Learning Shortcut Models for Efficient Recursive Reasoning

Shiv Shankar[1]
[1]University of Massachusetts,
Correspondence: sshankar@cs.umass.edu

Abstract

Recursive models that progressively refine latent representations have demonstrated strong performance on a variety of reasoning tasks. However, these models only control whether and when to stop early, not how computation is distributed. In this work, we introduce *shortcut reasoning*, a framework for distilling recursive latent reasoning into a multiscale jump model that enables flexible test-time compute. We reinterpret recursive reasoning as a latent-time dynamical process and train a student model to predict the effect of multiple reasoning steps at once. To ensure robustness, we augment shortcut transitions with a repair mechanism, where a denoising variant of the base model projects latent states back onto a valid reasoning manifold. We further introduce stepwise improvement supervision, encouraging each shortcut step to increase the likelihood of the correct answer. Experiments on ARC-AGI show that our approach achieves competitive accuracy compared to recursive baselines while requiring fewer sequential updates.

1 Introduction

Recent research has shown that Transformer-style models can be made more efficient by sharing parameters over blocks (Lan et al., 2019; Jaegle et al., 2021; Dutta et al., 2021). Xu and Sato (2024); Saunshi et al. (2025); Gatmiry et al. (2024) show that such looped models demonstrate meaningfully enhanced problem solving when paired with additional recurrent computation. A related phenomenon appears in the broader literature on large language models. Chain-of-thought prompting and test-time compute can improve output quality by encouraging intermediate reasoning or by sampling multiple outputs (Wei et al., 2022; Snell et al., 2024). These results have motivated an alternative view in which reasoning occurs *internally in latent space*, without requiring long explicit textual traces.

Recent recursive reasoning models, such as the Hierarchical Reasoning Model (HRM) and Tiny Recursive Models (TRMs), exemplify this pattern (Wang et al., 2025; Jolicoeur-Martineau, 2025). These models solve challenging reasoning tasks by repeatedly updating latent states through a small recurrent core, sometimes at multiple temporal scales. It is suggested that an inductive bias allows them to efficiently learn *latent reasoning*: these models progressively refine internal representations over multiple recurrent passes (Saunshi et al., 2024; Geiping et al., 2025).

However, these recursive reasoning models still have major limitations. First, inference remains sequential since each reasoning step must be executed in order. Second, training supervision is concentrated primarily at the final output, which weakens credit assignment for early refinement steps and makes optimization brittle, especially in low-data regimes. Intermediate supervision can mitigate this issue, but does not by itself remove the dependence on fixed-length recursive computation. Finally, these models cannot directly adapt to the amount of compute resources available at test time.

These observations motivate the central question of this work: *can recursive latent reasoning be distilled into models that preserve iterative reasoning ability while supporting variable compute budgets?* We argue that the answer is yes. Our starting point is to reinterpret the recursive updates of a TRM/HRM-style model as a discrete latent-time dynamical system (Chen et al., 2018). Once viewed this way, the repeated refinement process need not be executed step by step. Instead, one can learn a *shortcut or jump reasoner*: a model that predicts the effect of several reasoning steps at once and thereby traverses the latent trajectory in large jumps (Frans et al., 2024).

Contributions. First, we introduce a *shortcut reasoner* for recursive latent reasoning models, in-

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop), pages 1351–1360

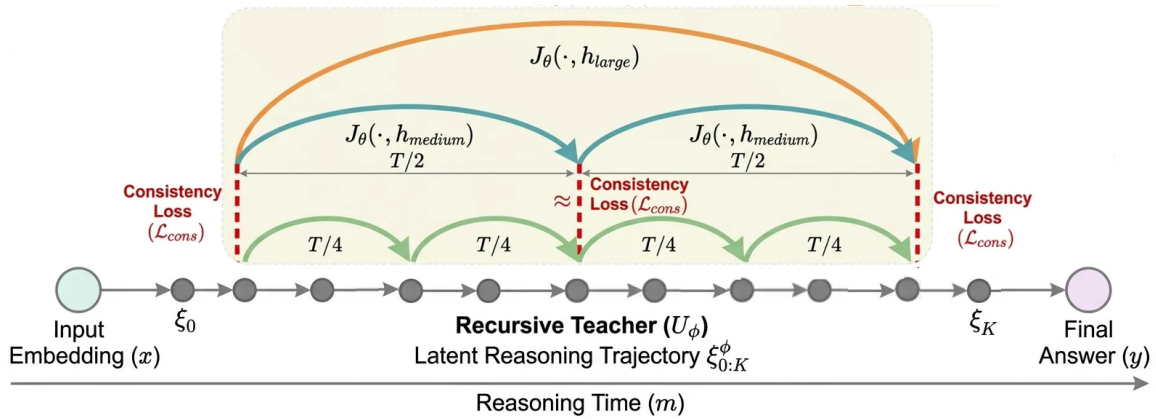

Figure 1: Overview of shortcut reasoning. A recursive teacher produces a latent reasoning trajectory through many small sequential updates $\xi_{m+1} = U_\phi(\xi_m, m; x)$. The shortcut model learns a scale-conditioned jump operator $J_\theta(\xi_m, m, h; x)$ that approximates the effect of h recursive steps at once, allowing the model to traverse the same latent trajectory with fewer updates. Training distills multistep teacher transitions and encourages consistency across jump sizes, so that one large jump agrees with compositions of smaller jumps. At inference time, different choices of h provide direct control over the compute–accuracy tradeoff.

cluding TRMs and HRMs, by viewing recursive latent updates as a discrete dynamical process and distilling them into a multiscale jump model. Second, we propose *stepwise shortcut supervision*, which encourages each shortcut transition to improve the gold-label likelihood rather than relying exclusively on final-output supervision. Third, we develop a *repair-aware shortcut framework* in which approximate long-range jumps are corrected by a noisy local denoising step implemented with the base recursive model, allowing the student to target repairable neighborhoods rather than exact latent states. Together, these ideas provide a path toward faster and more compute flexible reasoners.

