Constraints-Guided Diffusion Reasoner for Neuro-Symbolic Learning

Xuan Zhang[1,2], **Zhijian Zhou**[1,2], **Weidi Xu**[4], **Yanting Miao**[5], **Chao Qu**[1,3*], **Yuan Qi**[1,3*]
[1]Fudan University
[2]Shanghai Innovation Institute
[3]Shanghai Academy of Artificial Intelligence for Science
[4]INFTECH
[5]University of Waterloo
xuanzhang24@m.fudan.edu.cn, quchao@fudan.edu.cn

Abstract

Enabling neural networks to learn complex logical constraints and fulfill symbolic reasoning is a critical challenge. Bridging this gap often requires guiding the neural network's output distribution to move closer to the symbolic constraints. While diffusion models have shown remarkable generative capability across various domains, we employ the powerful architecture to perform neuro-symbolic learning and solve logical puzzles. Our diffusion-based pipeline adopts a two-stage training strategy: the first stage cultivates basic reasoning abilities, while the second emphasizes systematic learning of logical constraints. To impose constraints on neural outputs in the second stage, we formulate the diffusion reasoner as a Markov decision process and innovatively fine-tune it with an improved proximal policy optimization algorithm. We utilize a rule-based reward signal derived from the logical consistency of neural outputs and adopt a flexible strategy to optimize the diffusion reasoner's policy. We evaluate our methodology on some classical symbolic reasoning benchmarks, including Sudoku, Maze, pathfinding and preference learning. Experimental results demonstrate that our approach achieves outstanding accuracy and logical consistency.

Code — https://github.com/dd88s87/DDReasoner
Appendices — https://arxiv.org/pdf/2508.16524

Introduction

Despite the rapid advance of deep learning, it is still challenging for deep neural networks to solve problems with complex and hard constraints such as symbolic reasoning. For instance, the output of a neural network may achieve good approximation over most regions; however, local prediction errors can still result in outputs that violate the constraints, thus failing to obtain a correct solution. To address this issue, there has been an increasing amount of neuro-symbolic learning methods to inject domain knowledge constraints and improve the performance of complex reasoning. These methods primarily involve modifying the loss function (Xu et al. 2018; Wang and Pan 2019; Xie et al. 2019; Li et al. 2019; Ahmed, Chang, and den Broeck 2023), incorporating constraint layers (Giunchiglia and Lukasiewicz 2021; Ahmed et al. 2022a), and integrating logic engines or algorithmic components (Wang et al. 2019; Cornelio et al. 2023; Li et al. 2023; Agarwal, Shenoy, and Mausam 2021; Yang, Ishay, and Lee 2020; Petersen et al. 2021), among others. However, these methods may suffer from poor performance or expensive time consumption.

Parallel to this, diffusion models (Ho, Jain, and Abbeel 2020; Sohl-Dickstein et al. 2015; Nichol and Dhariwal 2021) have emerged as an outstanding paradigm of generative models and demonstrated immense potential in many related downstream tasks (Saharia et al. 2022; Dhariwal and Nichol 2021; Rombach et al. 2022; Ho et al. 2022; He et al. 2023; Janner et al. 2022; Chi et al. 2023). By learning how to denoise from noisy and corrupted samples, diffusion models can generate high-quality samples approximating the data distribution. Due to the inherent iterative and stochastic characteristics, some recent studies apply diffusion architectures to reasoning or planning tasks (Janner et al. 2022; Chi et al. 2023; Ye et al. 2024, 2025; Zhao et al. 2025; Du, Mao, and Tenenbaum 2024; Zhang et al. 2025; Fan et al. 2023; Black et al. 2024), which are primarily combined with text, visual information, or robot trajectories.

In this work, we apply denoising diffusion probabilistic models (DDPMs) to perform symbolic reasoning, i.e., **DDReasoner**. We employ a two-stage pipeline to train our neural network: (1) Basic Supervised Learning (SL): Other related efforts focused on boosting the performance of diffusion models in this stage (Du, Mao, and Tenenbaum 2024; Zeng et al. 2024). In this work, we use the simplest form of masked DDPM training to enable DDReasoner to preliminarily possess the capability to solve logical puzzles; (2) A Novel Application of Reinforcement Learning (RL): Generated solutions with minor errors in a few positions, which lead to the violation of overall constraints, may exhibit a minimal loss value during the SL phase, indicating that DDReasoner may fail to identify them as suboptimal outputs. Therefore, we introduce a novel application of RL. By imposing hard logical constraints on the reward function, we further guide DDReasoner to internalize constraints and refine its reasoning trajectory. During RL training, following the method of Black et al. (2024), we formulate the iterative denoising process as a Markov Decision Process (MDP), in which the termination state represents DDReasoner's pre-

*Corresponding Authors.

dicted solution. We utilize a rule-based reward function derived from the logical consistency of the generated solution and employ a dynamic and efficient fine-tuning strategy to optimize the policy. Despite previous explorations about logically-constrained RL (Fu and Topcu 2014; Hasanbeig, Abate, and Kroening 2019), we combine it with the representation capabilities of deep neural networks and develops this paradigm that operates effectively in low-dimensional discrete spaces while also offering transferability to high-dimensional spaces. To the best of our knowledge, our work is the first to utilize RL techniques for training diffusion models to solve symbolic logical reasoning tasks.

We evaluate our methodology on some symbolic reasoning benchmarks, including Sudoku, Maze, simple path prediction & preference learning and Minimum-cost path finding. Experimental results indicate that our integration of RL leads to a significant improvement in the performance of DDReasoner trained using supervised learning. Experiments on Maze also reveal that our RL method unlocks the full potential of a foundational model, enabling it to achieve perfect accuracy while maintaining high efficiency regarding the parameter count and data requirements. Our cross-dataset and varying-size experiments, detailed in Appendix B.4, confirm the superior generalizability of our RL approach.

Our contributions can be summarized as follows: (1) We innovatively employ reinforcement learning in our training pipeline, enabling diffusion models to internalize hard logical constraints; (2) We evaluate our methodology on some symbolic reasoning benchmarks. The outstanding performance demonstrates that our constraints-guided RL method facilitates the application of diffusion models to symbolic reasoning tasks that are based on constraint satisfaction.

