# Semantic Reasoning Denoising: Correcting Language Model Reasoning with Semantic Operators

**Yujiao Yang**
Dalian University of Technology
yjyang@mail.dlut.edu.cn



**Abstract**

Large language models can produce fluent reasoning traces whose local semantic errors propagate to an incorrect conclusion, while unconstrained self-correction may preserve, amplify, or introduce errors. Existing diffusion language models provide iterative refinement, but usually define noise as token masking or replacement rather than as errors in the reasoning process. We present Semantic Reasoning Denoising (SRD), an operatorized Markov denoising method for natural-language reasoning trajectories. SRD represents semantic noise with executable error operators that describe the error type, its location, and the corrupted and repaired propositions. Composing these operators constructs progressively noisier states. During training, the model learns to identify the semantic noise active in the current trajectory and to reconstruct the paired adjacent lower-noise state. During inference, noise-level-aware denoising repeatedly predicts an inverse operator and checks whether it is applicable, so each executed update makes a localized move toward a stable trajectory. Across six in-domain benchmarks spanning mathematics, code, knowledge, and commonsense, SRD improves the strongest same-backbone baseline by 3.2 points on average. On seven cross-dataset transfer targets, it remains competitive with Llama-3-8B-Instruct and improves the strongest Qwen3-8B baseline average by 2.9 points. Analyses of noise sources, objectives, and denoising depth further show that structured semantic-noise prediction and iterative operator execution are central to the improvement.

## 1 Introduction

Large language models (LLMs) have made remarkable progress on complex reasoning tasks, yet most approaches still focus on direct generation and lack an explicit mechanism for progressively correcting intermediate reasoning states with local errors. Diffusion models provide an instructive process-level perspective: generation proceeds through progressive corruption and reverse recovery, a structure naturally suited to describing iterative refinement from erroneous to correct reasoning (Ho et al. 2020; Sohl-Dickstein et al. 2015).

Existing diffusion language models typically define noise through masking, replacement, or other token-level perturbations, and generate text by progressively recovering the original sequence (Austin et al. 2021a; Nie et al. 2025). Such perturbations suit general text generation but do not directly capture reasoning errors, which often involve incorrect intermediate conclusions, misuse of conditions, or broken logical dependencies. Token-level diffusion therefore primarily recovers textual form rather than the underlying reasoning process.

To this end, we propose **SRD** (Semantic Reasoning Denoising), an operatorized Markov denoising method for natural-language reasoning trajectories. SRD borrows the progressive-corruption and reverse-recovery mechanisms of diffusion models, represents semantic noise with executable error operators, and composes those operators to construct a sequence of coherent corrupted reasoning states. Conditioned on a normalized process coordinate, the model learns ordered local inverse transitions that progressively recover the reasoning trajectory.

SRD comprises three stages (fig. 1): training-sample construction, diffusion-style training, and iterative inference. In the data-construction stage, we jointly use real error trajectories, synthetic noised trajectories, and interpolated trajectories between erroneous and correct ones to cover different noise levels. During training, the model learns both to identify the semantic noise in the current trajectory—represented by an executable error operator that specifies the error type, location, and corrupted and repaired content—and, conditioned on this diagnosis, to recover a lower-noise or clean reasoning state. During inference, several rounds of noise-level-aware semantic denoising progressively refine the trajectory. Each round predicts one inverse operator and checks its applicability, yielding an explicit adjacent transition instead of unconstrained regeneration.

Our experiments on 13 tasks spanning mathematical reasoning, code generation, and general understanding show that a few thousand transition pairs suffice to train the SRD adapter. Across six in-domain benchmarks, SRD averages 67.5, outperforming the strongest same-backbone baseline by 3.2 points and leading it on all six tasks; SRD is also the best same-backbone method on five. It remains competitive on Llama transfer and gives the best Qwen3 transfer average. These results support the effectiveness of an operatorized diffusion-style process over reasoning trajectories.

**Contributions.** In summary, our contributions are as follows:

- We formulate reasoning refinement as an operatorized semantic denoising process, in which executable and com-