2 Preliminaries

2.1 Background

Let the input sequence be $\tilde{x}$ and let

$$x = f_I(\tilde{x}) \in \mathbb{R}^{L\times D} \tag{1}$$

denote its embedding. In the Hierarchical Reasoning Model (HRM) (Wang et al., 2025) instantiation, the teacher maintains two latent states,

$$z_L, z_H \in \mathbb{R}^{L\times D}, \tag{2}$$

corresponding to low-level and high-level reasoning states. During a forward pass, the model applies n low-level updates for every high-level update, and repeats this outer schedule for T rounds. A typical schedule is $n = 2$ and $T = 2$. The final prediction is decoded from the high-level state. Overall, the HRM update equations are

$$z_L^{t+1} \leftarrow f_L^{\phi_L}\left((z_L^t + x) + z_H^t\right), \tag{3}$$

$$z_H^{t+1} \leftarrow f_H^{\phi_H}\left(z_L^{t+1} + z_H^t\right), \tag{4}$$

$$\hat{y} = \operatorname{argmax} f_O(z_H). \tag{5}$$

Tiny Recursive Models (Jolicoeur-Martineau, 2025) compress the larger HRM models into a single tiny network while retaining the recursive latent reasoning scheme. The main changes include (a) using the same function network for both the low-level and high-level modules, (b) adapting the adaptive computation time mechanism to use only one forward pass, and (c) using BPTT instead of an equilibrium model to backpropagate gradients. Ignoring the architectural choices that enable these modifications, TRM essentially repeats the same recursion as in Eqs. (3)–(5).

2.2 Related Work

Consistency, shortcut, and mean-flow models. Our work is closely related to recent approaches that learn direct transitions of an underlying iterative process rather than simulating all intermediate steps (Zhu et al., 2022; Zheng et al., 2023). Consistency models (Song and Dhariwal, 2023) learn mappings across all time steps of a trajectory by enforcing consistency across time translations, enabling few-step or one-step generation. Mean-flow

(Geng et al., 2025) style methods view dynamics through their averaged transport over a time interval rather than through infinitesimal updates. Similarly, shortcut models (Frans et al., 2024) learn scale-conditioned updates that predict the net effect of multiple small steps at once, providing a mechanism for variable-step-size integration. We build on this perspective, but apply it to recursive latent reasoning rather than generative sampling: the goal is not to accelerate diffusion toward a data sample, but to accelerate latent reasoning trajectories while preserving predictive improvement and allowing local repair.

Discrete diffusion and test-time compute. A second line of related work studies iterative inference and controllable test-time computation in discrete generative models. Discrete diffusion models refine a structured object through a sequence of denoising steps, and recent work has shown that these trajectories can be optimized to improve alignment and efficiency (Borso et al., 2025; Han et al., 2026). In parallel, test-time compute methods (Snell et al., 2024) have also been applied to diffusion-style generative models to trade compute for quality (Ramesh and Mardani, 2025; Ma et al., 2025). Our setting shares the same high-level motivation: to achieve better performance through adaptive inference-time computation. However, rather than allocating compute to repeated sampling or diffusion denoising, we allocate it directly in latent reasoning space through multiscale jumps and repair steps.

Latent reasoning and recurrent computation. Our method is also related to a growing body of work on latent reasoning in recurrent, looped, and recursive architectures. Parameter-shared Transformers, looped recurrent-depth models, and recent tiny recursive or hierarchical reasoning models all suggest that reasoning can emerge from repeated latent-state refinement rather than from explicit external chains of thought (Wang et al., 2025; Jolicoeur-Martineau, 2025; Saunshi et al., 2024; Jeddi et al., 2026; Dutta et al., 2021; Giannou et al., 2023). These models often benefit from additional recurrent depth, but are typically trained at fixed unroll lengths and therefore do not fully realize flexible compute at inference time (Jeddi et al., 2026). Adaptive halting partially addresses this issue by improving credit assignment or determining when to stop, but it still relies on local sequential updates (Jolicoeur-Martineau, 2025). We complement this line of work by introducing a shortcut model that learns to traverse reasoning trajectories at multiple temporal scales while remaining anchored to the base recursive dynamics through a repair mechanism. Recently, Asadulaev et al. (2025) proposed viewing latent recursion as a policy improvement operator and used that view to define an objective for per-step improvement. While our focus is different, we use a similar loss to provide temporal supervision to the shortcut model.

3 Shortcut Distillation of Recursive Latent Reasoning

We propose a method for distilling a recursive latent reasoning model, such as a TRM/HRM-style architecture, into a *shortcut* model that can jump across multiple reasoning steps at once while retaining the ability to recover valid latent states through a local repair step. The central idea is to reinterpret the teacher's recursive computation as a latent-time dynamical system, train a student shortcut field to approximate long-horizon latent displacements, and augment this student with a *repair operator* built from the base reasoning model so that approximate jumps can be projected back toward a valid reasoning manifold.