Related Work and Motivation

Neuro-Symbolic Learning

Despite the rapid advancement of deep learning, it remains challenging for deep neural networks to solve certain classes of problems, particularly those requiring the ability to apply logical rules to analyze information, identify connections, and make consistent decisions, i.e., symbolic reasoning.

There has been a lot of recent research regarding the imposition of constraints in neural networks to bridge the gap between deep neural networks and logical constraints, i.e., neuro-symbolic learning. A common class of work is modifying the loss function to quantify the level of disagreement with logical constraints and encourage the neural output to better satisfy the given constraints (Xu et al. 2018; Wang and Pan 2019; Xie et al. 2019; Li et al. 2019; Ahmed, Chang, and den Broeck 2023). Some work added a layer to the neural network to manipulate the output closer to the logical constraints (Giunchiglia and Lukasiewicz 2021; Ahmed et al. 2022a). However, these supervised learning-based methods frequently lead to challenges in generalization to adhere to constraints across unseen problems. Some other approaches attempt to integrate symbolic solvers or algorithmic components, which guarantees nearly perfect correctness, into deep learning architectures (Wang et al. 2019; Cornelio et al. 2023; Li et al. 2023; Agarwal, Shenoy, and Mausam 2021; Yang, Ishay, and Lee 2020; Petersen et al. 2021). In particular, Wang et al. (2019) introduced a differentiable maximum satisfiability (MAXSAT) solver into a traditional convolutional architecture. Cornelio et al. (2023) constructed a pipeline consisting of a neural solver, a hard-attention mask predictor and a symbolic solver.

Although integrating with a symbolic solver or algorithm yielded favorable results on corresponding tasks, they do not enable neural networks to truly acquire the capability to solve such problems independently. In our work, we aim to apply the diffusion model as a reasoner to solve complex symbolic reasoning tasks that require strict constraints. We utilize task-specific symbolic solvers or computational algorithms only once during the initial ground truth generation. Throughout the training process, these solvers or algorithms do not need to be re-executed; At inference time, our diffusion reasoner maintains a commendable level of accuracy and consistency without the need for auxiliary modules.

Diffusion Models as Sequential Decision-Makers

Diffusion models (Sohl-Dickstein et al. 2015; Ho, Jain, and Abbeel 2020; Nichol and Dhariwal 2021) have emerged as important generative models for images (Saharia et al. 2022; Dhariwal and Nichol 2021), videos (Ho et al. 2022; He et al. 2023), robotic control (Janner et al. 2022; Chi et al. 2023), and many other modalities. These models work by gradually adding noise to a data distribution in a forward process until it becomes a simple distribution, then learning to reverse the process step by step to recover original data, i.e., denoising.

Due to the inherent property of iterative refinement, the denoising process can be regarded as a sequential decision-making problem. Recent approaches conceptualized a diffusion model as a reasoner or planner, especially in the domain of natural language reasoning (Ye et al. 2024, 2025; Zhao et al. 2025), robotic control (Janner et al. 2022; Chi et al. 2023) and visual generation (Fan et al. 2023; Black et al. 2024; Fan and Lee 2024; Wallace et al. 2023; Gupta et al. 2025). In the field of visual generation, some work has emerged that employs reinforcement learning to fine-tune image synthesis processes. Fan and Lee (2024) introduced a policy gradient-based algorithm to optimize the generated distribution, allowing the DDPM sampler to explore more efficient sampling trajectories. Fan et al. (2023) treated the denoising process as a Markov Decision Process (MDP) and proposed DPOK, a KL-regularized policy gradient algorithm to align text-to-image diffusion models with human preferences. Black et al. (2024) proposed DDPO and applied Proximal Policy Optimization (PPO) (Schulman et al. 2017) to update the policy, while Wallace et al. (2023) adapted Direct Preference Optimization (DPO) (Rafailov et al. 2023) to diffusion models. More recently, Gupta et al. (2025) proposed LOOP, integrating the respective advantages of PPO and RLOO (Ahmadian et al. 2024).

Although some studies have attempted to employ continuous or discrete diffusion models to perform complex symbolic reasoning and planning (Du, Mao, and Tenenbaum 2024; Zhang et al. 2025; Zeng et al. 2024; van Krieken et al. 2025; Suresh et al. 2025), the integration of RL techniques remains unexplored. To analyze the effectiveness of

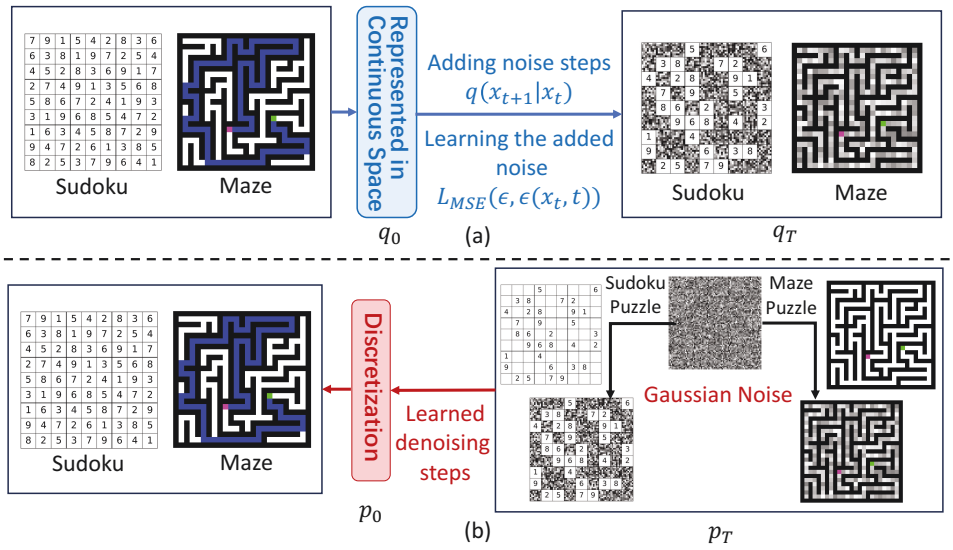

Figure 1: Overview: examples of DDReasoner solving Sudoku and Maze puzzles. (a) Supervised learning. (b) At inference time (greedy deterministic sampling). $q_0, ..., q_T$ and $p_T, ..., p_0$ are all representations in the continuous space, but for illustrative purposes, we render the digits at each cell (Sudoku) and walls (Maze) in q_T and p_T in a discrete format—representing the discretizations of distributions in the continuous space—while the regions with noised distributions are visualized as noise.