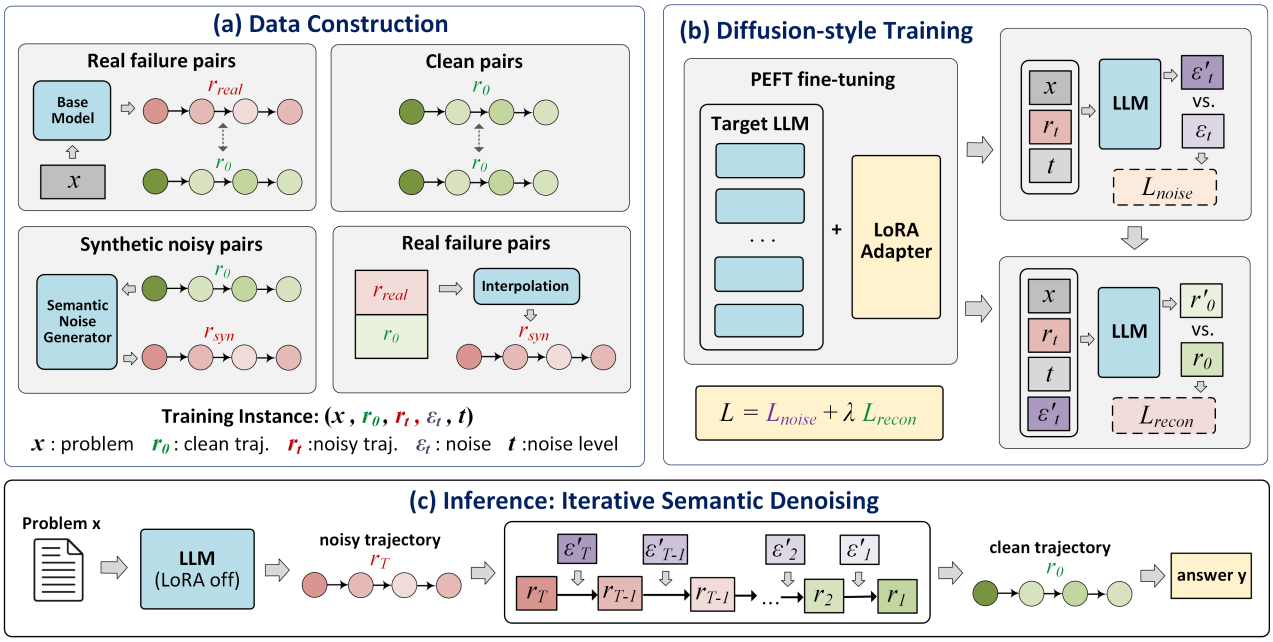


Figure 1: Overview of SRD. **(a)** Real, synthetic, and interpolated failures are aligned into executable semantic-noise operators and adjacent states. **(b)** A LoRA denoiser predicts the active operator and adjacent lower-noise state. **(c)** From the base CoT, applicability-checked inverse operators iteratively recover the final answer.

positional error operators define explicit transitions between coherent reasoning states.

- We propose SRD, which constructs trajectories at different semantic-noise levels and learns ordered local inverse transitions for iterative reasoning recovery.
- Experiments across math, code, and general reasoning demonstrate strong in-domain performance, sample efficiency, and cross-dataset transfer.

## 2 Related Work

**Reasoning generation, refinement, and correction.** Chain-of-thought prompting (Wei et al. 2022) elicits intermediate reasoning steps, while STaR (Zelikman et al. 2022) bootstraps training from model-generated rationales. Self-consistency (Wang et al. 2023) subsequently aggregates multiple sampled chains, and Self-Refine (Madaan et al. 2023) iteratively revises an answer using self-feedback. More recent methods learn correction behavior through recursive self-improvement in RISE (Qu et al. 2024), reinforcement learning in SCoRe (Kumar et al. 2025), supervised and reinforcement-learned self-verification in $S^2R$ (Ma et al. 2025), or executable pseudo-program feedback in ProgCo (Song et al. 2025). Recent analysis also distinguishes preserving correct answers from repairing incorrect ones (Yang et al. 2025a). Accordingly, our experiments compare standard SFT and STaR as generation-oriented baselines, and Self-Refine, RISE, and SCoRe as strong correction baselines. Unlike free-form response revision, SRD identifies typed semantic noise and applies an executable local transition, preserving unaffected portions of the trajectory.

**Diffusion language models and diffusion-style reasoning.** Discrete diffusion originally models text corruption through token masking or replacement (Austin et al. 2021a). Diffusion of Thoughts (Ye et al. 2024) applies token-level denoising to chain-of-thought generation. LLaDA-8B (Nie et al. 2025) and Dream-7B (Ye et al. 2025) subsequently scale discrete diffusion to general-purpose language models; we report them as diffusion-model references. Contemporaneous work broadens token corruption through Generalized Interpolating Discrete Diffusion (Von Rütte et al. 2025), moves diffusion to latent segments for controllable generation (Zhu et al. 2025), or converts pretrained autoregressive models to diffusion-style generation (Cetin et al. 2025). Most recently, DiffCoT (Cao et al. 2026) introduces a sliding window and causal noise schedule to jointly generate and retrospectively revise reasoning steps; we include it as an advanced same-backbone baseline. DiffCoT revises steps within an autoregressive chain, whereas SRD constructs coherent states through typed, executable semantic operators and learns their ordered local inverse transitions. Diffusion language models learn text generation in token space, whereas SRD learns semantic transitions in reasoning-state space; they share progressive corruption and reverse recovery but differ in both state space and learning objective.

## 3 Method

We introduce **Semantic Reasoning Denoising** (SRD), a discrete Markov process over reasoning trajectories with transitions induced by semantic error operators. Its forward process composes operators into corrupted states, and its learned re-