Our training objective combines four ingredients: (i) *trajectory distillation*, which matches shortcut jumps to teacher latent transitions; (ii) *shortcut self-consistency*, which enforces multiscale compositionality across jump sizes; (iii) *stepwise predictive improvement*, which encourages each reasoning step to improve the gold-label likelihood rather than supervising only the final state; and (iv) *repair consistency*, which allows shortcut states to be imperfect provided they lie in a basin from which one noisy base-model update restores good logits.

These terms play different roles. The trajectory matching and self-consistency losses are the core shortcut-distillation terms: they train s_θ to model latent state displacements over multiple horizons. The predictive-improvement loss does not replace this latent modeling objective; instead, it biases shortcut transitions toward states that improve the task prediction. The repair losses further relax exact endpoint matching by asking the shortcut state to be recoverable by the base dynamics, rather than requiring it to coincide exactly with the teacher state.

We begin with the TRM/HRM equations in Eqs. (3)–(5). To treat both the inner and outer

recursions uniformly, we flatten the nested computation into a single latent-time chain. Define the joint latent state

$$\xi_m = (z_{L,m}, z_{H,m}) \in \mathbb{R}^{2L\times D}, \tag{6}$$

where m indexes an *elementary reasoning step*. Let

$$K = T(n+1) \tag{7}$$

be the total number of elementary steps and let $\rho(m) = m \bmod (n+1)$ denote the phase within the low/high schedule. Then the teacher dynamics can be written as

$$\xi_{m+1} = U_\phi(\xi_m, m; x), \tag{8}$$

where $\phi = (\phi_L, \phi_H)$ and

$$U_\phi(\xi_m, m; x) = \begin{cases} \left(f_L^{\phi_L}(z_{L,m} + x + z_{H,m}),\ z_{H,m}\right), \\ \quad \text{if } \rho(m) \in \{0, \dots, n-1\}, \\ \left(z_{L,m},\ f_H^{\phi_H}(z_{L,m} + z_{H,m})\right), \quad \text{otherwise} \end{cases} \tag{9}$$

This piecewise map exactly reproduces the recursive computation of the base model. Starting from ξ_0, the teacher induces a latent trajectory

$$\xi_{0:K}^\phi = \left(\xi_0^\phi, \xi_1^\phi, \dots, \xi_K^\phi\right). \tag{10}$$

At any latent time m, one can decode from the high-level component:

$$\ell_m^\phi = f_O(z_{H,m}^\phi). \tag{11}$$

The likelihood of any generation y is denoted by

$$\pi_m^\phi(y \mid \tilde{x}) = \mathrm{softmax}(\ell_m^\phi)_y. \tag{12}$$

3.1 Latent-time dynamical view

We interpret the teacher trajectory as a discretization of an underlying latent-time process. While the actual teacher reasoner need not follow such a unique continuous flow, we treat it in such manner to present the intuition behind the distillation procedure. For any starting state ξ_m and horizon $h > 0$, define the normalized integrated displacement

$$\frac{d\xi(\tau)}{d\tau} = v_\star(\xi(\tau), \tau; x), \tag{13}$$

where the elementary teacher steps correspond to sampling the solution at integer times. For any starting state ξ_m and horizon $h > 0$, define the normalized integrated displacement

$$\begin{aligned} s_\star(\xi_m, m, h; x) &= \frac{\xi_{m+h} - \xi_m}{h} \qquad (14)\\ &= \frac{1}{h}\int_m^{m+h} v_\star(\xi(\tau), \tau; x)\, d\tau. \end{aligned}$$

Thus the exact evolution over horizon h is

$$\xi_{m+h} = \xi_m + h\, s_\star(\xi_m, m, h; x). \tag{15}$$

This suggests learning a student shortcut field that predicts the average latent direction over a horizon rather than simulating all intermediate steps.

3.2 Shortcut student model

Our goal is to approximate multiple reasoning steps with one jump, which we do via a parameterized shortcut model s_θ. We first define the shortcut jump operator:

$$J_\theta(\xi_m, m, h; x) = \xi_m + h\, s_\theta(\xi_m, m, h; x). \tag{16}$$

Given ξ_m, the student predicts a future latent state

$$\hat{\xi}_{m+h} = J_\theta(\xi_m, m, h; x). \tag{17}$$

We train over a dyadic set of horizons

$$\mathcal{H} = \left\{1, 2, 4, \dots, 2^{\lfloor \log_2 K \rfloor}\right\}, \tag{18}$$

restricted to pairs (m, h) such that $m + h \le K$.

The shortcut field should satisfy a semigroup consistency property (Frans et al., 2024). For the exact dynamics,

$$\begin{aligned} s_\star(\xi_m, m, 2h; x) &= \frac{\xi_{m+2h} - \xi_m}{2h} \qquad (19)\\ &= \frac{1}{2}\frac{\xi_{m+h} - \xi_m}{h} + \frac{1}{2}\frac{\xi_{m+2h} - \xi_{m+h}}{h} \qquad (20)\\ &= \frac{1}{2}s_\star(\xi_m, m, h; x) + \frac{1}{2}s_\star(\xi_{m+h}, m+h, h; x). \qquad (21) \end{aligned}$$

We use this identity to regularize the student across scales.

Shortcut trajectory distillation

The first objective matches the shortcut displacement to the teacher's latent finite difference:

$$\mathcal{L}_{\text{match}} = \mathbb{E}\left[\left\| s_\theta(\xi_m^\phi, m, h; x) - \frac{\xi_{m+h}^\phi - \xi_m^\phi}{h} \right\|_2^2\right]. \tag{22}$$

This term distills the teacher trajectory into a multiscale latent integrator and can be considered the primary matching loss used to train consistency or shortcut models (Song and Dhariwal, 2023; Frans et al., 2024).

