = σ/μ) of performance metrics across 8 independent trials per task, where a lower CV indicates more consistent behavior. To ensure a fair comparison across models, we restrict evaluation to the intersection of samples where all models within a size group achieve non-zero mean AUC. This avoids undefined CV where $\mu = 0$, and also prevents models outputting trivial solutions from appearing more consistent. This guarantees that every model is evaluated on identical samples where it demonstrates at least partial task engagement. Additionally, we report pass@k and pass^k metrics [Yao et al., 2024, Barres et al., 2025]. For inference, we set the temperature to 0. For $pass^3$, we report scores over the intersection of samples where all models within each benchmark–model-size group achieve non-zero mean task progress across trials. This ensures that all methods are evaluated on the same subset of samples.

A.4 Evaluation on Qwen-3.5 Models

We evaluate RSD on Qwen-3.5 models (4B and 9B parameters) across both τ^2-bench and ToolSandbox to investigate whether the consistency improvements extend across model families and scales. We compare RSD fine-tuning against the corresponding base models using AUC, final progress, and their respective coefficients of variation, as described in Section A.3.4.

Table 4: Performance and consistency, measured by the coefficient of variation (CV), for Qwen-3.5 models across both benchmarks. The blue downwards arrow ↓ indicates a reduction in variation, while the red upwards arrow ↑ indicates increased variation relative to the baseline.

Model	AUC	AUC CV	Progress	Progress CV
τ^2-bench Qwen-3.5-4B				
Base Qwen (baseline)	0.627	0.444	0.705	0.454
RSD	**0.686**	**0.355**↓20.0%	**0.768**	**0.359**↓20.9%
τ^2-bench Qwen-3.5-9B				
Base Qwen (baseline)	0.607	0.480	0.686	0.483
RSD	**0.650**	**0.428**↓10.8%	**0.720**	**0.427**↓11.6%
ToolSandbox Qwen-3.5-4B				
Base Qwen (baseline)	0.405	0.742	0.412	0.738
RSD	**0.447**	**0.662**↓10.8%	**0.451**	**0.662**↓10.3%
ToolSandbox Qwen-3.5-9B				
Base Qwen (baseline)	0.488	0.758	0.503	**0.720**
RSD	**0.507**	**0.738**↓2.6%	**0.515**	0.721↑0.1%

Table 4 summarizes the performance and consistency changes for Qwen-3.5 models. We observe that RSD reduces AUC CV across both benchmarks and model sizes, while simultaneously improving AUC and final progress. On τ^2-bench, RSD reduces AUC CV by 20.0% and progress CV by 20.9% for Qwen-3.5-4B, while Qwen-3.5-9B achieves reductions of 10.8% and 11.6%, respectively. On ToolSandbox, Qwen-3.5-4B achieves a 10.8% reduction in AUC CV and a 10.3% reduction in progress CV. For Qwen-3.5-9B, the improvements are smaller, with a 2.6% reduction in AUC CV and a marginal 0.1% increase in progress CV.

These results follow a similar trend to those observed in Table 1, where smaller models generally benefit more from RSD fine-tuning. The smaller improvements for Qwen-3.5-9B may be due to the same fine-tuning budget being used across model sizes, which may be insufficient for larger models. Overall, the results suggest that RSD can improve consistency and task performance across different model families, although the magnitude of improvement varies with model size and benchmark.

A.5 Validation of NLI Distance

A.5.1 Three-Class NLI Discretization

We adopt the three-class NLI formulation from Definition 1 to distinguish functionally equivalent, contradictory, and partially related actions. For tool-calling actions, we consider both the semantics of the tool and its arguments rather than relying on syntactic similarity. For example, two calls to the same tool with equivalent arguments are treated as entailment, while calls with conflicting arguments that lead to different outcomes are treated as contradiction. Neutrality captures cases where actions share similar intent but their functional equivalence is ambiguous.

To evaluate whether the three-class discretization is sufficient, we sample 12,000 action pairs from the τ^2-bench Airline domain and compare the proposed three-class scoring with a finer-grained 10-class scoring system (0–9). We observe that 42.5% of the scores fall in the range 0–1, 33.9% fall in the range 7–9, and only 6.1% fall in the middle range (4–6). We compute RSD using both scoring systems independently and compare the resulting rankings. The Spearman rank correlation is 0.97, indicating that the three-class formulation largely preserves the relative ordering obtained using finer-grained scoring.