RL methods in addressing such tasks, we adopt a flexible and efficient policy optimization algorithm to fine-tune the diffusion reasoner, enabling it to internalize logical constraints and enhancing its performance at inference time.

Preliminaries

Diffusion Models

The generative process of a denoising diffusion probabilistic model (DDPM) (Ho, Jain, and Abbeel 2020) relies on the sampler $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ to reverse the forward process. It is parameterized by a neural network $\epsilon_\theta(\mathbf{x}_t, t)$, which predicts the added noise $\tilde{\epsilon}(\mathbf{x}_0, t)$ that transforms $\mathbf{x}_0$ to $\mathbf{x}_T$ with the following objective:

$$\mathcal{L}_{\text{DDPM}}(\theta) = \mathbb{E}_{\mathbf{x}_0,t,\mathbf{x}_t}\left[\|\tilde{\epsilon}(\mathbf{x}_0, t) - \epsilon_\theta(\mathbf{x}_t, t)\|^2\right] \quad (1)$$

Sampling starts from a pure Gaussian noise $p(\mathbf{x})$ by iteratively denoising following $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ to produce a trajectory $\{\mathbf{x}_T, \mathbf{x}_{T-1}, ..., \mathbf{x}_0\}$. The sampler at each step is expressed as:

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}|\mu_t(\mathbf{x}_t, \epsilon_\theta(\mathbf{x}_t, t)), \sigma_t^2\mathbf{I}) \quad (2)$$

Above, μ_t is a function that maps $\mathbf{x}_t$ and ϵ_θ to the mean timestep $t-1$, and σ_t^2 is the variance term.

Markov Decision Processes and Reinforcement Learning

Reinforcement learning (RL) is a branch of machine learning where an agent learns to make sequential decisions by interacting with an environment to maximize cumulative rewards (Sutton and Barto 2018). It is typically modeled as a Markov Decision Process (MDP) defined by a tuple: $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \rho_0, P, R)$ with states $s\epsilon\mathcal{S}$, actions $a\epsilon\mathcal{A}$, initial state distribution ρ_0, transition probabilities P and reward function R. At each timestep t, the agent observes a state $s_t\epsilon\mathcal{S}$, takes an action $a_t \sim \pi(a_t|s_t)\epsilon\mathcal{A}$, transitions to a new state $s_{t+1} \sim P(s_{t+1}|s_t, a_t)$ and receives a reward $R(s_t, a_t)$. As the agent acts in the MDP according to the policy π, it generates $\tau = (s_0, a_0, s_1, a_1, ..., s_T, a_T)$, a sequence of states and actions, i.e., trajectory. We aim to optimize the expected cumulative reward, discounted by a function $\gamma(\cdot)$, over trajectories sampled from its policy:

$$\mathcal{J}_{RL} = \mathbb{E}_{\tau\sim p(\tau|\pi)}\left[\Sigma_{t=0}^{T}\gamma(t)R(s_t, a_t)\right] \quad (3)$$

Among various approaches to RL, policy gradient methods directly optimize the policy π by computing gradients of expected reward with respect to policy parameters. While standard approaches often suffered from high variance and instability, Schulman et al. (2017) proposed Proximal Policy Optimization (PPO) to improve training stability. PPO-CLIP is the most commonly used method, introducing a clipped surrogate objective to limit the deviation between the new and old policies during updates. The clipped objective is expressed as:

$$L^{\text{PPO}}(\theta) = \mathbb{E}_t\left[\min\left(I_t(\theta)\hat{A}_t, \text{clip}\left(I_t(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_t\right)\right],$$
$$\text{where} \quad I_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)} \quad (4)$$

Above, π_θ and π_{old} is the current and previous policy respectively, and $\hat{A}_t$ is the advantage estimate which describes how much better or worse action a_t is than what the current policy would average do in state s_t. More recently, Group Relative Policy Optimization (GRPO) (Shao et al. 2024) uses group statistics to derive advantages. For each question q, GRPO samples a group of outputs from the old policy and then computes the normalized reward in each group as the advantage to update the policy model.

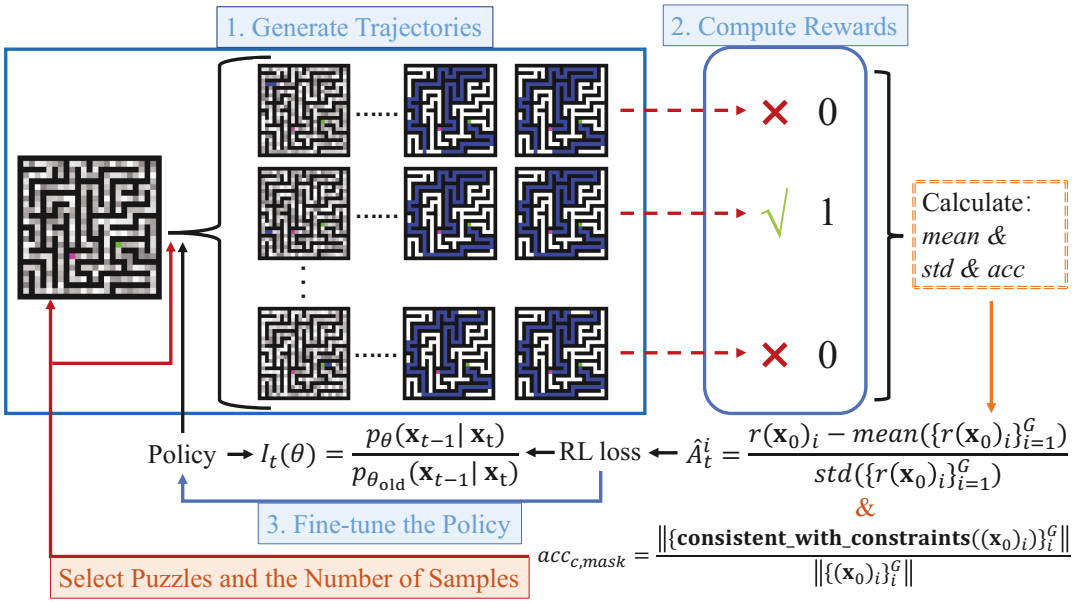

Figure 2: Overall workflow of DDReasoner's RL training.

