Diff-DAgger: Uncertainty Estimation with Diffusion Policy for Robotic Manipulation

Anonymous authors

Abstract: Diffusion policy has shown impressive results in handling multi-modal tasks in robotic manipulation. However, it has fundamental limitations in out-of-distribution failures that persist due to compounding errors and its limited capability to extrapolate. One way to address these limitations is robot-gated DAgger, an interactive imitation learning with a robot query system to actively seek expert help during policy rollout. While robot-gated DAgger has high potential for learning at scale, existing methods like Ensemble-DAgger struggle with highly expressive policies: They often misinterpret policy disagreements as uncertainty at multi-modal decision points. To address this, we introduce Diff-DAgger, an efficient robot-gated DAgger algorithm that leverages the training objective of diffusion policy. We evaluate Diff-DAgger across different robot tasks and show that Diff-DAgger overall achieves 81.5% accuracy on the task failure prediction and improves task completion rate by 13.8%. We hope that this work opens up a path for efficiently incorporating expressive policies into interactive robot learning.

Keywords: Robot-gated DAgger, Diffusion Policy

1 Introduction

Recently, significant advances have been made in offline imitation learning by adopting generative models for action sequence prediction. These models mitigate the compounding error issue by reducing the number of action inferences [1] and addressing multi-modality through generative capabilities [2]. In particular, diffusion policy [3] trains the model to predict the gradient of the action score function, and it has achieved state-of-the-art results for learning different robotic tasks.

While diffusion policy effectively handles highly multi-modal demonstrations, it still faces a core issue of behavior cloning: the out-of-distribution (OOD) failure, where the policy compounds errors in action prediction and cannot extrapolate in unseen states. A common strategy to address OOD failure is DAgger [4], an interactive learning method which iteratively refines the policy by incorporating expert interventions. In particular, robot-gated DAgger lets robots decide when to seek expert help, removing the need for constant human supervision and enabling large-scale learning across multiple robots [5]. The effectiveness of this approach, therefore, relies heavily on the robot's ability to accurately estimate its own uncertainty.

However, none of the existing robot-gated DAgger approaches fully addresses the multi-modality present in demonstration data. For example, the most widely adopted Ensemble-DAgger approaches use the action variance among an ensemble of policies to estimate uncertainty [6, 7, 8]. When we categorize demonstration data into in-distribution uni-modal data (single solution), in-distribution multi-modal data (multiple solutions), and out-of-distribution data, existing Ensemble-DAgger methods effectively distinguish between in-distribution uni-modal and out-of-distribution cases using action variance. However, they struggle with in-distribution multi-modal data, where multiple valid solutions cause policy disagreement and can mistakenly indicate high uncertainty.

Incorporating an expressive policy class that handles multi-modal data into the DAgger framework is a promising but under-explored direction. In light of this, we propose Diff-DAgger, an efficient robot-gated DAgger algorithm that addresses multi-modality in demonstration data. Diff-DAgger

8th Conference on Robot Learning (CoRL 2024), Munich, Germany.

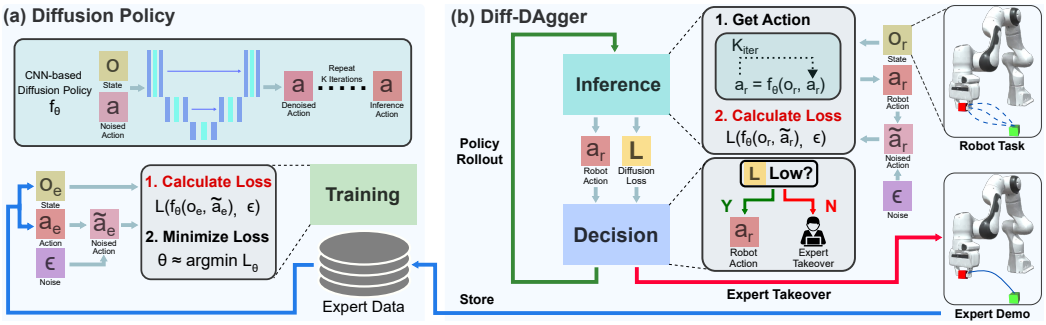

Figure 1: **Overview of Diff-DAgger. (a) Diffusion Policy and Training**: In our study, we use the diffusion policy with a U-net architecture, a CNN-based model. During the training phase, diffusion policy is trained on static expert data. **(b) Diff-DAgger during Deployment**: At each timestep during the online learning phase, the robot performs two steps—**inference** and **decision**. In the inference step, the policy makes action inference by iteratively denoising a random noise, conditioned on the current observation. Then the training loss function is used to compute the loss associated with the generated action. In the decision step, the robot proceeds with the action if the loss is low, otherwise it asks the expert to intervene. The arrows indicate the flow of steps.

uses a query system based on the loss function of the diffusion model, referred to as diffusion loss, which significantly improves the effectiveness of robot queries across various evaluation metrics.

