CARF: Contrastive Attraction–Repulsion of Failure-Guided Flow Matching

Shuqi Zhao, Bang Du, Cheng-en Wu, Yichen Xie, Yixiao Wang, Masayoshi Tomizuka

***Abstract*— Robot demonstration collection often produces imperfect or failed trajectories in addition to successful demonstrations. Existing methods typically exploit failed trajectories by identifying segments that still make progress toward task completion, but largely overlook *failure-critical behaviors* that directly lead to task failure. Here we argue that these two types of segments provide fundamentally asymmetric supervision: progressive segments should be imitated, whereas failure-critical segments should be explicitly avoided. Based on this observation, we propose CARF, a Contrastive Attraction-Repulsion of Failure-guided framework for learning from imperfect robot data. CARF introduces a progress-based importance scorer, trained solely on successful expert demonstrations and its perturbation results, to estimate step-wise contributions toward task completion and identify informative regions in failed trajectories. These scores guide a unified flow-matching objective that attracts the policy toward progressive behaviors and repels it from failure-critical ones, while excluding ambiguous segments. This enables more comprehensive utilization of imperfect data and avoids unreliable supervision from ambiguous failure segments. Extensive experiments in simulation and the real world demonstrate consistent improvements over competing baselines across diverse failure scenarios, with ablations further validating the effectiveness of the proposed scoring and attraction-repulsion mechanisms. Our website is .**

I. INTRODUCTION

Different from images or texts easily available from the Internet, the collection of robot data has always been a serious challenge due to its extremely time-consuming procedure and intensive human labor [1]–[6]. Consequently, it is of great necessity for modern robot learning to make full use of all available robot data [7]–[10], especially given that the human demonstration collection process inevitably involves not only successful but also imperfect or failed task executions [11]. Recent efforts have explored different ways to leverage imperfect robot data. Offline reinforcement learning is one possible direction, where policies are improved from fixed datasets through value estimation under sparse success or failure rewards [12]–[14]. However, such methods often depend on indirect long-horizon credit assignment through bootstrapped value propagation, which can make training unstable in practice. Another line of imitation learning methods selects useful suboptimal data based on feature similarity and reuses selected segments as additional demonstrations [15]–[17].

However, existing methods primarily focus on identifying *Progressive Segments* from imperfect data, i.e., segments that still make meaningful progress toward task completion, while largely overlooking the information contained in the remaining portions. In fact, these remaining segments can be further divided into two categories: *Indeterminate Segments* and *Failure-Critical Segments*. While *Indeterminate Segments* provide ambiguous supervision due to their mixed execution quality, *Failure-Critical Segments* can still be effectively exploited, as they correspond to decisive erroneous behaviors that directly prevent task success. Fundamentally, *Progressive Segments* and *Failure-Critical Segments* provide asymmetric learning signals: a policy should imitate the former while avoiding the latter, while focusing solely on *Progressive Segments* may leave valuable information in the remaining robot data underutilized.

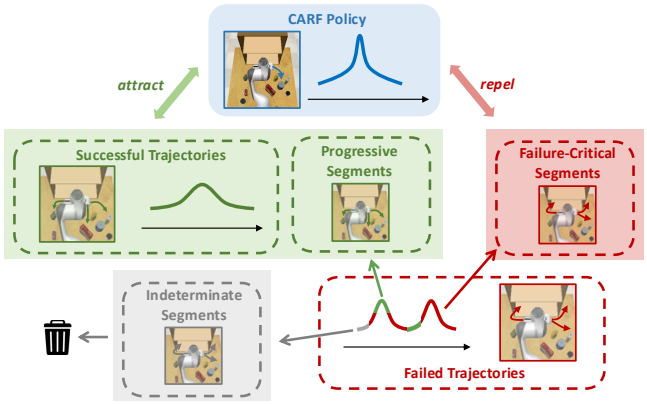

Fig. 1: CARF leverages a crucial yet largely underexplored source of data: *Failure-Critical Segments*. It further formulates an attraction-repulsion flow-matching policy that pulls the policy toward the correct action distribution while pushing it away from failure distributions.