Methodology

In this section, we introduce our diffusion-based pipeline for symbolic reasoning and our constraints-guided reinforcement learning technique to fine-tune the diffusion reasoner.

Denoising Diffusion for Symbolic Reasoning

We employ a diffusion model to perform symbolic reasoning, i.e., **DDReasoner**. After logical puzzles are represented as distributions in the continuous space (i.e., $p_{\text{puzzle}}(\mathbf{x}_T)$), DDReasoner converts them to a probability distribution over possible complete solutions. It generates samples from a masked distribution $p(\mathbf{x}_T) \triangleq mask \odot p_{\text{puzzle}}(\mathbf{x}_T) + (1 - mask) \odot \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$, where some positions (i.e., where $mask == 1$) are initially given as hints of the puzzle, and others are sampled from a Gaussian distribution. During the sampling process, observed hints remain unchanged, and other positions are denoised step-by-step to recover the distribution of a complete solution:

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}|((1 - mask) \odot \mu_t(\mathbf{x}_t, \epsilon_\theta(\mathbf{x}_t, t)) + mask \odot c), ((1 - mask) \odot \sigma_t^2)\mathbf{I}), \quad (5)$$

in which $c \sim p_{\text{puzzle}}(\mathbf{x}_T)$ is the known information from the puzzle. In the final step, x_0 in the continuous space is discretized to produce the predicted solution.

We first use masked DDPM training to enable DDReasoner to preliminarily possess the capability to solve logical puzzles. We progressively add noise to unmasked positions in the complete solution distribution to obtain a noisy representation. DDReasoner is trained to predict the added noise:

$$\mathcal{L}_{\text{SL}}(\theta) = \mathbb{E}_{\mathbf{x}_0,t,\mathbf{x}_t,c,mask}\Big[\| \tilde{\boldsymbol{\epsilon}}(\mathbf{x}_0, t) - \boldsymbol{\epsilon}_\theta((mask \odot c + (1 - mask) \odot \mathbf{x}_t), t)\|^2\Big], \quad (6)$$

where $c \sim p_{\text{puzzle}}(\mathbf{x}_T)$ and $mask$ indicates the observed position in c. At inference time, we adopt a deterministic greedy sampling strategy: $\mathbf{x}_{t-1} = mask \odot c + (1 - mask) \odot \mu_t(\mathbf{x}_t, \epsilon_\theta(\mathbf{x}_t, t)))$. The process of supervised learning and inference of DDReasoner is illustrated in Figure 1.

Constraints-Guided Policy Optimization

Following the supervised learning phase, we employ a flexible constraints-guided policy optimization technique for enabling the diffusion reasoner to internalize hard constraints.

A Denoising Diffusion MDP Given a puzzle denoted by $\{c, mask\}$, the pretrained DDReasoner generates a sample distribution $p_\theta(\mathbf{x}_0|c, mask)$ through a fixed sampling process. Considering the sequential property of the denoising process, we regard it as a multi-step MDP following Black et al. (2024), which is defined by:

$$s_t \triangleq (\mathbf{x}_t, t) \quad a_t \triangleq \mathbf{x}_{t-1} \quad \pi(a_t|s_t) \triangleq p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$$

$$P(s_{t+1}|s_t, a_t) \triangleq (\delta_{t-1}, \delta_{\mathbf{x}_{t-1}})$$

$$R(s_t, a_t) \triangleq \begin{cases} r(\mathbf{x}_0) & \text{if } t = 0 \\ 0 & \text{otherwise} \end{cases}$$

Above, δ_y is the Dirac delta distribution with non zero density only at y, and r is the reward signal.

The denoising sequence consists of $\mathcal{T}$ timesteps, after which the process transitions to a termination state representating DDReasoner's predicted solution to the given puzzle. The cumulative reward of each trajectory in this MDP is equal to $r(x_0)$, which aligns with the goal of DDReasoner's RL training. Therefore, in conjunction with Equation (3), the objective function is defined as:

$$\mathcal{J}_{\text{RL}} = \mathbb{E}_{\mathbf{x}_0 \sim p_\theta(\mathbf{x}_0|c,mask)}[r(\mathbf{x}_0)] \quad (7)$$

dataset name	# hints[*min-max*]	size	# of solutions	main challenge
big kaggle	33.82 [29-37]	100,000	unique	scaling
minimal 17	17.00 [17-17]	49,158	unique	minimal number of hints
multiple sol	34.75 [34-35]	10,000	multiple	2 or more solutions
satnet data	36.22 [31-42]	10,000	unique	-

Table 1: Summarization of Sudoku datasets. *min, max* separately represent the minimum and maximum of filled cells in a board.

We use trajectories $\tau = (\mathbf{x}_T, \mathbf{x}_{T-1}, ..., \mathbf{x}_0)$ collected during the sampling process to update model parameters via gradient descent. For multiple optimization steps per data collection round, we use an importance sampling estimator (Kakade and Langford 2002) to estimate the policy gradient:

$$\nabla_\theta \mathcal{J}_{\text{RL}} = \mathbb{E}\left[\sum_{t=0}^{T} \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{p_{\theta_{\text{old}}}(\mathbf{x}_{t-1}|\mathbf{x}_t)} \nabla log p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) r(\mathbf{x}_0)\right] \tag{8}$$

Constraint-Based Reward Modeling The consistency of neural predictions is crucial for neuro-symbolic learning tasks. To guide the generated solution to better satisfy hard logical constraints, we use the validity of the final generated solution as the outcome reward with the following rule:

$$r(\mathbf{x}_0) = \begin{cases} 1 & \text{if } \textbf{consistent\_with\_constraints}(\mathbf{x}_0) \\ 0 & \text{otherwise} \end{cases}, \tag{9}$$

where $\textbf{consistent\_with\_constraints}(\mathbf{x}_0) =$

$$\begin{cases} \textbf{equivalent}(\textbf{Discretize}(\mathbf{x}_0), a) & \text{if } \textbf{unique}(a) \\ \textbf{check\_rules}(\textbf{Discretize}(\mathbf{x}_0)) & \text{otherwise} \end{cases} \tag{10}$$

The ground-truth answer a is acquired from a task-specific symbolic solver or computational algorithm in advance, so the computational cost of checking $\mathbf{x}_0$ and a is negligible if a is unique. For tasks that may admit multiple consistent solutions, the computational complexity of this verification based on constraint rules is nevertheless substantially lower than that of solving the problem from scratch.