In addition to per-step consistency, larger jumps require additional self-consistency. Let

$$\hat{\xi}_{m+h} = J_\theta(\xi_m^\phi, m, h; x) = \xi_m^\phi + h\, s_\theta(\xi_m^\phi, m, h; x), \tag{23}$$

and define the two-half-step target as:

$$s_{\text{target}} = \frac{1}{2} s_\theta(\xi_m^\phi, m, h; x) + \frac{1}{2} s_\theta(\text{sg}[\hat{\xi}_{m+h}], m+h, h; x), \tag{24}$$

where $\text{sg}[\cdot]$ denotes stop-gradient. The self-consistency loss is

$$\mathcal{L}_{\text{sc}} = \mathbb{E}\left[\left\| s_\theta(\xi_m^\phi, m, 2h; x) - \text{sg}[s_{\text{target}}] \right\|_2^2\right]. \tag{25}$$

Together, Eqs. (22) and (25) train the student to be a multiscale latent shortcut reasoning model.

Stepwise predictive improvement

Supervising only the teacher's final output provides only indirect supervision to the intermediate latent states. This hampers credit assignment and can slow learning. HRM (Wang et al., 2025) uses only one-step backpropagation by comparing latent updates to a fixed point computation. TRM (Jolicoeur-Martineau, 2025), on the other hand, performs BPTT (backpropagation through time), similar to RNNs, to provide supervision to earlier states. We adopt a TRM-style approach of providing supervision throughout, but instead of BPTT, we follow the perspective of diffusion-model-style tuning. We require that a shortcut transition should, on average, improve the likelihood of the gold tokens relative to alternate tokens.

Given a latent state ξ_m, the pre-reasoning logits are given by

$$\ell_m = f_O([\,\xi_m\,]_H), \tag{26}$$

where $[\,\cdot\,]_H$ extracts the high-level latent component. For the post-jump state $\hat{\xi}_{m+h} = J_\theta(\xi_m, m, h; x)$, define

$$\ell_{m+h} = f_O([\,\hat{\xi}_{m+h}\,]_H). \tag{27}$$

Let $\pi_m(\cdot \mid \tilde{x}), \pi^+_{m,h}(\cdot \mid \tilde{x})$ be the softmax likelihoods obtained from ℓ_m, ℓ_{m+h} respectively. Using these likelihoods, one can define the improvement

$$\Delta_{m,h}(y) = \log \pi^+_{m,h}(y \mid \tilde{x}) - \log \pi_m(y \mid \tilde{x}). \tag{28}$$

On average, a reasoning step is beneficial if the gold label $y^\star$ improves more than the average alternate token. This motivates the stepwise advantage (Asadulaev et al., 2025)

$$A_{m,h}(y^\star) = \Delta_{m,h}(y^\star) - \mathbb{E}_{y \sim \pi_m(\cdot|\tilde{x})}[\Delta_{m,h}(y)]\,. \tag{29}$$

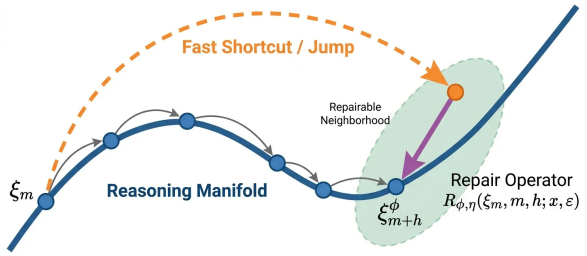

Figure 2: Illustration of how large jumps can deviate from desired trajectory, necessitating repair operation

which we optimize through the margin loss

$$\mathcal{L}_{\text{imp}} = \mathbb{E}_{(\tilde{x}, y^\star),\, m,\, h}\left[\text{softplus}(\gamma - A_{m,h}(y^\star))\right], \tag{30}$$

where $\gamma \geq 0$ is the desired improvement margin.

Remark 1. *Our presentation assumes finite and exact K-step training. In practice, recursive models are trained by sampling different numbers of steps. However, the finite-K formulation is still useful as a bounded teacher horizon for distillation.*

3.3 Repair-aware training

Why repair is needed

In our initial experiments, we saw that the shortcut reasoner was learning, yet even when we made the jumps the same size as those of the base reasoning model, its performance lagged behind TRM. We attributed this to a distribution shift between shortcut-based outputs and those of the TRM/HRM dynamics. A pure shortcut model requires the jump state $\hat{\xi}_{m+h}$ to lie near the teacher latent state ξ^ϕ_{m+h}. In practice, this is unnecessarily strict and hard to learn with lower-capacity models. A similar phenomenon has been observed for integrated vector field models like mean-flow models and shortcut diffusion, where the faster models do not adequately capture the output of the full dynamic integration of the vector field (Shankar and Geffner, 2025; Nguyen et al., 2025).

From the perspective of latent reasoning, we do not need the shortcut jump to produce the exact teacher latent. If the shortcut transports the latent state inside a basin of attraction of the teacher dynamics, then one noisy base update can denoise or project the state back to a good reasoning manifold. This observation motivates our *repair-aware* formulation, which calls the base TRM/HRM to recover a valid reasoning state in one local update.