To this end, we propose CARF, a framework that explicitly models these asymmetric learning signals through attraction and repulsion. To achieve this, we introduce a progress-based importance scorer learned solely from successful expert trajectories to quantify the contribution of each step's state-action pairs to the final task completion, providing a step-wise usefulness estimate in failed task executions. Both successful and failure data are then integrated into a unified training process, where the frozen trained scorer decides the asymmetric attraction–repulsion behavior of the flow dynamics for each step-wise action. This explicitly steers the flow matching policy towards success and away from failure. Compared to previous methods, our approach enables more comprehensive utilization of available robot data by extracting complementary learning signals from both progressive and failure-critical behaviors. Overall, our contributions are listed as follows:

- We propose **CARF**, a novel **C**ontrastive **A**ttraction-

Repulsion of **F**ailure-guided framework that enables policy to systematically extract informative signals from both *Progressive Segments* and *Failure-Critical Segments* action clips, enabling more comprehensive data utilization.
- We introduce a progress-based importance scoring mechanism that explicitly quantifies step-wise contribution to task success and integrate it into an attraction–repulsion strategy, leading to more stable and targeted policy learning.
- We conduct extensive experiments with thorough ablations and visualizations to validate our methods in both simulation and real world. Our method outperforms all proposed baselines in multiple types of failure scenarios.

II. RELATED WORK

Imitation Learning Imitation learning provides a simple and effective paradigm for visuomotor policy learning by directly fitting policies to expert demonstrations [18]–[22]. Early behavior cloning methods commonly predict single-step actions, while recent sequence-modeling approaches such as ACT [23] use action chunking and transformer to model temporally coherent action sequences. Generative policies further improves the expressiveness of imitation learning: Diffusion Policy [24] represents action trajectories through an iterative denoising process, enabling multimodal action prediction under image observations, while flow-matching-based policies learn continuous vector fields that transport noise toward expert action distributions with more direct training objectives [25], [26]. Despite these advances, most imitation learning methods still rely primarily on successful demonstrations and treat failed rollouts as unusable or harmful. Instead, our method extends flow-based imitation learning to imperfect datasets by explicitly modeling both attractive and repulsive signals, allowing visuomotor policies to learn from broader robot experience.

Learning from failure A natural way to improve data efficiency in imitation learning is to reuse imperfect or failed trajectories rather than discarding them entirely [12], [27]–[32]. Existing methods commonly estimate the usefulness of such data through classifier guidance [33], trajectory-quality prediction, or similarity-based scoring between failed segments and successful demonstrations [15]–[17]. Related to this goal, offline reinforcement learning also learns from fixed datasets without additional environment interaction [34]–[36], but many representative methods are primarily evaluated in state-based settings, while recent visuomotor extensions often involve human-in-the-loop correction during training [13]. Similarity-based reuse can be misleading: near-success segments may have high similarity to successful demonstrations, yet still encode critical failure modes that the final policy should avoid. Our method addresses this limitation with an attraction–repulsion flow-matching objective, which attracts the policy toward successful behaviors while explicitly repels it from near-success-but-still-failed behaviors, enabling a simple imitation-learning pipeline to benefit from imperfect visuomotor data without online interaction or human intervention during training.

III. METHODOLOGY

The overall framework of CARF is illustrated in Fig. 2. To realize the above objective, we first construct progress supervision from successful trajectories in Section III-A. We then train a progress prediction scorer to estimate the contribution of action segments in Section III-B. Finally, the resulting importance scores obtained from unseen failure trajectories guide the attraction-repulsion flow-matching objective in Section III-C.

A. Progress Score Generation for Success Trajectories

To estimate the importance of each action step to final success, we assign a *progress score* to each step for most tasks. We then derive the *importance score* of a transition from the temporal difference between two consecutive progress scores, reflecting how much the action advances the task toward success based on current observation. Considering a complex long-horizon task execution, the *importance score* increases over time within each subtask, as later stages are closer to the subtask outcome, allow fewer feasible corrective actions, and tolerate smaller execution errors.