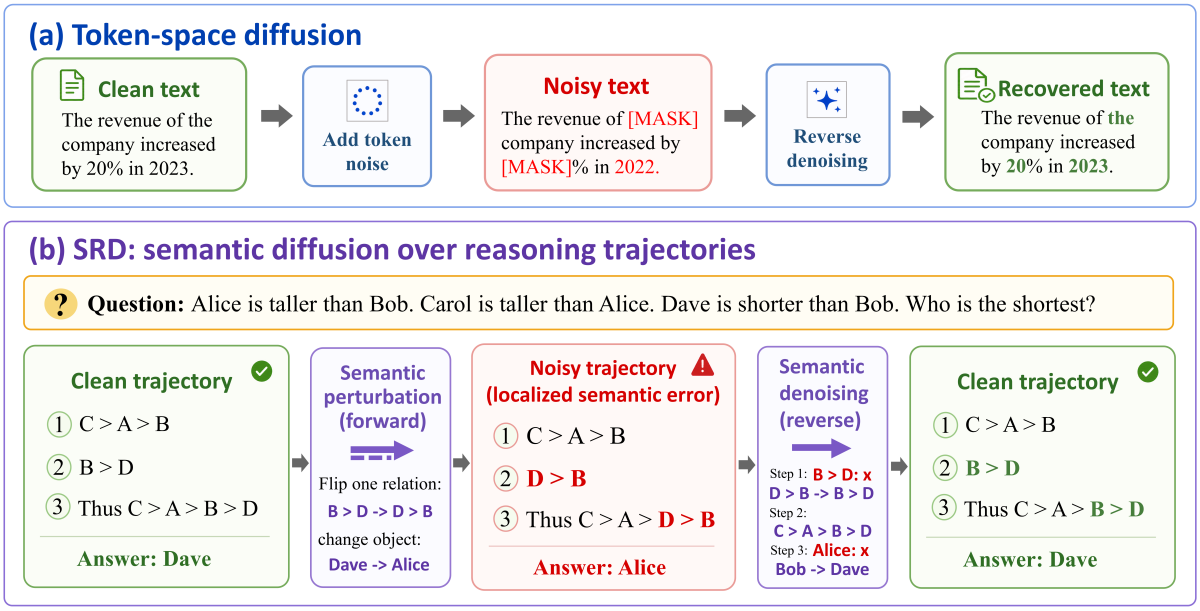


Figure 2: Token-space and semantic denoising. **(a)** Token-space diffusion corrupts surface tokens without modeling reasoning validity. **(b)** SRD applies and inverts executable semantic operators over coherent reasoning states to recover the trajectory and answer.

verse process predicts and inverts the active operator (fig. 2).

### 3.1 Preliminaries

Let a reasoning problem be a pair $(x, r_0)$, where $x$ is a question and $r_0$ a *correct* reasoning trajectory whose executed conclusion matches the ground-truth answer, $\mathcal{E}(r_0) = y^\star$. We treat $r_0$ as the clean reference state of the operatorized Markov denoising process. Let $\mathcal{Z}$ be a library of semantic error operators. An operator $z = (\kappa, \ell, u, v) \in \mathcal{Z}$ records its error type $\kappa$, step location $\ell$, original proposition $u$, and corrupted proposition $v$. The typed library covers premise insertion/deletion, relation reversal, intermediate-value substitution, step reordering, and conclusion substitution. We define the semantic-noise variable itself by this executable operator,

$$\epsilon_t := z_t = (\kappa_t, \ell_t, u_t, v_t). \tag{1}$$

Here $\epsilon_t$ denotes semantic noise and $z_t$ its structured, executable operator representation. For a state $r_t$ produced by $t$ forward transitions, the ordered prefix $z_{1:t} = (z_1, \ldots, z_t)$ represents its accumulated semantic noise, while $z_t$ is the component inverted by the next adjacent reverse transition. Its executable action $\mathcal{A}_z$ maps one coherent trajectory to another; a paired repair action $\mathcal{R}_z$ removes that corruption. We do not require every language edit to have a unique algebraic inverse, only that the recorded pair satisfies $\mathcal{R}_z(\mathcal{A}_z(r)) = r$ on constructed transitions. For two trajectories $r$ and $r_0$, define their operator distance as

$$d_{\mathcal{Z}}(r, r_0) = \min\{m : \exists z_{1:m},\ r = \mathcal{A}_{z_m} \circ \cdots \circ \mathcal{A}_{z_1}(r_0)\}. \tag{2}$$

This distance is the discrete counterpart of noise magnitude: it counts the shortest executable corruption path rather than surface-token differences.

Continuous diffusion defines a Gaussian forward process $q(r_t \mid r_0) = \mathcal{N}(r_t; \sqrt{\bar{\alpha}_t}\, r_0, (1 - \bar{\alpha}_t)\mathbf{I})$ and learns a reverse chain $p_\theta(r_{t-1} \mid r_t)$ (Ho et al. 2020; Sohl-Dickstein et al. 2015). In contrast, SRD uses compositions of operators in $\mathcal{Z}$ to define explicit semantic-state transitions, without relying on a Gaussian terminal distribution or a likelihood-based diffusion objective.

### 3.2 Forward Process

Starting from $r_0$, the forward process samples and applies one operator (or a small commuting bundle) at each step:

$$z_t \sim q_\phi(\cdot \mid r_{t-1}, x), \qquad r_t = \mathcal{A}_{z_t}(r_{t-1}), \tag{3}$$

Here, $q_\phi$ is the fixed empirical mixture induced by recovered real failures and the synthetic operator sampler; it is not optimized jointly with the denoiser. Together, these transitions define the Markov chain

$$q_\phi(r_{1:T}, z_{1:T} \mid r_0, x) = \prod_{t=1}^{T} q_\phi(z_t \mid r_{t-1}, x) \cdot \mathbf{1}[r_t = \mathcal{A}_{z_t}(r_{t-1})]. \tag{4}$$

Each primitive operator is one semantic corruption step (a commuting bundle counts as the number of its primitives). We therefore set $\tau_t = t/T$ as the normalized process coordinate of the constructed forward operator chain. It marks the relative position of a training state from the clean start to the corrupted endpoint.