Group-Based Dynamic Sampling Following Black et al. (2024) and Shao et al. (2024), we develop a group-based dynamic sampling method to optimize the diffusion reasoner's denoising process. For each puzzle representation $\{c, mask\} \epsilon D$, DDReasoner samples a group of trajectories $\{\tau_1, \tau_2, ..., \tau_G\}$ from the old policy $p_{\theta_{\text{old}}}$. The cumulative reward of trajectory τ_i is equivalent to $r(\mathbf{x}_0)_i$, so the estimated advantage of the i-th trajectory is calculated by normalizing the group-level rewards $\{r(\mathbf{x}_0)_i\}_{i=1}^G$:

$$\hat{A}_t^i = \frac{r(\mathbf{x}_0)_i - mean(\{r(\mathbf{x}_0)_i\}_{i=1}^G)}{std(\{r(\mathbf{x}_0)_i\}_{i=1}^G)}, \tag{11}$$

which quantifies the relative level of constraint satisfaction of the i-th trajectory compared to the average. During training, we record its average solved rate by calculating $acc_{c,mask} = \frac{\|\{\textbf{consistent\_with\_constraints}((\mathbf{x}_0)_i)\}_i^G\|}{\|\{(\mathbf{x}_0)_i\}_i^G\|}$. In the initial epoch, G of each puzzle is set to G_{initial}; In subsequent iterations, DDReasoner dynamically adjusts the selection of puzzles and the number of samples per puzzle based on previous $\{acc_{c,mask}\}_D$. We retain only the unsolved problems from the previous epoch. Additionally, at the start of a new epoch, we utilize the records from the previous round of training and adjust the number of samples allocated to each puzzle accordingly. For more challenging puzzles (i.e., those with lower solution accuracy), we increase the number of samples (i.e., assign $G\ \epsilon[G_{\text{initial}}, \gamma * G_{\text{initial}}]$, where $\gamma > 1$ is a scaling factor) to encourage the model to explore a broader range of potential solution paths, with the goal of discovering trajectories that successfully solve the puzzle.

To prevent model overfitting and the occurrence of reward hacking, we introduce a clipping operation to ensure that p_θ does not deviate too far from $p_{\theta_{\text{old}}}$, forcing the policy update ratio $I_t(\theta) = \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{p_{\theta_{\text{old}}}(\mathbf{x}_{t-1}|\mathbf{x}_t)}$ is constrained within a threshold range, which is commonly used in PPO.

Finally, the clipped objective is expressed as follows:

$$\mathcal{J}_{\text{RL}}(\theta) = \tag{12}$$
$$\mathbb{E}_t\left[\sum_{i=1}^{G} \min\left(I_t^i(\theta)\hat{A}_t^i, \text{clip}\left(I_t^i(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_t^i\right)\right]$$

Above, $I_t^i(\theta)$ and $\hat{A}_t^i$ have been defined, and G is decided based on the specific puzzle.

The overall pipeline of the policy optimization algorithm is summarized in Figure 2 and Algorithm 1 in Appendix B.2.

Experiment

Tasks

These constraint satisfaction problems are trivial for computers to solve, but are all challenging for neural networks. In the following part, we provide a brief introduction to each task; more experimental details are shown in Appendix B.1.

Sudoku Sudoku is a classical logic-based puzzle game, in which a player must fill in a 9 × 9 partially-filled board such that each row, each column, and each of nine 3 × 3 subgrids contains exactly one of each number from 1 through 9. Following the setup of Cornelio et al. (2023), we consider three public Sudoku datasets: 1) big kaggle, a subset of a bigger dataset containing 1 million boards hosted on Kaggle; 2) minimal 17, a dataset of minimal Sudoku boards, with only 17 clues (the minimum of hints in a Sudoku board leading to a unique solution); and 3) multiple sol, a dataset of Sudoku boards with two or more solutions. 4) Finally, we considered satnet_data, released by Wang et al. (2019). The information of these datasets is summarized in Table 1. For each Sudoku dataset, we use Prolog or a brute-force algorithm to obtain

model	big kaggle	minimal 17	multiple sol	satnet data
Baseline* (Cornelio et al. (2023))	62.68	0.00	46.70	24.00
RNN+PseudoSL (Ahmed, Chang, and den Broeck (2023))	——	——	——	28.20
DDReasoner-SL	78.19	8.03	96.30	70.60
DDReasoner-RL (Ours)	**97.79**↑	**18.25**↑	**100.00**↑	**92.60**↑

Table 2: Our experimental results on Sudoku. *Baseline is the sub-module **SolverNN** in NASR(Cornelio et al. 2023). It is a Transformer model with a linear layer mapping the input distribution to the probability distribution over the possible complete solutions. Its inputs and labels are consistent with our experiment, so we choose it as the Baseline. The metric used in the table for each dataset is the percentage of completely correct predicted boards, and no partial credit is given for getting individual digits correct. —— in the table means that the code is inaccessible, so we were unable to reproduce its results on other datasets.

model		maze grid size			
name	# parameters	5×5	10×10	15×15	20×20
Baseline* (Nolte et al. (2024))	8M	**100.00**	**100.00**	**100.00**	**100.00**
DDReasoner-SL	5.3M	95.45	70.95	51.35	43.30
DDReasoner-RL (Ours)	5.3M	**100.00**↑	**100.00**↑	**100.00**↑	**100.00**↑

Table 3: Our experimental results on Maze. *Baseline is tailored for Maze navigation based on Transformer and MLM-$\mathcal{U}$ (Kitouni et al. 2024). It uses the same dataset configuration as ours, and can perfectly solve mazes of grid sizes up to 20×20.

solutions (if unlabeled), and split it into a training set with 90% examples and a test set with 10% examples.

