Recursive Semantic Divergence for LLM Agent Consistency

Harshavardhan Abichandani Penny Chong Atin Ghosh Daniel Dahlmeier

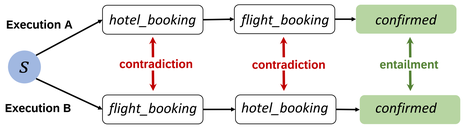
SAP SE

Paper Code

NEURAL INFORMATION PROCESSING SYSTEMS

Motivation

- Repeated runs of the same LLM agent can call different tools, follow different strategies, or even contradict themselves.
- Correctness metrics (success, pass@k) do not measure consistency of intermediate behavior
- Token entropy and semantic entropy focus on immediate outputs, not how disagreement propagates through future turns.
- We need a state level score that predicts future trajectory inconsistency.



Same state can lead to different futures. Therefore, agent consistency should be a trajectory level property

Proposed Idea

1 Bisimulation

- States are bisimilar if immediate outcomes are similar and future states are bisimilar

$$d(s_i, s_j) = |r_i - r_j| + \gamma W_d(P_i, P_j)$$

- W_d uses optimal coupling (Wasserstein distance)
- $d(s,s) = 0$, self distance is 0. However, calculating this for LLM state space is intractable.

2 Matching Under Independent Coupling (MICo)

- To make the recursion tractable, replace optimal transport over successor distributions with independently sampled successors

$$\underbrace{W_d(P_i, P_j)}_{\text{Optimal coupling}} \xrightarrow{\text{MICo}} \underbrace{\mathbb{E}_{\substack{X\sim P_i \\ Y\sim P_j}}[U(X,Y)]}_{\text{Independent sampling}}$$

$$P_i = P_j = P \implies \mathbb{E}_{X,Y\overset{\text{iid}}{\sim}P}[U(X,Y)] \text{ can be } > 0$$

- Thus, self-distance can be interpreted as a measure of stochastic variability

3 Recursive Semantic Divergence (RSD)

- Use Natural Language Inference (NLI) instead of reward difference.
- Evaluate on the self-pair, i.e. only on the required state.

$$U(s_i, s_j) = \mathbb{E}\left[d_{\text{NLI}}(a_i, a_j) + \gamma U(s_i', s_j')\right]$$

$$\text{RSD}(s) = U(s, s)$$

- Non-zero self distance becomes a lookahead measure of agent inconsistency

Key References

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

[2] 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.

Recursive Semantic Divergence (RSD)

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

RSD is the expected discounted semantic disagreement between two independent executions from the same interaction history

- RSD looks ahead over the entire trajectory until termination.
- Low RSD means futures mostly agree, High RSD means they disagree semantically.
- The learned metric is used to compute the score at the current state, before the full trajectory is completed.

Symbol	Meaning
s	interaction-history state
τ, τ'	independent future rollouts
a_t, a_t'	actions at step t
$d_{\text{NLI}}(a_t, a_t')$	semantic action disagreement
$\Pi(\cdot \mid s)$	trajectory distribution from s
γ	discount factor

Perfectly Consistent $U(s,s) = 0$ | **Mostly Consistent** Low $U(s,s)$ | **Highly Inconsistent** High $U(s,s)$

$$d_{\text{NLI}}(a_t, a_t') = 0 \text{ (entailment)}, \quad 0.5 \text{ (neutral)}, \quad 1 \text{ (contradiction)}$$

Learn the Scorer

1 Independent Rollouts — Run the same agent multiple times from same initial state

2 Pair by rollout depth — Pair $\left(s_t^{(i)}, a_t^{(i)}\right)$ and $\left(s_t^{(j)}, a_t^{(j)}\right)$ at the same depth t