Building upon this, progress scores for each demonstration trajectory are automatically labeled. Specifically, we first divide long-horizon tasks into multiple subtasks based on *change in the gripper state*, which typically indicates a meaningful transition in manipulation progress. Let a successful trajectory be segmented into K subtasks. We assign the k-th subtask a normalized progress interval

$$\mathcal{I} * k = [b * k - 1, b_k], \qquad b_k = \frac{k}{K}, \quad k = 1, \ldots, K. \tag{1}$$

Within each subtask, all steps preceding the last t_0 steps are assigned the lower bound b_{k-1} of the corresponding progress interval. For the last t_0 steps, let $r_t \in [0, 1]$ denote the normalized temporal position within this transition region. Their progress labels are defined as

$$p_t = b_{k-1} + (b_k - b_{k-1})\phi_{\exp}(r_t), \tag{2}$$

where

$$\phi_{\exp}(r_t) = \frac{\exp(\beta r_t) - 1}{\exp(\beta) - 1}, \qquad \beta > 0, \tag{3}$$

and $\phi_{\exp} : [0, 1] \to [0, 1]$ is a monotonically increasing exponential mapping satisfying $\phi_{\exp}(0) = 0$ and $\phi_{\exp}(1) = 1$. Thus, the progress label remains constant at b_{k-1} for the early portion of each subtask and increases smoothly from b_{k-1} to b_k during its last t_0 steps. This formulation guarantees monotonically increasing progress within each subtask, while the interval width $1/K$ maintains a consistent global progress range across tasks. As a result, it guarantees importance scores to be also monotonous over time due to exponential labeling. The middle plot ('Labeled scores') in Figure 2 (a) illustrates progress scores for a complete trajectory at each action step, where the red dashed lines indicate the boundaries between subtasks.