Let $U_\phi(\xi, m; x)$ denote one elementary teacher update as defined in Eq. (8). We introduce an optional denoising adapter D_η, which receives a perturbed shortcut latent and outputs a repaired latent

before one base-model update. The repair operator $R_{\phi,\eta}(\xi_m, m, h; x, \varepsilon)$ is defined as

$$U_\phi\Big(D_\eta\big(J_\theta(\xi_m, m, h; x) + \sigma_h\varepsilon,\ m+h,\ x\big),\ m+h;\ x\Big), \tag{31}$$

where $\varepsilon \sim \mathcal{N}(0, \Sigma)$ and Σ is the covariance of the states ξ over the trajectories. If no separate denoising adapter is used, we simply set $D_\eta = \mathrm{Id}$, in which case the base TRM/HRM cell itself acts as the denoiser.

Equation (31) formalizes the notion that a shortcut state may be imperfect. We first allow stochastic perturbation of the shortcut latent, then apply a local denoising transformation, and finally run one teacher update. This produces a repaired state that should have valid semantics and good logits.

Repair targets

For teacher trajectory states, the repaired shortcut should approximate the teacher state one base step after the target horizon:

$$\xi^\phi_{m+h+1} = U_\phi(\xi^\phi_{m+h}, m+h; x). \tag{32}$$

A direct state-level repair loss $\mathcal{L}_{\text{rep-state}}$ is

$$\mathbb{E}\left[\left\| R_{\phi,\eta}(\xi^\phi_m, m, h; x, \varepsilon) - \xi^\phi_{m+h+1}\right\|_2^2\right]. \tag{33}$$

However, exact latent recovery may still be stronger than necessary. What ultimately matters is that the repaired state supports good predictions. Let

$$\pi^{\text{rep}}_{m,h+1}(\cdot \mid \tilde{x}) = \mathrm{softmax}\left(f_O\left([\, R_{\phi,\eta}(\xi^\phi_m, m, h; x, \varepsilon)\,]_H\right)\right) \tag{34}$$

be the repaired predictive distribution, and let

$$\pi^\phi_{m+h+1}(\cdot \mid \tilde{x}) = \mathrm{softmax}\Big(f_O([\,\xi^\phi_{m+h+1}\,]_H)\Big) \tag{35}$$

denote the teacher distribution at the same repaired time index. We define the repair logit loss

$$\mathcal{L}_{\text{rep-logit}} = \mathbb{E}\Bigg[D_{\mathrm{KL}}\left(\pi^\phi_{m+h+1} \,\|\, \pi^{\text{rep}}_{m,h+1}\right) - \lambda_{\text{gold}} \log \pi^{\text{rep}}_{m,h+1}(y^\star \mid \tilde{x})\Bigg]. \tag{36}$$

This term says that after a shortcut, even with latent noise, one base-model update should recover a state whose logits agree with the teacher and favor the correct label.

3.4 Stepwise repair improvement and repair consistency

Since repaired states are the trusted outputs of the jump-repair mechanism, the stepwise reward is more naturally defined after repair than after the raw jump. Let the pre-reasoning distribution remain

$$\pi_m(\cdot \mid \tilde{x}) = \mathrm{softmax}(f_O([\,\xi_m\,]_H)), \tag{37}$$

and let $\pi^{\text{rep}}_{m,h+1}$ be defined as in Eq. (34). We define the repair-aware per-action improvement as

$$\Delta^{\text{rep}}_{m,h}(a) = \log \pi^{\text{rep}}_{m,h+1}(a \mid \tilde{x}) - \log \pi_m(a \mid \tilde{x}). \tag{38}$$

Similar to Asadulaev et al. (2025), we can define a repaired stepwise advantage as

$$A^{\text{rep}}_{m,h}(y^\star) = \Delta^{\text{rep}}_{m,h}(y^\star) - \mathbb{E}_{y\sim\pi_m(\cdot\mid\tilde{x})}\left[\Delta^{\text{rep}}_{m,h}(y)\right]. \tag{39}$$

Similar to Eq. (30), we can provide supervision via the corresponding margin loss. Thus each shortcut step is asked not merely to move toward a future teacher state, but to land in a region from which one local repair step improves predictive confidence in the gold label.

A central structural constraint is that shortcut and repair operations should agree with larger repaired jumps. Concretely, starting from ξ_m, define the *jump–repair–jump* path

$$\xi^{\text{JRJ}}_{m+2h+1} = J_\theta\Big(R_{\phi,\eta}(\xi_m, m, h; x, 0), m+h+1, h; x\Big). \tag{40}$$

This corresponds to jumping h steps, taking one base-model update, and jumping another h steps.

Now define the *big-jump repair* path

$$\xi^{\text{BJR}}_{m+2h+1} = R_{\phi,\eta}(\xi_m, m, 2h; x, \varepsilon), \tag{41}$$

which corresponds to making one large jump of size $2h$.

Both paths terminate at the same logical latent time $m + 2h + 1$. We therefore enforce

$$\xi^{\text{JRJ}}_{m+2h+1} \approx \xi^{\text{BJR}}_{m+2h+1}. \tag{42}$$

This yields the state-level path consistency loss $\mathcal{L}_{\text{JRJ}}$:

$$\mathbb{E}_{(\tilde{x},y^\star),\,m,\,h,\,\varepsilon}\left[\left\|\xi^{\text{JRJ}}_{m+2h+1} - \mathrm{sg}\big[\xi^{\text{BJR}}_{m+2h+1}\big]\right\|_2^2\right]. \tag{43}$$

This encodes the principle that shortcut composition should remain coherent under local repair. A large jump followed by one repair step should agree with two smaller jumps with an intermediate repair.