We make the following contributions: First, we develop a novel robot-gated query system based on diffusion loss to handle multi-modality in expert demonstration. Second, we quantitatively evaluate Diff-DAgger's efficiency in terms of task failure prediction and task completion during interactive learning to demonstrate its superiority over baseline methods.

2 Method

2.1 Problem Statement

The goal of imitation learning is to collect a dataset and train a robot policy to complete a set of tasks with expert demonstrations. In the robot-gated DAgger scenario, the robot autonomously queries for expert intervention. We assume the expert is skilled in completing the task, which is considered successful if the goal is reached within the time limit. We assume access to success information. Our objective is to design an efficient robot query system that accurately predicts task failures, requesting expert intervention only when necessary.

2.2 Diffusion Policy

In diffusion policy, robot behavior is generated through an iterative denoising process based on Langevin dynamics, formally modeled as a Denoising Diffusion Probabilistic Model (DDPM) [2]. This denoising process begins with a Gaussian prior, $\mathcal{N}(0, I)$, and iteratively removes noise to reconstruct the original data. Diffusion policy conditions on observation sequences and generates an action sequence, achieving state-of-the-art performance across various robotic tasks [3].

During training, diffusion policy samples a denoising timestep and a Gaussian noise vector to corrupt the action sequence. It then predicts the target objective given the corrupted action sequence and the observation sequence, and performs optimization by minimizing the following loss function:

$$L_\pi(o, a, \epsilon, t) = \|f(\epsilon, a) - f_\theta(o, \sqrt{\alpha_t}a + \sqrt{1-\alpha_t}\epsilon, t)\|^2 \quad (1)$$

where o is the observation sequence, a is the action sequence, and ϵ is a noise vector from Gaussian distribution $\mathcal{N}(0, I)$. The parameter α_t is a time-dependent factor that modulates the influence of the original data and the added noise during the diffusion process, and t is the denoising timestep.

The loss function $L(o, a, \epsilon, t)$ calculates the mean squared error (MSE) between the target variable $f(\epsilon, a)$, which is a function of noise and ground truth action depending on the choice of target prediction, and the noise predicted by the model f_θ, given the noised data at time t [9].

2.3 Diff-DAgger

During robot rollout, the uncertainty of the current action plan is estimated using the diffusion policy's loss value for the given observation and the generated action. To establish a baseline, we calculate the expected diffusion loss $\mathscr{L}$ for each data point in the training dataset using equation 2.

$$\mathscr{L}(o, a, \pi) = E_{\epsilon \sim \mathcal{N}(0,I), t \sim \mu(1,T)} L_\pi(o, a, \epsilon, t) \tag{2}$$

For practical implementation, we select a batch size N_b to sample both the noise and diffusion timesteps and compute the average loss over the batch. Given a quantile threshold α, we assume OOD if the diffusion loss of the generated action using Eq. 2 exceeds the α-quantile of the expected diffusion losses from the training data. Additionally, expert supervision takes place only if K past states consecutively violate the qauntile threshold α. It helps distinguish brief fluctuations from persistent out-of-distribution cases, significantly reducing the rate of false positives.

We now summarize the data aggregation and policy update procedures. Diff-DAgger first initializes the policy π_r by training it on $\mathrm{N_i}$ initial expert demonstrations. After training the policy, the robot continues rolling out the action in the environment until high diffusion loss is detected. When the robot query is made, experts intervene from the current scene and complete the task.

3 Experimental Results

We address the following three research questions to understand the effectiveness of Diff-DAgger.

RQ1 Can diffusion loss predict task failure (Section 3.3)?
RQ2 Given the same number of demonstrations, can Diff-DAgger outperform other methods in task completion (Section 3.4)?

3.1 Baselines

We compare Diff-DAgger to the following algorithms: Behavior Cloning, which uses the offline human demonstration data as the training data; Ensemble-DAgger, which uses five policies to calculate action variance for uncertainty estimation; and ThriftyDAgger [7], a SOTA robot-gated DAgger algorithm which incorporates risk measure into the decision rule using an offline RL algorithm. All methods use the diffusion policy with the same model architecture for consistency.

3.2 Tasks and Experts Setup

We include three tasks in simulation—stacking, pushing, and plugging—using the ManiSkill environment [10]. We design experts as either multi-modal or long-horizon planners to evaluate different expert types. Specifically, we use RL agents and motion planners as the experts: the motion planner, is a long-horizon planner that generates actions based on future goals, whereas multiple RL agents with different reward functions provide different modes for task completion. For stacking and plugging tasks, we use a rule-based RRT motion planner to simulate the expert [11]. For pushing, two experts are trained using PPO algorithm [12] with varying rewards: in addition to task-related reward, one agent is additionally rewarded for rotating the object clockwise direction, whereas other agent is rewarded for rotating counter-clockwise (2 Experts). We additionally compare learning from the counter-clockwise agent only (1 Expert).