A clean trajectory is corrupted from three complementary sources represented in the shared operator space:

1. **Real failures (typically high $\tau$)**: the base model's own incorrect trajectories on training questions provide the

**Algorithm 1** SRD inference for one problem $x$.

**Require:** problem $x$, maximum denoising iterations $K$, denoiser $f_\theta$

$r \leftarrow \textsc{BaseCoT}(x)$ ▷ adapter disabled: initial state $r^{(0)}$
$\tau \leftarrow 1$
**for** $k = 1$ to $K$ **do**
  $(\hat{z}, \hat{r}_{\text{adj}}) \leftarrow f_\theta(r,\, x,\, \tau)$
  **if** $\hat{z} = \varnothing$ **then**
    **break**
  **end if**
  $r' \leftarrow \mathcal{R}_{\hat{z}}(r)$
  **if** $r' = r$ **then**
    **break**
  **end if**
  $r \leftarrow r'$;  $\tau \leftarrow \max(0, \tau - 1/K)$
**end for**
**return** $\mathcal{E}(r)$ ▷ execute/extract the answer from the stable trace

closest samples of its natural error distribution. Aligning a failure with $r_0$ recovers its accumulated semantic noise as an ordered operator sequence.

2. **Synthetic corruptions (controllable $\tau$)**: operators sampled from the library perturb an intermediate result, omit a premise, or alter a relation. Their number and composition directly control the process coordinate and provide exact inverse supervision.
3. **Interpolated traces (intermediate $\tau$)**: applying only a prefix of a recovered or synthetic operator sequence constructs states between the clean trajectory and the complete erroneous trajectory.

All three sources are represented as paired supervision between a corrupted state $r_t$, its ordered noise operators $z_{1:t}$, and the correct trajectory $r_0$. States at different process coordinates simulate the intermediate stages encountered when progressively denoising a natural failure; each transition pair uses $z_t$ as the next corruption component to invert. These sources induce the training-state mixture

$$q_{\text{mix}} = \alpha q_{\text{real}} + \beta q_{\text{syn}} + \gamma q_{\text{interp}}, \tag{5}$$

where the coefficients are the empirical source proportions.

**Operator extraction and executability checks.** We split each failure and its clean trajectory into ordered reasoning steps, align them monotonically, and map each mismatch to the smallest applicable typed operator, including its location and corrupted/repaired payloads. We retain only sequences whose execution reconstructs the paired states and whose prefixes remain well formed; full construction details are given in the supplementary material. Clean ($\tau = 0$) identity pairs teach preservation. Since all states remain coherent natural-language trajectories, a LoRA adapter only needs to learn the structured mapping from corrupted states to local inverse edits.

### 3.3 Reverse Process

The reverse process learns to invert the corruption. A single denoiser $f_\theta$ —an instruction-tuned LLM backbone with low-rank adapters (Hu et al. 2022)—takes a corrupted trace $r_t$, the question $x$, and the normalized process coordinate $\tau_t$ (serialized as a special `[t]` token, mirroring instruction-guided editing (Brooks et al. 2023)). It jointly predicts the active operator $z_t$ and the supervised adjacent lower-noise state $r^\star_{t-1}$ paired with $r_t$. The superscript distinguishes this reconstruction label from the state actually produced at inference. Adjacent-state prediction is an auxiliary transition-reconstruction objective: it teaches recovery from semantic noise, whereas the predicted inverse operator alone determines the next state actually entered. Formally, the training output factorizes as

$$\begin{aligned} &p_\theta(z_t, r^\star_{t-1} \mid r_t, x, \tau_t) \\ &\quad = p_\theta(z_t \mid r_t, x, \tau_t)\, p_\theta(r^\star_{t-1} \mid r_t, x, \tau_t, z_t). \end{aligned} \tag{6}$$

The operative state transition remains $r_{t-1} = \mathcal{R}_{z_t}(r_t)$.

The model is trained with a joint supervised objective for structured operator identification and adjacent-state reconstruction:

$$\begin{aligned} \mathcal{L}_{\text{SRD}} = \mathbb{E}_{x, r_t, z_t, r^\star_{t-1}} \Big[ & \underbrace{-\log p_\theta(z_t \mid r_t, x, \tau_t)}_{\mathcal{L}_{\text{op}}\ \text{(operator prediction)}} \\ & + \lambda \underbrace{\big( -\textstyle\sum_j w_j \log p_\theta(r^{\star(j)}_{t-1} \mid r_t, x, \tau_t, z_t) \big)}_{\mathcal{L}_{\text{adj}}\ \text{(adjacent-state prediction)}} \Big]. \end{aligned} \tag{7}$$

The output sequence first specifies the current operator $z_t = (\kappa_t, \ell_t, u_t, v_t)$ and then reconstructs $r^\star_{t-1}$ conditioned on $r_t$ and $z_t$; $w_j$ upweights affected positions. Removing either loss leaves only adjacent-state reconstruction or operator diagnosis, respectively.

**Iterative denoising.** Let $p_{\text{base}}(r \mid x)$ denote the trajectory distribution induced by the frozen backbone under the fixed prompting and decoding protocol. At inference, we disable the adapter and draw the initial trajectory $r_{\text{init}} \sim p_{\text{base}}(r \mid x)$, which we denote by $r^{(0)}$. We then apply $f_\theta$ for at most $K$ iterations. Here $K$ is a maximum denoising budget, not the unknown semantic-corruption depth of the naturally generated initial trajectory. The schedule $\tau^{(k)} = 1 - k/K$ traverses the same process coordinate in reverse. At each step it predicts the current inverse operator together with an auxiliary adjacent-state reconstruction:

$$\begin{aligned} (\hat{z}^{(k)}, \hat{r}^{(k)}_{\text{adj}}) &= f_\theta(r^{(k)}, x, \tau^{(k)}), \\ r^{(k+1)} &= \mathcal{R}_{\hat{z}^{(k)}}(r^{(k)}), \qquad k = 0, \ldots, K-1. \end{aligned} \tag{8}$$

We stop on an empty operator, an unchanged state, or at the step budget, then read $\mathcal{E}(r)$.