Training Procedure In practice, the full objective admits a natural curriculum. One may first train the shortcut field with $\mathcal{L}_{\text{match}} + \mathcal{L}_{\text{sc}}$, then introduce repair objectives $\mathcal{L}_{\text{rep-state}} + \mathcal{L}_{\text{rep-logit}}$, and finally add the predictive-improvement and path-consistency losses. This decomposition stabilizes optimization because the shortcut first learns approximate multiscale motion before being asked to satisfy repair and reward constraints.

Inference At inference time, the original recursive teacher requires $K = T(n+1)$ sequential latent updates. By contrast, the shortcut student can traverse latent time using a small number of dyadic jumps, optionally interleaved with sparse repair steps. For example, to advance by $K = 12$ latent steps, the student may use a decomposition $12 = 8 + 4$ and evaluate two shortcut jumps rather than all 12 base updates. If repair is inserted after selected jumps, the number of expensive base-model evaluations remains small.

Thus, the proposed method compresses recursive latent reasoning into a small number of semantically meaningful long-range jumps, while preserving robustness through repair by the original reasoning dynamics.

Remark 2. *Note that many recursive reasoners also use an adaptive computation time (ACT) halting mechanism, which allows the reasoning model to stop looping early. While this also saves computational resources, our focus is different. In ACT, the model decides whether to take another step, but each step is executed at the same granularity (Wang et al., 2025; Jolicoeur-Martineau, 2025). Thus, ACT provides adaptive stopping, but not adaptive* step size. *We, on the other hand, introduce another axis of granularity (i.e., step size) for the recursive operation. The repair mechanism makes this practical by relaxing the requirement that a long jump land exactly on the teacher trajectory; instead, it suffices for the jump to reach a repairable neighborhood from which one noisy base-model update restores a valid reasoning state with good logits.*

4 Experiments

In this section, we evaluate our shortcut reasoning model on the ARC-1 and ARC-2 benchmarks, focusing on both final accuracy and the ability to control test-time compute.

4.1 Reasoning Experiments

Backbone. We build on the Tiny Recursive Model (TRM) backbone and first train a standard TRM teacher, which is then used for distillation. Both teacher and student use a Transformer architecture, and we keep the student as close as possible to the TRM design. The primary modification is the addition of a shortcut head that conditions on the *jump size* h. Since teacher trajectories are discretized, we use an integer latent time index m.

We follow the original TRM training recipe wherever applicable. Specifically, we use AdamAtan with $\beta_1 = 0.9$ and $\beta_2 = 0.95$, a 2k step warm-up, and the stable-max cross-entropy variant for improved stability. For ARC experiments, we use weight decay 0.1 and do not observe significant gains from EMA. We also match batch sizing and retain the elevated embedding learning rate and warm-up schedule used in TRM.

Model configurations. We consider three baselines and our proposed model: following Jolicoeur-Martineau (2025); Asadulaev et al. (2025), **TRM** trained with $T = 1$, $n = 2$, and $N_{\text{sup}} = 16$; **TRM+ACT** with the halting mechanism active; **DIS** from Asadulaev et al. (2025); and our proposed **Shortcut model**, trained via trajectory distillation from the TRM teacher with multiscale jump sizes. To ensure a fair comparison, we use a pretrained TRM and then distill it via the multi-stage curriculum described in the previous section. For training, we use the same hyperparameters. We compare three versions of our shortcut model with $r = 8, 4, 2, 1$ sized jumps against the baseline TRM, corresponding to 0.125, 0.25, 0.5, and 1 times the FLOPs of TRM during inference.

Results. The results are presented in Table 1. We can see that the shortcut model achieves competitive performance while using significantly fewer effective reasoning steps/FLOPs. These results support the hypothesis that recursive latent reasoning trajectories can be approximated with jumps without significant deterioration in predictive quality.

4.2 Ablation

Beyond reporting final accuracy, we conduct ablation experiments to better understand the impact of different components of the proposed method.

Jump size ablation. In Figure 3, we plot the aggregated correlation between the embedding outputs of the shortcut model and those in the teacher

Method	Params	ARC-1	ARC-2
Pretrained, CoT			
DeepSeek R1	671B	15.8	1.3
Claude 3.7		28.6	0.7
o3-mini-high		34.5	3.0
Gemini 2.5 Pro	?	37.0	4.9
Grok-4-thinking	1.7T	66.7	16.0
Bespoke (Grok-4)	1.7T	79.6	29.4
Small-sample training			
TRM	7M	40.4	3.3
DIS	7M	41.3	6.0
Ours (8-hop)	7.2M	28.6	1.7
Ours (4-hop)	7.2M	34.2	2.4
Ours (2-hop)	7.2M	39.7	3.0
Ours (1-hop)	7.2M	40.8	5.8

Table 1: Comparison on ARC-AGI benchmarks. Our method matches and sometimes improves over TRM. The hop factor indicates how large a step size the shortcut model takes compared to the baseline TRM model. Baseline results are as reported in Asadulaev et al. (2025).