3.3 Predictability of Robot Query on Task Failure

A robot query system's effectiveness depends on accurately predicting the need for expert intervention, ensuring queries are made only when necessary. To quantitatively evaluate this, we use a

confusion matrix between robot query decision and task outcome. For each task, we train the robot policy on a fixed number of expert demonstrations and rollout from 100 different initial configurations. We record task completion without expert intervention and whether the policy estimates high uncertainty during each episode, as described in Section 2.3. The results show that Diff-DAgger achieves a significant improvement over the baseline methods in terms of its ability to predict task failure: Compared to Ensemble-DAgger, Diff-DAgger achieves 47% higher F1 and 45% greater accuracy scores on average across the visuomotor tasks. Compared to Thrifty-DAgger, Diff-DAgger improves F1 scores by 37% and accuracy by 43%. The results are summarized in Table 1.

Task	N_D	Method	TP	TN	FP	FN	F1 ↑	Acc ↑	Task	N_D	Method	TP	TN	FP	FN	F1 ↑	Acc ↑
Stacking	20	Ours	29	59	10	2	**0.84**	**0.88**	Pushing (1 Expert)	50	Ours	29	51	9	11	**0.74**	**0.80**
		Ensemble-DAgger	27	33	37	6	0.55	0.60			Ensemble-DAgger	38	10	52	0	0.59	0.48
		Thrifty-DAgger	33	30	34	3	0.49	0.63			Thrifty-DAgger	37	6	57	0	0.57	0.43
Pushing (2 Experts)	25/25	Ours	27	51	8	14	**0.71**	**0.78**	Plugging	100	Ours	31	49	13	7	**0.76**	**0.80**
		Ensemble-DAgger	15	45	5	35	0.43	0.60			Ensemble-DAgger	22	37	18	23	0.52	0.59
		Thrifty-DAgger	25	38	15	22	0.57	0.63			Thrifty-DAgger	30	34	25	11	0.62	0.64

Table 1: Confusion matrix comparing the binary robot query decisions (query or no query) with the binary task outcomes (success or failure) in 100 policy rollouts for visuomotor tasks. Here, *True Positive (TP)* is a query made to a failing episode. *True Negative (TN)* is a successful episode without query. *False Positive (FP)* is an unnecessary query made to a successful episode, and *False Negative (FN)* is a failure episode without query. N_D is the number of expert demonstrations.

3.4 Task Performance

We evaluate the policy performance for both state-based and image-based observations. Diff-DAgger method consistently outperforms the baselines in task completion rate across tasks given the same number of expert demonstrations. On average, Diff-DAgger achieves a 12.6% improvement over the second-best method in state-based tasks. For visuomotor tasks, Diff-DAgger on average achieves a 15.0% improvement over the other methods. The results are summerized in Table 2.

	Stacking		Pushing (1 Expert)		Pushing (2 Experts)		Plugging	
	State	Image	State	Image	State	Image	State	Image
Behavior Cloning	0.83	0.85	0.92	0.87	0.91	0.91	0.73	0.68
Ensemble-DAgger	0.90	0.87	0.90	0.80	0.66	0.58	0.57	0.53
Thrifty-DAgger	0.88	0.87	0.85	0.78	0.77	0.73	0.69	0.60
Ours	**0.98**	**1.00**	**0.97**	**0.99**	**0.98**	**0.97**	**0.87**	**0.83**

Table 2: Performance comparison across different tasks with state-based and image-based observations. We report the success rates using 100 environment configurations after the final iteration of DAgger. For each task, the number of expert demonstrations is fixed (detailed in Table 4) across methods. Our results show that Diff-DAgger consistently outperforms the other DAgger baseline methods given the same number of demonstrations from the expert.

4 Conclusion

In this work, we present Diff-DAgger, an efficient robot-gated DAgger method with diffusion policy. Our work requires no additional training or data collection steps for failure prediction, making it an efficient algorithm in comparison to other methods like Ensemble DAgger. Our future work includes learning from real human expert for challenging real-world tasks. We hope that this work paves a way to open up more discussion for incorporating expressive policies into interactive learning frameworks in anticipation that computing will become faster over time.

References

[1] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

[2] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[3] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems*, July 2023.

[4] S. Ross, G. Gordon, and D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

[5] R. Hoque, L. Y. Chen, S. Sharma, K. Dharmarajan, B. Thananjeyan, P. Abbeel, and K. Goldberg. Fleet-dagger: Interactive robot fleet learning with scalable human supervision. In *Conference on Robot Learning*, pages 368–380, 2023.