3 NLI Disagreement + TD target — Compute $d_{\text{NLI}}\left(a_t^{(i)}, a_t^{(j)}\right)$ unroll n-step Bellman target

4 Learn and Infer — Fit $\widehat{U}_\theta$ and estimate RSD from the current state

Bellman equation $$U(s_i, s_j) = \mathbb{E}\left[d_{\text{NLI}}(a_i, a_j) + \gamma U(s_i', s_j')\right]$$

n-step TD target $$y_t = \sum_{k=0}^{n-1} \gamma^k d_{\text{NLI}}\left(a_{t+k}^{(i)}, a_{t+k}^{(j)}\right) + \gamma^n \widehat{U}_{\bar{\theta}}\left(s_{t+n}^{(i)}, s_{t+n}^{(j)}\right)$$

State encoder $$z_s = \text{Encoder}(s)$$

MICo Parameterization $$\hat{U}_\theta(s_i, s_j) = \alpha\left(\|\phi_\theta(z_{s_i})\|_2^2 + \|\phi_\theta(z_{s_j})\|_2^2\right) + \beta\, d_\angle\left(\phi_\theta(z_{s_i}), \phi_\theta(z_{s_j})\right)$$

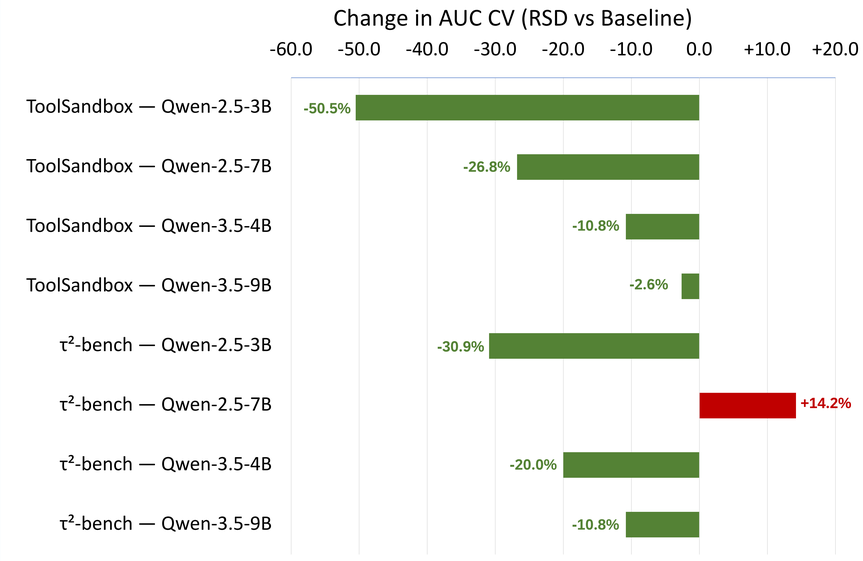
Self-pair Inference $\widehat{U}_\theta(s,s) = 2\alpha\|\phi_\theta(z_s)\|_2^2$

Consistency Reward $r_{\text{consistency}}(s) = -\hat{U}_\theta(s,s)$

Once the scorer is trained, inference is cheap with no additional rollouts.

Results

- RSD reduces Area Under the Progress Curve (AUC) Coefficient of Variation (CV) in 7 of 8 model-benchmark settings
- $\text{CV} = \frac{\sigma}{\mu}$ over repeated runs (lower means more consistent performance)
- Final task progress improves in all 8 settings
- Largest AUC CV reduction: ToolSandbox Qwen-2.5-3B (-50.5%)



Contributions & Limitations

Contributions

- We introduce Recursive Semantic Divergence (RSD), an unsupervised trajectory-level consistency metric that adapts bisimulation from reinforcement learning to evaluate LLM agents.
- 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.
- 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.

Limitations

- RSD may require retraining or adaptation when transferred to new domains or environments.
- Consistency does not necessarily imply correctness, and therefore the metric cannot distinguish consistently failing behaviors from consistently successful ones.

Trajectory Divergence

$$D(\tau, \tau') = \sum_{t=0}^{\infty} \gamma^t d_{\text{NLI}}(a_t, a_t')$$

Theorem (Concentration Bounds)

$$\Pr\left(D(\tau, \tau') \geq \delta\right) \leq \frac{\text{RSD}(s)}{\delta}$$

© 2026 SAP SE. All rights reserved