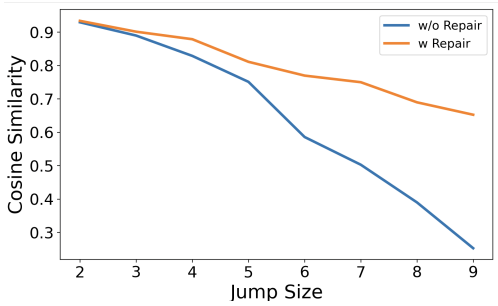

Figure 3: Correlation between the shortcut/jump model output and the same number of recursive steps in the base model. We see that without the ability to repair, the shortcut model fails to track the teacher for jump sizes greater than 5. With repair, the shortcut model maintains good correlation with the teacher trajectory.

trajectory over different jump sizes. We can see that the shortcut model can track the teacher model to a certain extent, but the tracking ability drops significantly after a jump size of 4–5.

Repair ablation. To improve the tracking of the teacher trajectory, we introduced the repair mechanism, which allows the model to denoise its output with a trained denoiser plus one step of the teacher model. With this repair mechanism available, the model is able to keep much better track of the teacher trajectory.

Stepwise improvement ablation. In Figure 4, we present model accuracy over training steps with and without per-step improvement loss. We see that the model learns significantly faster with per-step supervision. With per-step supervision, the model reaches an accuracy of 25% within 50k steps, which the non-per-step trained model takes roughly 300k steps to reach.

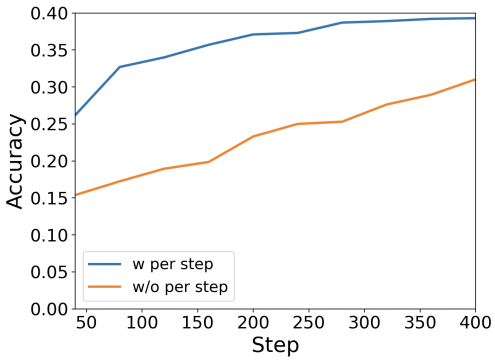

Figure 4: Pass@2 scores for training with and without per-step supervision (Eq. (30)). The x-axis is in units of 1000 training steps. Per-step supervision allows the model to learn much faster.

5 Conclusion

We presented a shortcut reasoning framework for distilling recursive latent reasoners into multiscale jump models that enable flexible test-time compute. By viewing recursive updates as a latent dynamical process, our method replaces long sequences of small refinement steps with a small number of learned jumps, augmented by local repair through a denoising version of the base model. Our results on ARC-AGI demonstrate that shortcut reasoning can match or improve the performance of recursive baselines while significantly reducing sequential computation. Moreover, the proposed formulation provides a richer notion of *budgeted control*, in which the model can decide not only whether to spend more compute, but also how much latent progress to make before the next decision point.

Future work includes formulating an ACT-like module that will allow the model to control shortcut reasoning and developing a deeper theoretical understanding of latent reasoning dynamics.

Limitations

Our approach has several limitations. First, compute allocation is largely global rather than instance-adaptive: we fix jump sizes and supervision schedules across tasks, despite significant variability in problem difficulty, whereas prior work shows adaptive halting can be critical for performance (Jolicoeur-Martineau, 2025; Jeddi et al.,

2026). Second, training introduces additional overhead due to multiscale shortcut supervision and consistency objectives across trajectories. Third, our experiments are limited to ARC-AGI, and conclusions may not generalize to other domains. Finally, while we reduce inference FLOPs, we introduce an additional network for shortcut prediction, increasing the overall memory usage.

LLM Usage The authors used ChatGPT as a writing assistant in preparing this manuscript. All interpretations and any errors remain solely the responsibility of the authors.

References

Arip Asadulaev, Rayan Banerjee, Fakhri Karray, and Martin Takac. 2025. Deep improvement supervision. *arXiv preprint arXiv:2511.16886*.

Umberto Borso, Davide Paglieri, Jude Wells, and Tim Rocktäschel. 2025. Preference-based alignment of discrete diffusion models. *Preprint*, arXiv:2503.08295.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K. Duvenaud. 2018. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, volume 31.

Subhabrata Dutta, Tanya Gautam, Soumen Chakrabarti, and Tanmoy Chakraborty. 2021. Redesigning the transformer architecture with insights from multiparticle dynamical systems. In *Advances in Neural Information Processing Systems*, volume 34, pages 5531–5544.

Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. 2024. One step diffusion via shortcut models. *Preprint*, arXiv:2410.12557.

Khashayar Gatmiry, Nikunj Saunshi, Sashank J. Reddi, Stefanie Jegelka, and Sanjiv Kumar. 2024. On the role of depth and looping for in-context learning with task diversity. *Preprint*, arXiv:2410.21698.

Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian R. Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. 2025. Scaling up test-time compute with latent reasoning: A recurrent depth approach. *Preprint*, arXiv:2502.05171.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J Zico Kolter, and Kaiming He. 2025. Mean flows for one-step generative modeling. *arXiv preprint arXiv:2505.13447*.

Angeliki Giannou, Shashank Rajput, Jy-yong Sohn, Kangwook Lee, Jason D. Lee, and Dimitris Papailiopoulos. 2023. Looped transformers as programmable computers. In *Proceedings of the 40th International Conference on Machine Learning*, pages 11398–11442.

Jiaqi Han, Austin Wang, Minkai Xu, Wenda Chu, Meihua Dang, Haotian Ye, Huayu Chen, Yisong Yue, and Stefano Ermon. 2026. Discrete diffusion trajectory alignment via stepwise decomposition. *Preprint*, arXiv:2507.04832.

Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. 2021. Perceiver: General perception with iterative attention. In *Proceedings of the 38th International Conference on Machine Learning*, pages 4651–4664.