For an exact operator posterior, reversing the terminal operator on a shortest corruption path decreases $d_{\mathcal{Z}}$ by exactly one. This property formalizes the unit-step geometry of the constructed reverse path and motivates adjacent inverse-transition supervision; its proof is given in the supplementary material.

At inference, every step after the first consumes a trajectory the denoiser produced itself rather than one drawn from the constructed training distribution, which could in principle

| Group | Method | GSM8K | MATH | HEval+ | MBPP | MMLU | PIQA | Avg. |
|---|---|---|---|---|---|---|---|---|
| Base | CoT | 77.2 | 31.0 | 52.1 | 51.3 | 63.0 | 81.9 | 59.4 |
| Learn-to-solve | SFT | 79.3 | 35.6 | 55.5 | 52.4 | 65.5 | 82.0 | 61.7 |
| | STaR | 80.6 | 39.8 | 54.7 | 52.9 | 66.3 | 84.2 | 63.1 |
| | DiffCoT | 82.1 | 39.1 | 55.9 | 56.1 | 65.4 | 87.3 | 64.3 |
| Learn-to-correct | Self-Refine | 80.4 | 34.5 | 52.8 | 51.6 | 62.7 | 83.5 | 60.9 |
| | RISE | 82.3 | 38.5 | 53.2 | 52.0 | 68.9 | 83.4 | 63.1 |
| | SCoRe | 80.6 | 36.7 | 53.5 | 55.4 | **70.1** | 86.1 | 63.7 |
| Reference | Dream-7B | 77.2 | 39.6 | 57.9 | 56.2 | 69.5 | 75.8 | 62.7 |
| | LLaDA-8B | 70.9 | 30.7 | 32.9 | 39.0 | 65.9 | 74.8 | 52.4 |
| **Ours** | **SRD** | **84.8** | **43.1** | **57.9** | **58.4** | 69.6 | **91.0** | **67.5** |

Table 1: Main in-domain results; diffusion-model rows are reference-only.

expose the model to states it never saw during training. We mitigate this by construction: the three noise sources jointly approximate the severity and diversity of the backbone's own errors (section 4.3), so a self-produced intermediate trajectory likely resembles the corruption space the denoiser was trained on. We find aggregate accuracy remains stable in practice: after the first few steps, increasing the denoising budget produces a performance plateau rather than a systematic decline (section 4.4).

# 4 Experiments

We evaluate SRD on in-domain performance, cross-dataset transfer, semantic-noise construction, sample and inference efficiency, and core ablations.

## 4.1 Main In-Domain Results

**Setup.** We evaluate GSM8K, MATH, HumanEval+, MBPP, MMLU, and PIQA (Cobbe et al. 2021; Hendrycks et al. 2021a; Liu et al. 2023; Austin et al. 2021b; Hendrycks et al. 2021b; Bisk et al. 2020). All same-backbone methods use Llama-3-8B-Instruct and the corresponding training split; because HumanEval+ has no supervised training split, we use MBPP for code adaptation. Baselines follow their official papers and code. SRD transitions $(x, r_t, z_t, r^{\star}_{t-1}, \tau_t)$ come from aligned backbone failures, executable synthetic operators, and operator-chain prefix states (section 3). All trainable models use the checkpoint with the lowest validation loss. For benchmarks with more than 1,000 test instances, every method uses the same fixed 1,024-instance subset sampled with seed 42; otherwise, we use the full test set. Test subsets are not used for model selection, and main results are averaged over five independent random seeds.

**Baselines.** We compare with three families of methods. *Learn-to-solve* baselines (SFT, DiffCoT, STaR) improve reasoning through supervised trajectories, self-generated rationales, or diffusion-styled reasoning optimization. *Learn-to-correct* baselines (Self-Refine, RISE, SCoRe) revise an existing answer or reasoning trace. Dream-7B and LLaDA-8B are included as diffusion-language-model references.

**Metrics.** We report accuracy for GSM8K, MATH, MMLU, and PIQA, and execution-based pass@1 for HumanEval+ and MBPP. Within each benchmark, all same-backbone methods use the same prompt family, answer extractor, and metric implementation.

Table 1 shows that SRD obtains the strongest overall performance among same-backbone methods, with an average score of 67.5. The strongest non-SRD baseline is DiffCoT, with an average of 64.3; SRD improves over it by 3.2 points on average, while also outperforming the strongest learn-to-correct baseline SCoRe by 3.8 points. The advantage is most pronounced on tasks where local reasoning errors directly determine the final answer: compared with STaR, SRD is 4.2 points higher on GSM8K, 3.3 points higher on MATH, 5.5 points higher on MBPP, and 6.8 points higher on PIQA. Across five independent runs per method, SRD exceeds DiffCoT by 3.1 macro-average points; all six improvements remain significant after Holm correction (see the supplementary material). On MMLU, SCoRe is slightly stronger than SRD, suggesting that factual multiple-choice questions may benefit less from iterative trajectory denoising than math, code, and commonsense reasoning. Compared with diffusion-language-model references, SRD matches Dream-7B on HumanEval+ and is stronger on GSM8K, MBPP, and PIQA.