Maze Maze is a structured navigation task. The player begins at a starting point and must navigate a connected path to reach a specified endpoint. Movement is generally allowed in four directions: up, down, left, and right, and walls or obstacles cannot be crossed. Our Maze datasets are generated using the configuration of Ivanitskiy et al. (2023). For Maze grid sizes of 5×5, 10×10, 15×15 and 20×20, we separately generate 20'000 puzzles and their corresponding solution pathways, with 18'000 pairs as training set and 2'000 pairs as test set. Each puzzle is generated by Depth-First Search (DFS) and is acyclic, so its solution is unique.

Simple Path Prediction & Preference Learning We adopt the neuro-symbolic benchmarks of Xu et al. (2018). 1) Simple Path Prediction: Given a source and destination node in a 4-by-4 unweighted grid $G = (V, E)$, we need to find the shortest unweighted path connecting them. The known condition is a binary vector of length $|V|+|E|$, where $|V|$ indicates the source and destination nodes and $|E|$ indicates which edges are removed, thus obtaining a subgraph $G' \subseteq G$. The corresponding solution is a binary vector of length $|E|$ indicating which edges are in the shortest path. The dataset consists of 1610 such examples. 2) Preference Learning: Given a user's ranking over a subset of items, we want to predict the user's ranking over remaining items. We use preference ranking data over 10 types of sushi, taking the ordering over 6 types of sushi as the known condition to predict the ordering (a strict total order) over the remaining 4 types. We use preference ranking data over 10 types of sushi for 4926 individuals. For both tasks, we split the data 60%/20%/20% into train/validation/test split.

Minimum-Cost Path Finding Given a weighted $k \times k$ grid, we need to predict the path with minimum cost from the upper left to the lower right vertices. The complete solution can be represented as a $k \times k$ matrix. Notably, the minimum-cost path is possibly not unique, i.e., for a puzzle, two or more minimum-cost paths may exist, and they are all considered correct. Adopting datasets from Pogančić et al. (2020), we consider $k = 12, 18, 24, 30$, and use 10'000 examples as the training set and 1'000 examples as the test set.

Experiment Setup

We allocate a certain percentage of training data for the SL phase. In the RL phase, we continue training from the SL checkpoint using the whole training set. Detailed configurations of both training phases are provided in Appendix B.3.

Main Results

DDReasoner-SL is developed on a basic diffusion model and partial data via supervised learning, a stage where other efforts have been directed (Du, Mao, and Tenenbaum 2024; Zhang et al. 2025; Zeng et al. 2024). We do not focus on elaborate supervised learning nor rely on test-time scaling. Instead, we pay more attention on employing RL techniques to directly internalize logical rules and specific constraints into DDReasoner's capabilities while training, thus allowing for better single-pass performance at inference time. In following results, solutions produced by DDReasoner are obtained via an efficient single-pass deterministic sampling.

Sudoku As demonstrated in Table 2, we outperform the purely neural network Baseline on all four datasets. For the most challenging dataset (minimal_17 with only 17 hints each Sudoku board), the baseline SolverNN cannot produce any completely correct solution board, while DDReasoner

model	exact	hamming	consistent
MLP	5.62	85.91	6.99
I. MLP + Semantic Loss	28.51	83.14	69.89
II. MLP + NESYENT	30.10	83.00	91.60
III. MLP + SPL	37.60	88.50	**100.00**
DDReasoner-SL	66.46	92.71	74.53
DDReasoner-RL (Ours)	**70.81**↑	**93.96**↑	80.12↑

Table 4: Our experimental results on simple path prediction. I, II and III are taken from Xu et al. (2018), Ahmed et al. (2022b) and Ahmed et al. (2022a), respectively. I and II used constraint-based losses, and III added a constraint layer.

model	exact	hamming	consistent
MLP	1.01	75.78	2.72
I. MLP + Semantic Loss	13.59	72.43	55.28
II. MLP + NESYENT	18.20	71.50	96.00
III. MLP + SPL	20.80	72.40	**100.00**
DDReasoner-SL	21.02	78.15	99.80
DDReasoner-RL (Ours)	**22.54**↑	**78.57**↑	**100.00**↑

Table 5: Our experimental results on preference learning.

is capable of yielding a number of consistent predictions, and further strengthens the reasoning ability through RL training. Particularly on the dataset featuring multiple solutions that satisfy the hard constraints (multiple_sol), our RL method based on constraint satisfaction verification empowers DDReasoner to comprehensively grasp this rule, thereby reaching perfect accuracy. These results show that once DDReasoner is equipped with a preliminary capability for generating consistent solutions, we can further refine this particular capability via our policy optimization algorithm.

Maze As demonstrated in Table 3, we achieve 100% accuracy for mazes of grid sizes up to 20×20 via our policy optimization algorithm. Figure 6 in Appendix B.4 shows a performance comparison between our DDReasoner-RL and Baseline on 5×5 mazes. We observe that this perfect performance of our DDReasoner-RL is accompanied by approximately 1.51x more parameters-efficient, 1.39x more data-efficient, and 2.33x more time-efficient than Baseline, respectively. It underscores that training DDReasoner with our RL method can effectively elicit its potential reasoning capabilities while ensuring notable efficiency.

Simple Path Prediction & Preference Learning As demonstrated in Table 4 and Table 5, our DDReasoner-RL surpasses previous loss-based methods on the two tasks in terms of the accuracy of exact match and hamming distance, while also achieving a high level of consistency satisfaction.

Minimum-Cost Path Finding As demonstrated in Table 6, for cases where $k = 12, 18, 24, 30$, the use of RL yielded notable enhancements to both exact and consistent.

model		k=12	k=18	k=24	k=30
DDReasoner-SL	I	88.70	81.90	75.00	70.60
	II	98.70	97.70	96.60	93.20
DDReasoner-RL (Ours)	I	**90.40**	**86.20**	**81.30**	**80.10**
	II	**99.10**	**99.10**	**99.10**	**98.40**

Table 6: Our experimental results on minimum-cost path finding. I and II denote the percentage of finding the shortest path and a connected path, respectively.

task	DDReasoner-SL*		DDReasoner-RL	
	train	test	train	test
Sudoku (big kaggle)	99.30	82.82	99.97	**97.79**
Sudoku (minimal 17)	99.88	9.48	99.04	**18.25**
Sudoku (multiple sol)	95.81	97.20	100.00	**100.00**
Sudoku (satnet data)	99.87	71.70	99.87	**92.60**
Maze (5×5)	99.90	99.85	100.00	**100.00**
Maze (10×10)	95.08	91.35	99.98	**100.00**
Maze (15×15)	59.94	58.10	100.00	**100.00**
Maze (20×20)	62.77	58.35	99.99	**100.00**

Table 7: Partial results of ablation studies. DDReasoner-SL* in this table is obtained by scaling up supervised learning.