Furthermore, finer-grained scoring requires the NLI classifier to distinguish between more semantic categories, potentially increasing sensitivity to calibration errors. We also hypothesize that finer-grained rewards may reduce the separation between consistent and inconsistent behaviors in group-relative advantage estimation during GRPO fine-tuning. These considerations motivate our use of the three-class formulation.

A.5.2 NLI Validation for Tool-Calling Actions

We further evaluate the reliability of GPT-4.1 NLI classifications for tool-calling actions. We programmatically categorize action pairs into three groups: calls to the same tool with identical arguments, calls to the same tool with different arguments, and calls to different tools. Table 5 reports the distribution of NLI classifications for each category.

Table 5: GPT-4.1 NLI classification distributions for tool-calling action pairs.

Category	N	Entailment	Neutral	Contradiction
Same tool + same args	1,602	97.6%	0.1%	2.3%
Same tool + different args	343	6.1%	0.0%	93.9%
Different tools	695	0.0%	0.0%	100.0%

We observe that 97.6% of action pairs with the same tool and arguments are classified as entailment, while 93.9% of pairs with the same tool but different arguments are classified as contradiction. All sampled pairs involving different tools are classified as contradiction. These results indicate that GPT-4.1 can distinguish between different tool-calling actions and their arguments in the evaluated samples.

We additionally manually annotate 30 randomly sampled action pairs across the three NLI categories (10 entailment, 15 contradiction, and 5 neutral) for functional equivalence. We observe 96.7% agreement between the manual annotations and GPT-4.1 judgments.

A.5.3 Cross-Model NLI Agreement

To investigate whether the NLI classifications depend on the choice of GPT-4.1, we compare its predictions with other LLMs and an open-source NLI model on randomly sampled action pairs from ToolSandbox. Table 6 reports the agreement between GPT-4.1 and the alternative models.

Table 6: Cross-model NLI agreement on sampled ToolSandbox action pairs.

Model Pair	Agreement (%)
GPT-4.1 vs Kimi-K2.5	95.0
GPT-4.1 vs GPT-5.1	98.0
GPT-4.1 vs GPT-4o	97.5
GPT-4.1 vs GPT-4o-mini	97.0
GPT-4.1 vs DeBERTa-v3-large	41.6

We observe high agreement between GPT-4.1 and the other LLM-based judges, ranging from 95.0% to 98.0%. In contrast, DeBERTa-v3-large achieves substantially lower agreement (41.6%). Unlike general-purpose LLMs, traditional NLI models such as DeBERTa are primarily trained on natural-language inference datasets and may not reliably capture the functional equivalence of structured tool calls. These results suggest that the NLI classifications are not specific to GPT-4.1.

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: Yes, the main claims in abstract and introduction is supported by theorem and proofs in Section 5 and experimental results in Section 7.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

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

   Answer: [Yes]

   Justification: We discuss the limitations in Section 8.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

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: The full set of assumptions are contained in the reference works [Larsen and Skou, 1991, Ferns et al., 2004, Castro et al., 2021] and in our work we only state the changes we make to their proofs. The complete and correct proofs are in Section 5 and Section A.2.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

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 experimental settings are described in Section 6 and Section A.3.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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 open access to the data and code with sufficient instructions to reproduce the main experimental results are in the code repository link placed in the footer of the first page.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

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) necessary to understand the results?

   Answer: [Yes]

   Justification: The experimental settings are described in Section 6 and Section A.3.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

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: Since we are measuring agent consistency, we are reporting the error bars in the form of the variance (CV) in the Tables 1 and 2.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

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: 8 $\times$ A100, 40GB VRAM GPUs. For Qwen 3.5 results 8 $\times$ B200 GPUs.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

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 research involves no personal data collection,no human subjects, and no scraping of third-party content. The paper also complies with the NeurIPS anonymity requirements for the submission phase.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: It does not involve personal data, human subjects, or deployment in real systems, and hence we foresee no direct societal impact beyond enabling privacy-preserving research infrastructure.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

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., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: This is non-applicable as the paper is about improving agent consistency.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

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: Yes, we have cited the papers we have used in this work.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [Yes]

    Justification: Code are provided and the usage are documented in the README.md file.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

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: [N/A]

    Justification: The paper involves no crowdsourcing and no research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

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: [N/A]

    Justification: The paper involves no research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

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? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: We have described the LLM usage in this paper in the conclusion and limitation Section 8.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.