## 4.2 Cross-Dataset Transfer

**Setup.** The second experiment evaluates whether a learned corrector transfers beyond the dataset used to build its supervision. Unlike the in-domain setting, we construct source data by transfer group: ProofWriter for BBH; StrategyQA and QASC for CSQA, WinoGrande, BoolQ, and ARC-C; and HotpotQA for SQuAD and QuAC (Tafjord et al. 2021; Suzgun et al. 2023; Geva et al. 2021; Khot et al. 2020; Talmor et al. 2019; Sakaguchi et al. 2020; Clark et al. 2019, 2018; Yang et al. 2018; Rajpurkar et al. 2016; Choi et al. 2018). SRD uses the same operator-based semantic-noise construction and joint operator/adjacent-state objective in each group. We evaluate two strong learn-to-solve baselines (DiffCoT, STaR) and two strong learn-to-correct baselines (RISE, SCoRe) on both Llama-3-8B-Instruct and Qwen3-8B (Yang et al. 2025b). For Qwen3, we disable thinking

| Backbone | Method | BBH | CSQA | WinoG. | BoolQ | ARC-C | SQuAD | QuAC | Avg. |
|---|---|---|---|---|---|---|---|---|---|
| Reference | Dream-7B | 59.4 | 80.0 | 63.0 | 45.0 | 68.0 | 66.1 | 15.9 | 56.8 |
| | LLaDA-8B | 47.4 | 71.2 | 73.5 | 41.2 | 47.5 | 59.7 | 13.2 | 50.5 |
| Llama-3 | CoT (base) | 79.3 | 74.5 | 66.0 | 78.5 | 72.0 | 83.6 | 19.7 | 67.7 |
| | STaR | 83.4 | 79.6 | 72.0 | 87.0 | 78.4 | 85.7 | 21.4 | 72.5 |
| | DiffCoT | **83.7** | 79.5 | 81.1 | 84.0 | **84.1** | 88.1 | 24.0 | 74.9 |
| | RISE | 81.1 | 80.2 | 75.7 | 85.0 | 77.6 | 82.9 | 22.4 | 72.1 |
| | SCoRe | 80.5 | 81.4 | 80.2 | 85.4 | 80.3 | 84.5 | 20.8 | 73.3 |
| | **SRD (ours)** | 83.3 | **82.6** | **82.3** | **88.0** | 83.5 | **89.2** | **25.6** | **76.4** |
| Qwen3 | CoT (base) | 77.8 | 82.5 | 67.5 | 89.5 | 90.5 | 85.5 | 17.3 | 72.9 |
| | STaR | 81.4 | 85.0 | 73.5 | 92.1 | 90.2 | 86.2 | 24.1 | 76.1 |
| | DiffCoT | 79.2 | 86.4 | 82.3 | 91.6 | 90.8 | 87.4 | 23.6 | 77.3 |
| | RISE | 79.1 | 85.5 | 80.8 | 90.7 | 88.9 | 86.9 | 25.7 | 76.8 |
| | SCoRe | 80.4 | 83.9 | 81.6 | 91.5 | 91.5 | 87.6 | 24.3 | 77.3 |
| | **SRD (ours)** | **85.2** | **89.8** | **84.7** | **92.4** | **93.7** | **89.0** | **26.4** | **80.2** |

Table 2: Cross-dataset transfer results; reading tasks report F1.

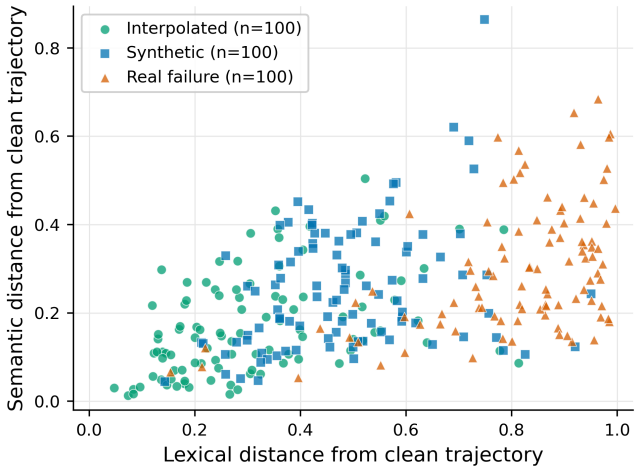


Figure 3: Semantic-noise sources in MATH ($n = 100$ each).

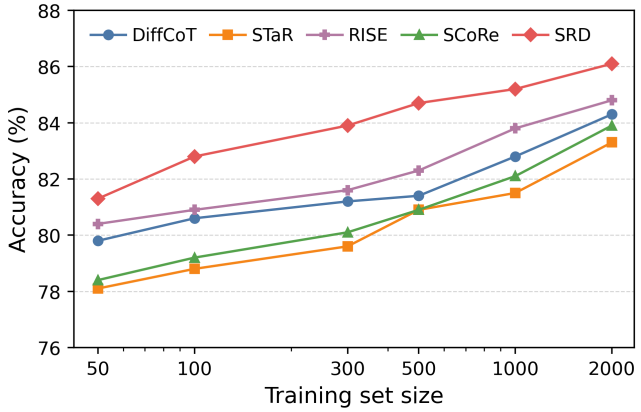


Figure 4: GSM8K accuracy versus training size (log-scale $x$-axis).

mode so that all methods use the same answer extractor.

