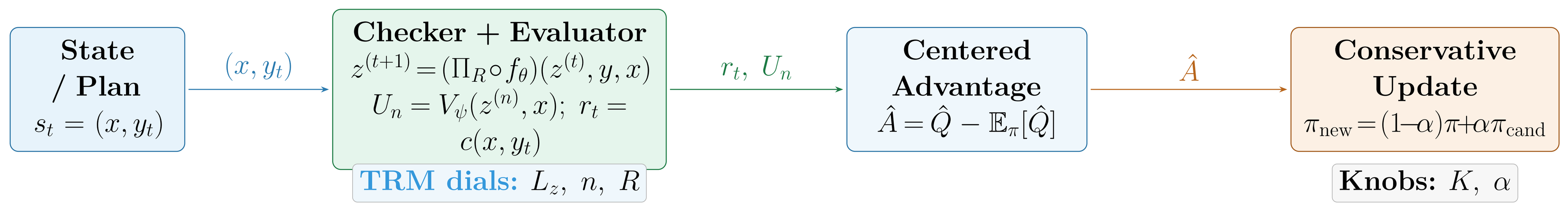
Unrolled Policy Iteration for Tiny Recursive Models

Bahram Behzadian*[1] Brett Daley*[1] Gopeshh Subbaraj[2] Houssam Nassif[1]

[1]Meta [2]Mila – Quebec AI Institute *Equal contribution

Checker-only UPI–TRM solves 9×9 Sudoku where PPO/A2C/DQN score 0%.
33× lower policy drift at 8× depth mismatch under inner-loop contraction.



Checker feedback and the n-step unrolled evaluator produce the advantage signal; the conservative update edits the plan and repeats.

Motivation

Tiny Recursive Models (TRMs) solve reasoning tasks by iteratively editing a candidate plan—but require ground-truth supervision.

Can we train from checker feedback alone? (e.g., "does this Sudoku satisfy constraints?")

Challenge: The evaluator is **approximate** (limited capacity) and **truncated** (finite depth n). How does truncation interact with policy improvement?

Key idea: Formalize plan editing as a **discounted MDP**; analyze via policy iteration with a compute-truncated value oracle.

Plan-Space MDP & Dials

State $s = (x, y)$: instance + plan
Action $y' = \text{edit}(y, a; x)$
Reward Checker $c(x, y)$ + shaping

Type	Param	Controls
TRM	$L_z < 1$	Contraction
Dials	n	Unroll depth
	R	Projection radius
Std	K	Bootstrap horizon
Knobs	α	Mixture weight

Value-Error Decomposition

Proposition (simplified). Under contraction ($L_z < 1$), over the advantage-evaluation closure:

$$\|U_n - V^\pi\|_{\infty,\pi} \leq \underbrace{\frac{\varepsilon^*_{\text{res}}}{1-\gamma^K}}_{\text{architectural}} + \underbrace{L_V \frac{L_z^n}{1-L_z} C_z}_{\text{truncation}}$$

Truncation bias decays **geometrically** as L_z^n.

Conservative Improvement

Theorem (ideal exact-centering case). For $\pi_{\text{new}} = (1-\alpha)\pi + \alpha\pi_{\text{cand}}$:

$$\eta(\pi_{\text{new}}) \geq \widehat{L}_\pi(\pi_{\text{new}}) - \frac{\alpha\varepsilon_{A,\text{cand}}}{1-\gamma} - \frac{2\varepsilon_{\text{CPI}}\gamma\alpha^2}{(1-\gamma)^2}$$

Exact centering makes evaluation error scale with α; with GAE, an added centering-defect term $\varepsilon_{\text{cent}}$ appears.

Main Contributions

1. **Checker-only UPI–TRM**
Learns plan edits from checker feedback and rewards, without expert trajectories.
2. **Value-error decomposition**
Separates architectural residual from geometric truncation bias $O(L_z^n)$.
3. **Conservative improvement**
With exact statewise centering, CPI evaluation error is linear in α.

UPI–TRM Algorithm

1. **Value Regression:** K-step bootstrap, min $(U_n(s) - \text{sg}(G^{(K)}))^2$
2. **Advantage Estimation:** $\bar{A} = \bar{Q} - \mathbb{E}_\pi[\bar{Q}]$ (statewise centered)
3. **Conservative Update:** $\pi_{\text{new}} = (1-\alpha)\pi + \alpha\pi_{\text{cand}}$, distill to π_ϕ

4×4 Sudoku Feasibility

Setting	UPI–TRM	PPO/A2C/DQN
Easy (1–4 empties)	90.7–93.3%	52% (random)
Hard (6–8 empties)	48–57%	**0%**

5k–20k steps, 3 seeds. Ranges denote contraction-on to no-contraction results; on hard instances, 11 tuned baseline configs all score 0%.

Depth-Mismatch Stability

$n_{\text{train}} = 2$, eval at 8× mismatch (both conditions use $R = 10$; projection alone gives 6–10× Δ_V reduction):

	$\Delta_V\downarrow$	$\Delta_\pi\downarrow$	Argmax↑
No Contr.	0.156	0.0063	96.0%
Contr. ($L_z = 0.9$)	**0.038**	**0.0002**	**99.0%**
Gain	4.1×	33×	—

Takeaway: Contraction lowers drift at every tested mismatch depth; at 8×, it cuts Δ_V by **4.1**× and Δ_π by **33**×.

9×9 Sudoku Results

Method	Success	Score	Δ
UPI–TRM	**4.0±4.0%**	**53.3±1.1**	**+26.5**
A2C	0.0±0.0%	31.9±0.4	+5.1
DQN	0.0±0.0%	29.5±0.3	+2.7
PPO	0.0±0.0%	28.2±1.5	+2.2

81 cells, 729 actions, 50k steps, mean±std over 3 seeds. UPI–TRM is the **only method solving any puzzles**; baselines cannot propagate constraints across the 81-step horizon.

Takeaways

(1) L_z, n, R are stability dials: truncation bias decays as L_z^n. **(2)** Conservative α limits sensitivity to imperfect evaluation. **(3)** UPI–TRM solves from checker feedback alone; PPO/A2C/DQN get 0%.

Contact: buiksat@meta.com