Ablation Studies and Overall Analysis

We take the SL checkpoint that is prepared for subsequent RL training, and scale up supervised learning using unsolved puzzles of the entire training set. As shown in Table 7 and additional results in Appendix B.4, utilizing the constraints-based reward signal to guide DDReasoner's policy optimization has refined the SL checkpoint's ability to adhere to logical constraints to a large extent. Furthermore, additional experiments on the generalization ability of our DDReasoner-RL across different Sudoku datasets and varying Maze grid sizes are provided in Appendix B.4.

At inference time, DDReasoner-RL can generate predicted solutions in batch for puzzles in the test set within a few seconds using 4 CPUs and 1 NVIDIA H100 GPU.

Conclusion and Future Work

We use a diffusion model (i.e., DDReasoner) as a symbolic reasoner. We establish DDReasoner's preliminary reasoning skills via supervised learning, and then, use reinforcement learning to finetune it via formulating the denoising process as a Markov decision process. Our reward signal adopts constraint satisfaction verification to prioritize logically consistent outputs. Experimental results demonstrate that our integration of RL ultimately enables the diffusion model to be a proficient reasoner capable of addressing diverse symbolic reasoning tasks. While our current tasks focus on the low-dimensional state space, future work could apply this method to high-dimensional problems, e.g., visual Sudoku and visual generation related to physical consistency. The only change is adding another neural network to discretize the final denoising state and calculate the subsequent reward.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (Grant No. 82394432, and 92249302), and the Shanghai Municipal Science and Technology Major Project (Grant No. 2023SHZDZX02).

References

Agarwal, A.; Shenoy, P.; and Mausam. 2021. End-to-End Neuro-Symbolic Architecture for Image-to-Image Reasoning Tasks. arXiv:2106.03121.

Ahmadian, A.; Cremer, C.; Gallé, M.; Fadaee, M.; Kreutzer, J.; Pietquin, O.; Üstün, A.; and Hooker, S. 2024. Back to Basics: Revisiting REINFORCE-Style Optimization for Learning from Human Feedback in LLMs. In Ku, L.-W.; Martins, A.; and Srikumar, V., eds., *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 12248–12267. Bangkok, Thailand: Association for Computational Linguistics.

Ahmed, K.; Chang, K.-W.; and den Broeck, G. V. 2023. A Pseudo-Semantic Loss for Autoregressive Models with Logical Constraints. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Ahmed, K.; Teso, S.; Chang, K.-W.; Van den Broeck, G.; and Vergari, A. 2022a. Semantic Probabilistic Layers for Neuro-Symbolic Learning. In Koyejo, S.; Mohamed, S.; Agarwal, A.; Belgrave, D.; Cho, K.; and Oh, A., eds., *Advances in Neural Information Processing Systems*, volume 35, 29944–29959. Curran Associates, Inc.

Ahmed, K.; Wang, E.; Chang, K.-W.; and den Broeck, G. V. 2022b. Neuro-Symbolic Entropy Regularization. In *Conference on Uncertainty in Artificial Intelligence*.

Black, K.; Janner, M.; Du, Y.; Kostrikov, I.; and Levine, S. 2024. Training Diffusion Models with Reinforcement Learning. arXiv:2305.13301.

Chi, C.; Feng, S.; Du, Y.; Xu, Z.; Cousineau, E.; Burchfiel, B.; and Song, S. 2023. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*.

Cornelio, C.; Stuehmer, J.; Hu, S. X.; and Hospedales, T. 2023. Learning where and when to reason in neuro-symbolic inference. In *The Eleventh International Conference on Learning Representations*.

Dhariwal, P.; and Nichol, A. Q. 2021. Diffusion Models Beat GANs on Image Synthesis. In Beygelzimer, A.; Dauphin, Y.; Liang, P.; and Vaughan, J. W., eds., *Advances in Neural Information Processing Systems*.

Du, Y.; Mao, J.; and Tenenbaum, J. B. 2024. Learning Iterative Reasoning through Energy Diffusion. In *International Conference on Machine Learning (ICML)*.

Fan, Y.; and Lee, K. 2024. Optimizing DDPM Sampling with Shortcut Fine-Tuning. arXiv:2301.13362.

Fan, Y.; Watkins, O.; Du, Y.; Liu, H.; Ryu, M.; Boutilier, C.; Abbeel, P.; Ghavamzadeh, M.; Lee, K.; and Lee, K. 2023. Reinforcement Learning for Fine-tuning Text-to-Image Diffusion Models. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Fu, J.; and Topcu, U. 2014. Probably Approximately Correct MDP Learning and Control With Temporal Logic Constraints. arXiv:1404.7073.

Giunchiglia, E.; and Lukasiewicz, T. 2021. Multi-Label Classification Neural Networks with Hard Logical Constraints. *Journal of Artificial Iintelligence Research (JAIR)*, 72.

Gupta, S.; Ahuja, C.; Lin, T.-Y.; Roy, S. D.; Oosterhuis, H.; de Rijke, M.; and Shukla, S. N. 2025. A Simple and Effective Reinforcement Learning Method for Text-to-Image Diffusion Fine-tuning. arXiv:2503.00897.

Hasanbeig, M.; Abate, A.; and Kroening, D. 2019. Logically-Constrained Reinforcement Learning. arXiv:1801.08099.

He, Y.; Yang, T.; Zhang, Y.; Shan, Y.; and Chen, Q. 2023. Latent Video Diffusion Models for High-Fidelity Long Video Generation. arXiv:2211.13221.

Ho, J.; Jain, A.; and Abbeel, P. 2020. Denoising Diffusion Probabilistic Models. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *Advances in Neural Information Processing Systems*, volume 33, 6840–6851. Curran Associates, Inc.

Ho, J.; Salimans, T.; Gritsenko, A.; Chan, W.; Norouzi, M.; and Fleet, D. J. 2022. Video Diffusion Models. arXiv:2204.03458.