Table 2 evaluates whether the learned correction behavior transfers beyond the dataset on which it was trained. On Llama-3, SRD improves the base average from 67.7 to 76.4 (+8.7 points), with particularly large gains on WinoGrande and ARC-C. SRD is best on CSQA, WinoGrande, BoolQ, SQuAD, and QuAC, and obtains the highest Llama average. On Qwen3-8B, SRD leads every reported target and improves the base average from 72.9 to 80.2 (+7.3 points). This two-backbone pattern supports cross-dataset and cross-backbone transfer of the learned correction behavior.

### 4.3 Semantic-Noise Analysis and Sample Efficiency

We analyze SRD's semantic corruption space and sample efficiency using the construction in section 4.1.

Figure 3 shows how the three noise sources populate the corruption space. Lexical distance is one minus `SequenceMatcher` similarity; semantic distance is one minus cosine similarity between MiniLM embeddings. Interpolated trajectories are the mildest and synthetic corruptions have moderate, controllable severity. The model's real failures are the most diverse, with the heaviest tail toward large lexical and semantic distances. Their overlapping ranges create a gradual severity continuum rather than disconnected clusters: interpolation and synthetic corruption stabilize mild reverse transitions, while real failures better approximate the natural errors encountered during inference.

Figure 4 shows that SRD is consistently more sample-efficient than the correction and self-training baselines. Its advantage is already visible with 50 training examples and persists as the training set grows to 2,000, indicating that learning structured semantic reverse transitions provides a useful inductive bias rather than merely benefiting from additional data.

| $K$ | Calls | Gen. tok. | Latency (s) | Tok./fwd. |
|---|---|---|---|---|
| 1 | 2 | 304 | 5.7 | 1.7 |
| 2 | 3 | 445 | 6.4 | 2.5 |
| 4 | 5 | 735 | 7.1 | 4.2 |
| 8 | 9 | 1366 | 9.5 | 5.4 |
| 16 | 17 | 2378 | 11.2 | 8.0 |

Table 3: MATH inference cost by denoising budget.

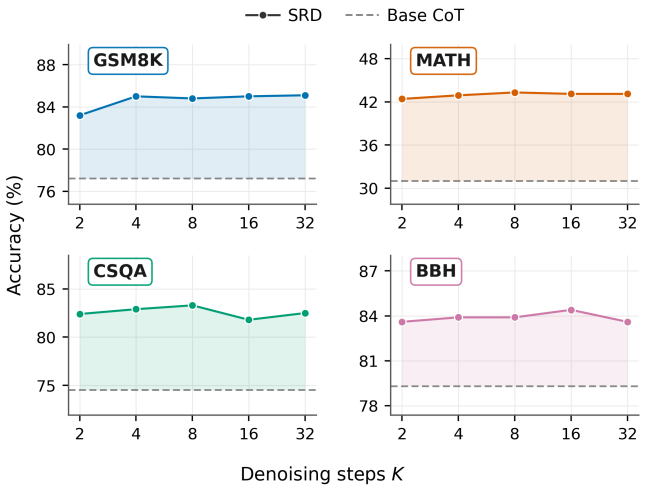


Figure 5: Accuracy versus denoising budget $K$ on GSM8K, MATH, CSQA, and BBH; dashed lines show base CoT.

### 4.4 Inference Efficiency

This analysis measures how the cost of SRD scales with the number of denoising steps, using the same corrector and denoising procedure as section 4.1 and varying only the number of steps. Measurements use BF16 precision and greedy decoding on a single NVIDIA RTX 4090 D GPU.

Table 3 reports the cost of SRD across denoising budgets. As $K$ increases from 1 to 16, the generated tokens grow by $7.8\times$ but the total latency grows only $2.0\times$. Self-speculative decoding verifies matching draft tokens in parallel and reuses shared-prefix KV caches, amortizing repeated decoding of unchanged content and keeping iterative denoising practical.

Figure 5 shows that SRD improves over base at every tested $K$, with most gains realized in the first few steps. Larger budgets add cost (table 3) with little accuracy benefit, supporting a small default $K$.

### 4.5 Core Ablations

We ablate SRD's noise sources, training objective, and supervision target, changing only the component named in each row.

Table 4 shows that full SRD is the most robust configuration. Removing a noise source generally hurts, with synthetic noise contributing most overall. Operator-only and adjacent-state-only objectives weaken all four tasks, confirming their complementarity; answer-only and full-CoT supervision also underperform semantic trajectory denoising.

| Variant | GSM8K | MATH | CSQA | BBH |
|---|---|---|---|---|
| **Full SRD** | **84.8** | 43.1 | **82.6** | **83.3** |
| w/o real failures | 83.1 | 43.6 | 81.7 | 82.7 |
| w/o synthetic | 80.4 | 41.4 | 81.2 | 82.1 |
| w/o interpolation | 82.7 | **44.5** | 81.6 | 82.1 |
| operator-only | 79.1 | 40.8 | 80.9 | 81.8 |
| adjacent-state-only | 80.6 | 40.3 | 81.2 | 82.4 |
| answer-only supervision | 81.4 | 41.5 | 80.5 | 81.6 |
| full-CoT supervision | 77.5 | 40.8 | 81.9 | 83.1 |

Table 4: Core ablations (best per column in **bold**).