Ahmadreza Jeddi, Marco Ciccone, and Babak Taati. 2026. Loopformer: Elastic-depth looped transformers for latent reasoning via shortcut modulation. *arXiv preprint arXiv:2602.11451*.

Alexia Jolicoeur-Martineau. 2025. Less is more: Recursive reasoning with tiny networks. *Preprint*, arXiv:2510.04871.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. *Preprint*, arXiv:1909.11942.

Nanye Ma, Shangyuan Tong, Haolin Jia, Hexiang Hu, Yu-Chuan Su, Mingda Zhang, Xuan Yang, Yandong Li, Tommi Jaakkola, Xuhui Jia, et al. 2025. Inference-time scaling for diffusion models beyond scaling denoising steps. *arXiv preprint arXiv:2501.09732*.

Anh Nguyen, Viet Nguyen, Duc Vu, Trung Dao, Chi Tran, Toan Tran, and Anh Tran. 2025. Improved training technique for shortcut models. *arXiv preprint arXiv:2510.21250*.

Vignav Ramesh and Morteza Mardani. 2025. Test-time scaling of diffusion models via noise trajectory search. *arXiv preprint arXiv:2506.03164*.

Nikunj Saunshi, Nishanth Dikkala, Zhiyuan Li, Sanjiv Kumar, and Sashank J. Reddi. 2025. Reasoning with latent thoughts: On the power of looped transformers. *Preprint*, arXiv:2502.17416.

Nikunj Saunshi, Stefani Karp, Shankar Krishnan, Sobhan Miryoosefi, Sashank J. Reddi, and Sanjiv Kumar. 2024. On the inductive bias of stacking towards improving reasoning. *Preprint*, arXiv:2409.19044.

Shiv Shankar and Tomas Geffner. 2025. Learning straight flows by learning curved interpolants. *arXiv preprint arXiv:2503.20719*.

Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling LLM test-time compute optimally can be more effective than scaling model parameters. *Preprint*, arXiv:2408.03314.

Yang Song and Prafulla Dhariwal. 2023. Improved techniques for training consistency models. *arXiv preprint arXiv:2310.14189*.

Guang Wang, Jialin Li, Yifan Sun, Xiaoyu Chen, Cheng Liu, Yuxuan Wu, Ming Lu, Shiji Song, and Yasin Abbasi Yadkori. 2025. A hierarchical reasoning model. *Preprint*, arXiv:2506.21734.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed H. Chi, Quoc V. Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837.

Kevin Xu and Issei Sato. 2024. On expressive power of looped transformers: Theoretical analysis and enhancement via timestep encoding. *Preprint*, arXiv:2410.01405.

Hongkai Zheng, Weili Nie, Arash Vahdat, Kamyar Azizzadenesheli, and Anima Anandkumar. 2023. Fast sampling of diffusion models via operator learning. In *International conference on machine learning*, pages 42390–42402. PMLR.

Aiqing Zhu, Pengzhan Jin, Beibei Zhu, and Yifa Tang. 2022. On numerical integration in neural ordinary differential equations. In *International Conference on Machine Learning*, pages 27527–27547. PMLR.

A Appendix

A.1 Stepwise trajectory decomposition perspective

The above construction may be viewed as a tractable approximation to trajectory-level alignment over the full latent reasoning chain. Let $q_\theta(\xi_{0:K} \mid x)$ denote the student-induced trajectory distribution and let $p_\phi(\xi_{0:K} \mid x)$ denote the teacher trajectory distribution. A global RLHF-style objective would be

$$\max_{q_\theta} \mathbb{E}_{q_\theta(\xi_{0:K}|x)}\left[r(x,y)\right] - \beta D_{\mathrm{KL}}\left(q_\theta(\xi_{0:K} \mid x) \,\|\, p_\phi(\xi_{0:K} \mid x)\right), \tag{44}$$

where r is the cumulative reward. Optimizing Eq. (44) directly is expensive because it requires reasoning over entire latent chains. Furthermore, even when possible, this is not suitable for credit assignment because the reward is global and there is no per-step supervision.

Stepwise decomposition (Han et al., 2026) instead solves local subproblems over conditional transitions:

$$\max_{q_{\theta,h}} \mathbb{E}_{q_{\theta,h}(\xi_{m+h}|\xi_m,x)}\left[r_{m,h}(\xi_m, \xi_{m+h}, y)\right] - \beta_{m,h} D_{\mathrm{KL}}\left(q_{\theta,h}(\xi_{m+h} \mid \xi_m, x) \,\|\, p_\phi(\xi_{m+h} \mid \xi_m, x)\right),$$

where the local reward is given by the repair-aware stepwise advantage,

$$r_{m,h}(\xi_m, \xi_{m+h}, y) = \frac{\mathbb{E}_{q(y|\xi_{m+h},x)}[\frac{1}{\beta}\exp(r(x,y))]}{\mathbb{E}_{q(y|\xi_m,x)}[\frac{1}{\beta}\exp(r(x,y))]}. \tag{45}$$

Using the likelihood of the answer as the reward, the above expression simplifies to $A^{\mathrm{rep}}_{m,h}(y)$, motivating Eq. (39), where the reward for $y^\star$ is maximized while others are suppressed.

Our deterministic shortcut-repair model approximates this by parameterizing the future latent through

$$\xi_{m+h} \approx J_\theta(\xi_m, m, h; x) \tag{46}$$

$$\xi_{m+h+1} \approx R_{\phi,\eta}(\xi_m, m, h; x, \varepsilon), \tag{47}$$

and then optimizing the resulting state-matching, consistency, and reward terms.