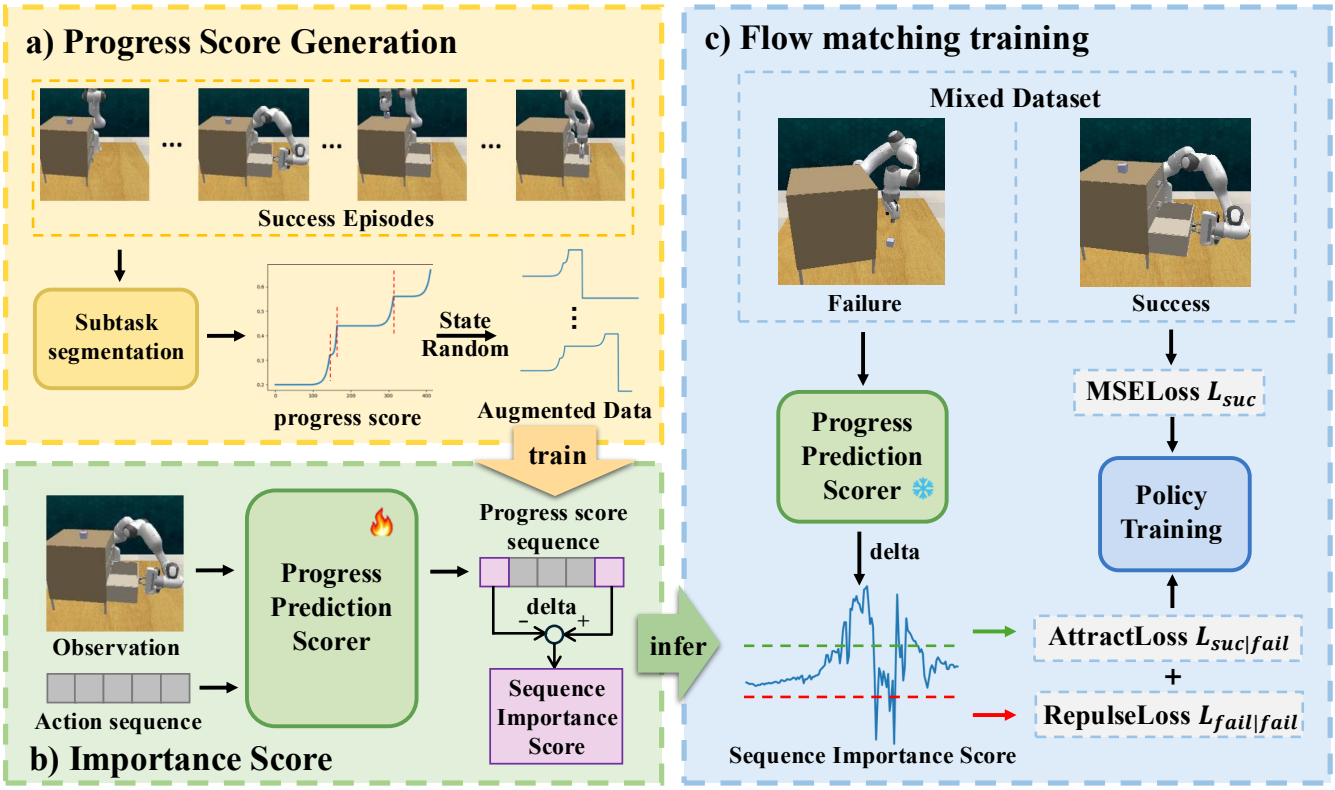

Fig. 2: Overview framework of CARF. After automatically labeling success dataset based on subtask segmentation, CARF trains a progress prediction scorer that provides importance score for observation-action chunking sequence, which is used for attraction-repulsion flow matching training.

To expand the data distribution and prevent overfitting, we generate augmented trajectories by randomly selecting several action steps and perturbing gripper states at those time steps in successful demonstrations (referred to as 'random state' in Figure 2 (a)). Such perturbations typically cause the trajectory to fail at that point, and the progress score drops back to zero afterwards. Since the perturbation time steps are sampled randomly, a single successful trajectory can produce multiple failure variants, each with a different score profile. As a result, the training data of our scorer is mixed with action executions of real successes and synthetic failures, improving scorer's generalizability.

B. Importance Score Generation

After preparing the dataset, as illustrated in Figure 2(b), we train a progress prediction scorer to estimate the task progress associated with each action step in failed execution segments conditioned on the current observation. Specifically, given an observation and the corresponding H-step action at each timestep, the scorer predicts H step-wise progress scores (one for each action), and is supervised using the progress annotations introduced in Section III-A.

Since flow matching policy operates over a multi-step action chunk, importance score of this action chunk is computed by the numerical difference of progress scores between the last and first timestep of this chunk. Specifically, we define the importance score of a chunk spanning timesteps t to $t+H-1$ as

$$g(x_{t:t+H-1}) = \hat{p}_{t+H-1} - \hat{p}_t. \tag{4}$$

Equivalently,

$$g(x_{t:t+H-1}) = \sum_{i=t}^{t+H-2} (\hat{p}_{i+1} - \hat{p}_i), \tag{5}$$

showing that the chunk-level score measures the accumulated progress over the corresponding action sequence rather than its absolute progress value. This design is grounded on the monotonic increase of progress scores in the data labeling process (Section III-A) in each successful task executions, so that the failure-critical action chunks can be clearly indicated with non-positive delta values.

The progress prediction scorer is first trained independently on the labeled successful trajectories and their augmented variants. After training, its parameters are frozen and the scorer is only used to provide importance scores for policy training. The scorer is not required during policy inference.

C. Attraction-Repulsion Contrastive Flow Matching Training

We train flow matching policy on mixture of successful and real failed demonstrations instead of the augmented ones.

During training, each failure trajectory is evaluated by the frozen scorer (Section III-B) that produces an importance score for each action chunk, as exemplified in the lower-left plot of Figure 2(c). High values indicate useful action segments in failed task executions that the policy should learn (attraction), while low scores reflect undesirable behaviors that should be suppressed (repulsion).

We incorporate these scores into policy training through adaptive losses to guide the training of our flow matching policy with both successful and failed demonstrations in a balanced way as shown in Figure 2(c). Specifically, we first form a mixed dataset with both success and failure data, where the failure data used here are not the artificially generated failures from Section III-A, but come directly from the existing dataset. During training, the loss contributions of each failure trajectory segmentation are adaptively leveraged by the sequence importance score from the frozen progress prediction scorer. In this way, the policy is encouraged to focus on informative failure segments (*Progressive Segments*) while suppressing misleading ones (*Failure-Critical Segments*).