**Compositional operator diagnosis.** We test one-, two-, and three-operator corruptions across all six in-domain benchmarks; full metrics and sample counts appear in the supplementary material. SRD retains strong parsing and operator recognition as depth increases. With three operators, execution reaches 81.4% on GSM8K and 85.7% on PIQA, while answer recovery remains 77.0% and 84.6%, respectively. Performance declines more on MATH and code, identifying localization as the main bottleneck for longer compositions.

## 5 Conclusion

We presented SRD, an operatorized Markov denoising method that learns executable inverse transitions over reasoning trajectories. SRD improves the strongest same-backbone baseline by 3.2 points and transfers across Llama-3 and Qwen3. Ablations and compositional diagnostics support joint operator and adjacent-state supervision, while identifying localization as the bottleneck for longer chains.

**Limitations and future work.** Evaluation is limited to 8B backbones and fixed operators, motivating larger models and adaptive operators for long or ambiguous error chains.

## References


Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588*, 2022.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, 2021.

Tim Brooks, Aleksander Holynski, and Alexei A. Efros. InstructPix2Pix: Learning to follow image editing instructions. In *Proceedings of CVPR*, 2023.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. PAL:

Program-aided language models. In *Proceedings of ICML*, 2023.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. In *Advances in Neural Information Processing Systems*, 2023.

Abhimanyu Dubey et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *Proceedings of ICLR*, 2022.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with Gumbel-Softmax. In *Proceedings of ICLR*, 2017.

Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. In *Proceedings of ICLR*, 2014.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *Proceedings of ICLR*, 2024.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *Proceedings of ICLR*, 2019.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, 2023.

Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *Proceedings of ICLR*, 2017.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. In *Advances in Neural Information Processing Systems*, 2025.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of CVPR*, 2022.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of ICML*, 2015.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *Proceedings of ICLR*, 2023.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, 2022.

Aviral Kumar et al. Training language models to self-correct via reinforcement learning. In *Proceedings of ICLR*, 2025.

Yuxiao Qu, Tianjun Zhang, Naman Garg, and Aviral Kumar. Recursive introspection: Teaching language model agents how to self-improve. In *Advances in Neural Information Processing Systems*, 2024.

Jiacheng Ye et al. Diffusion of thoughts: Chain-of-thought reasoning in diffusion language models. In *Advances in Neural Information Processing Systems*, 2024.

Shidong Cao, Hongzhan Lin, Yuxuan Gu, Ziyang Luo, and Jing Ma. DiffCoT: Diffusion-styled chain-of-thought reasoning in LLMs. In *Findings of the Association for Computational Linguistics: ACL 2026*, 2026.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. STaR: Bootstrapping reasoning with reasoning. In *Advances in Neural Information Processing Systems*, 2022.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7B: Diffusion large language models. *arXiv preprint arXiv:2508.15487*, 2025.

Ruotian Ma, Peisong Wang, Cheng Liu, Xingyan Liu, Jiaqi Chen, Bang Zhang, Xin Zhou, Nan Du, and Jia Li. $S^2R$: Teaching LLMs to self-verify and self-correct via reinforcement learning. In *Proceedings of ACL*, 2025.

Xiaoshuai Song, Yanan Wu, Weixun Wang, Jiaheng Liu, Wenbo Su, and Bo Zheng. ProgCo: Program helps self-correction of large language models. In *Proceedings of ACL*, 2025.

Zhe Yang, Yichang Zhang, Yudong Wang, Ziyao Xu, Junyang Lin, and Zhifang Sui. Confidence vs. critique: A decomposition of self-correction capability for LLMs. In *Proceedings of ACL*, 2025.

Edoardo Cetin, Tianyu Zhao, and Yujin Tang. Large language models to diffusion finetuning. In *Proceedings of ICML*, 2025.

Dimitri Von Rütte, Janis Fluri, Yuhui Ding, Antonio Orvieto, Bernhard Schölkopf, and Thomas Hofmann. Generalized interpolating discrete diffusion. In *Proceedings of ICML*, 2025.

Xiaochen Zhu, Georgi Karadzhov, Chenxi Whitehouse, and Andreas Vlachos. Segment-level diffusion: A framework for controllable long-form generation with diffusion language models. In *Proceedings of ACL*, 2025.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Advances in Neural Information Processing Systems*, 2021.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by ChatGPT really correct? Rigorous evaluation of large language models for code generation. In *Advances in Neural Information Processing Systems*, 2023.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *Proceedings of ICLR*, 2021.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of AAAI*, 2020.

Mirac Suzgun et al. Challenging BIG-Bench tasks and whether chain-of-thought can solve them. In *Findings of ACL*, 2023.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of NAACL-HLT*, 2019.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial Winograd schema challenge at scale. In *Proceedings of AAAI*, 2020.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of NAACL-HLT*, 2019.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? Try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of EMNLP*, 2016.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. QuAC: Question answering in context. In *Proceedings of EMNLP*, 2018.

Oyvind Tafjord, Bhavana Dalvi Mishra, and Peter Clark. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of ACL-IJCNLP*, 2021.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did Aristotle use a laptop? A question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 2021.

Tushar Khot, Peter Clark, Michal Guerquin, Peter Jansen, and Ashish Sabharwal. QASC: A dataset for question answering via sentence composition. In *Proceedings of AAAI*, 2020.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of EMNLP*, 2018.

An Yang et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.