[6] K. Menda, K. Driggs-Campbell, and M. J. Kochenderfer. Ensembledagger: A bayesian approach to safe imitation learning. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5041–5048. IEEE, 2019.

[7] R. Hoque, A. Balakrishna, E. Novoseller, A. Wilcox, D. S. Brown, and K. Goldberg. ThriftyDAgger: Budget-aware novelty and risk gating for interactive imitation learning. In *Conference on Robot Learning*, 2021.

[8] S. Dass, K. Pertsch, H. Zhang, Y. Lee, J. J. Lim, and S. Nikolaidis. Pato: Policy assisted teleoperation for scalable robot data collection. In *Proceedings of Robotics: Science and Systems*, 2023.

[9] T. Salimans and J. Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022.

[10] T. Mu, Z. Ling, F. Xiang, D. Yang, X. Li, S. Tao, Z. Huang, Z. Jia, and H. Su. Maniskill: Generalizable manipulation skill benchmark with large-scale demonstrations. *arXiv preprint arXiv:2107.14483*, 2021.

[11] J. J. Kuffner and S. M. LaValle. Rrt-connect: An efficient approach to single-query path planning. In *IEEE International Conference on Robotics and Automation*, pages 995–1001. IEEE, 2000.

[12] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

A Implementation Details

Here we describe the details of the implementation not included in the main text.

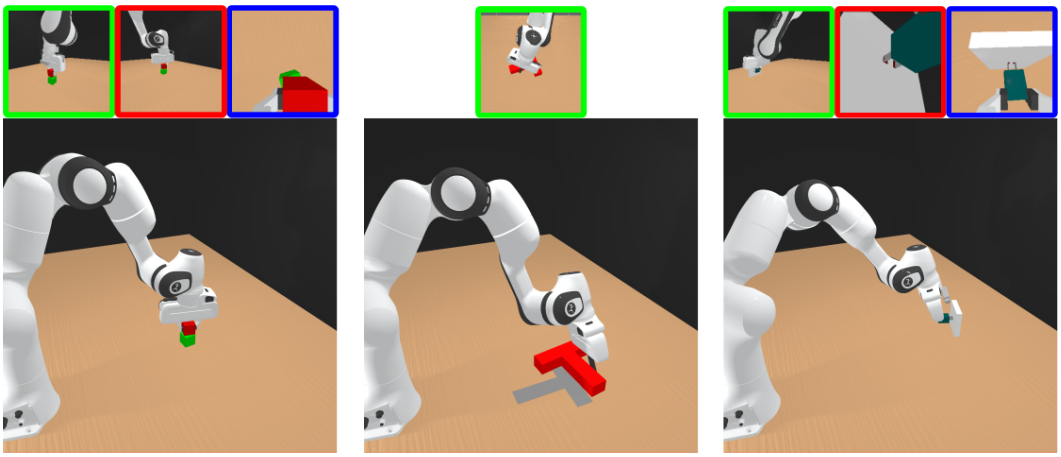

Figure 2: Tasks (**Bottom**) and Camera Views (**Top**). For pushing, a single camera is used. For stacking and plugging, three cameras are used to account for occlusion and gripper motion.

Task	# Obj	Contact Rich	High-Fidel	T_L	Expert Type	Task Dur	Succ Rate
Stacking	2	False	False	250	Motion Planner	60.3	1.00
Pushing	1	True	False	250	RL Agent	65.7	0.93
Plugging	2	False	True	500	Motion Planner	147	0.98

Table 3: **Task and Expert Summary.** # Obj: number of objects, Contact-rich: contact-richness between robot and objects, High-Fidel: High-fidelity task, T_L: time limit in simulation steps at 20 Hz, Expert Type: type of expert, Task Dur: average expert task duration in simulation steps, Exp SR: expert success rate during intervention.

Task	Action	T_o	T_p	T_a	N_c	RN18	PredTg	T_d	N_i	N_f	N_d	α	K	N_b
Stacking	EEF	1	24	8	3	PT	V-obj	8	20	52	4	0.99	2	256
Pushing	Joint	1	24	2	1	FT	Epsilon	8	40	92	4	0.98	1	256
Plugging	EEF	1	48	16	3	FT	V-obj	64	20	100	8	0.99	3	256

Table 4: Hyperparameters for Diffusion Policy and DAgger. Action: action space, T_o: observation horizon, T_p: prediction horizon, T_a: rollout horizon, N_c: number of cameras, RN18: training strategy for vision encoder where PT means pre-trained and FT means fine-tuned, PredTg: prediction target when training diffusion policy, T_d: number of diffusion timesteps for training and inference, N_i: number of initial expert demonstrations, N_f: final number of expert demonstrations, N_d: number of expert intervention after each training session, K: patience for consecutive violation of threshold, N_b: Diff-DAgger specific batch size for calculating the expected diffusion loss.