Ivanitskiy, M. I.; Shah, R.; Spies, A. F.; Räuker, T.; Valentine, D.; Rager, C.; Quirke, L.; Mathwin, C.; Corlouer, G.; Behn, C. D.; and Fung, S. W. 2023. A Configurable Library for Generating and Manipulating Maze Datasets. arXiv:2309.10498.

Janner, M.; Du, Y.; Tenenbaum, J. B.; and Levine, S. 2022. Planning with Diffusion for Flexible Behavior Synthesis. In *International Conference on Machine Learning*.

Kakade, S. M.; and Langford, J. 2002. Approximately Optimal Approximate Reinforcement Learning. In *International Conference on Machine Learning*.

Kitouni, O.; Nolte, N.; Williams, A.; Rabbat, M.; Bouchacourt, D.; and Ibrahim, M. 2024. The Factorization Curse: Which Tokens You Predict Underlie the Reversal Curse and More. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Li, T.; Gupta, V.; Mehta, M.; and Srikumar, V. 2019. A Logic-Driven Framework for Consistency of Neural Models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*.

Li, Z.; Huang, Y.; Li, Z.; Yao, Y.; Xu, J.; Chen, T.; Ma, X.; and Lu, J. 2023. Neuro-symbolic Learning Yielding Logical Constraints. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Nichol, A.; and Dhariwal, P. 2021. Improved Denoising Diffusion Probabilistic Models. *Cornell University - arXiv,Cornell University - arXiv*.

Nolte, N.; Kitouni, O.; Williams, A.; Rabbat, M.; and Ibrahim, M. 2024. Transformers Can Navigate Mazes With Multi-Step Prediction. arXiv:2412.05117.

Petersen, F.; Borgelt, C.; Kuehne, H.; and Deussen, O. 2021. Learning with Algorithmic Supervision via Continuous Relaxations. In Beygelzimer, A.; Dauphin, Y.; Liang, P.; and Vaughan, J. W., eds., *Advances in Neural Information Processing Systems*.

Pogančić, M. V.; Paulus, A.; Musil, V.; Martius, G.; and Rolinek, M. 2020. Differentiation of Blackbox Combinatorial Solvers. In *International Conference on Learning Representations*.

Rafailov, R.; Sharma, A.; Mitchell, E.; Manning, C. D.; Ermon, S.; and Finn, C. 2023. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; and Ommer, B. 2022. High-Resolution Image Synthesis with Latent Diffusion Models. arXiv:2112.10752.

Saharia, C.; Chan, W.; Saxena, S.; Li, L.; Whang, J.; Denton, E.; Ghasemipour, S. K. S.; Ayan, B. K.; Mahdavi, S. S.; Lopes, R. G.; Salimans, T.; Ho, J.; Fleet, D. J.; and Norouzi, M. 2022. Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. arXiv:2205.11487.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal Policy Optimization Algorithms. arXiv:1707.06347.

Shao, Z.; Wang, P.; Zhu, Q.; Xu, R.; Song, J.; Zhang, M.; Li, Y.; Wu, Y.; and Guo, D. 2024. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models.

Sohl-Dickstein, J.; Weiss, E.; Maheswaranathan, N.; and Ganguli, S. 2015. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. *arXiv: Learning,arXiv: Learning*.

Suresh, T.; Banerjee, D.; Ugare, S.; Misailovic, S.; and Singh, G. 2025. DINGO: Constrained Inference for Diffusion LLMs. In *ICML 2025 Workshop on Reliable and Responsible Foundation Models*.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement Learning: An Introduction*. The MIT Press, second edition.

van Krieken, E.; Minervini, P.; Ponti, E.; and Vergari, A. 2025. Neurosymbolic Diffusion Models. arXiv:2505.13138.

Wallace, B.; Dang, M.; Rafailov, R.; Zhou, L.; Lou, A.; Purushwalkam, S.; Ermon, S.; Xiong, C.; Joty, S.; and Naik, N. 2023. Diffusion Model Alignment Using Direct Preference Optimization. arXiv:2311.12908.

Wang, P.; Donti, P. L.; Wilder, B.; and Kolter, J. Z. 2019. SATNet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver. *CoRR*, abs/1905.12149.

Wang, W.; and Pan, S. J. 2019. Integrating Deep Learning with Logic Fusion for Information Extraction. *ArXiv*, abs/1912.03041.

Xie, Y.; Xu, Z.; Kankanhalli, M. S.; Meel, K. S.; and Soh, H. 2019. Embedding Symbolic Knowledge into Deep Networks. In Wallach, H.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Xu, J.; Zhang, Z.; Friedman, T.; Liang, Y.; and Van den Broeck, G. 2018. A Semantic Loss Function for Deep Learning with Symbolic Knowledge. In Dy, J.; and Krause, A., eds., *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, 5502–5511. PMLR.

Yang, Z.; Ishay, A.; and Lee, J. 2020. NeurASP: Embracing Neural Networks into Answer Set Programming. In Bessiere, C., ed., *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, 1755–1762. International Joint Conferences on Artificial Intelligence Organization. Main track.

Ye, J.; Gao, J.; Gong, S.; Zheng, L.; Jiang, X.; Li, Z.; and Kong, L. 2025. Beyond Autoregression: Discrete Diffusion for Complex Reasoning and Planning. In *The Thirteenth International Conference on Learning Representations*.

Ye, J.; Gong, S.; Chen, L.; Zheng, L.; Gao, J.; Shi, H.; Wu, C.; Li, Z.; Bi, W.; and Kong, L. 2024. Diffusion of Thoughts: Chain-of-Thought Reasoning in Diffusion Language Models. *arXiv preprint arXiv:2402.07754*.

Zeng, H.; Wang, J.; Das, A.; He, J.; Han, K.; Hu, H.; and Sun, M. 2024. Effective Generation of Feasible Solutions for Integer Programming via Guided Diffusion. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, 4107–4118. New York, NY, USA: Association for Computing Machinery. ISBN 9798400704901.

Zhang, T.; Pan, J.-S.; Feng, R.; and Wu, T. 2025. T-SCEND: Test-time Scalable MCTS-enhanced Diffusion Model. arXiv:2502.01989.

Zhao, S.; Gupta, D.; Zheng, Q.; and Grover, A. 2025. d1: Scaling Reasoning in Diffusion Large Language Models via Reinforcement Learning. arXiv:2504.12216.