Based on the standard interpolation process of flow matching with target velocity

$$v^*(x_t, t) = \frac{dx_t}{dt} = x_1 - x_0, \tag{6}$$

we modify the flow matching objective conditioned on the scorer prediction, denoted as $g(x)$. With a predefined threshold, we first partition the failure data x_{fail} into following categories:

$$\begin{aligned} x_{\text{suc|fail}} &= \{x \in x_{\text{fail}} \mid g(x) \geq \tau_1\}, \\ x_{\text{fail|fail}} &= \{x \in x_{\text{fail}} \mid g(x) \leq \tau_2\}. \\ x_{\text{ind}} &= \{x \in x_{\text{fail}} \mid \tau_2 < g(x) < \tau_1\}. \end{aligned} \tag{7}$$

Among them, two subsets correspond to distinct and informative categories within the failure dataset: Progressive Segments $x_{\text{suc|fail}}$, which contain high-quality behavioral segments despite originating from failed trajectories, and Failure-Critical Segments $x_{\text{fail|fail}}$, which represent genuinely incorrect behaviors that policy should avoid. Both of them provide supervision signals in the flow matching training, while Indeterminate Segments x_{ind} are discarded. Based on this partition, the attraction-repulsion objective at each training iteration is formulated as:

$$L_{\text{suc}}(\theta) = \mathbb{E}_{(x,t)\sim x_{\text{suc}}}\left[\|v^*(x_t,t) - v_\theta(x_t,t)\|^2\right], \tag{8}$$

$$L_{\text{suc|fail}}(\theta) = \mathbb{E}_{(x,t)\sim x_{\text{suc|fail}}}\left[\|v^*(x_t,t) - v_\theta(x_t,t)\|^2\right], \tag{9}$$

$$L_{\text{fail|fail}}(\theta) = \mathbb{E}_{(x,t)\sim x_{\text{fail|fail}}}\left[\|v^*(x_t,t) - v_\theta(x_t,t)\|^2\right], \tag{10}$$

$$\begin{aligned} L(\theta) = L_{\text{suc}}(\theta) &+ \lambda_1 L_{\text{suc|fail}}(\theta) \\ &+ \lambda_2 \,\text{ReLU}\Big(\,\text{stopgrad}\big(L_{\text{suc|fail}}(\theta)\big) - L_{\text{fail|fail}}(\theta)\Big). \end{aligned} \tag{11}$$

where λ_1 and λ_2 are the hyperparameters to control the strengths of attraction and repulsion respectively.

It is worth noting that directly introducing a negative loss term, e.g., $-\mathcal{L}_{\text{fail|fail}}$, may lead to unstable optimization, since the objective can be reduced by indefinitely increasing the prediction error on failure-critical samples. To avoid such unbounded repulsion, we instead use the positive $\mathcal{L}_{\text{suc|fail}}$ as a dynamic reference and constrain the repulsion through a ReLU margin. Specifically, let

$$c = \text{stopgrad}\left(\mathcal{L}_{\text{suc|fail}}\right), \tag{12}$$

and denote the repulsion term as

$$\mathcal{R}(\theta) = \text{ReLU}\left(c - \mathcal{L}_{\text{fail|fail}}(\theta)\right). \tag{13}$$

Since both the flow-matching losses and $\mathcal{R}(\theta)$ are non-negative, the overall training objective remains lower bounded. Moreover, except at the ReLU boundary, gradient in policy training process is

$$\nabla_\theta \mathcal{R}(\theta) = \begin{cases} -\nabla_\theta \mathcal{L}_{\text{fail|fail}}(\theta), & \mathcal{L}_{\text{fail|fail}} < c, \\ 0, & \mathcal{L}_{\text{fail|fail}} \geq c. \end{cases} \tag{14}$$

Therefore, when the failure loss is smaller than the positive reference, minimizing $\mathcal{R}$ explicitly increases the flow-matching error on Failure-Critical Segments, producing the desired repulsive gradient. Once the failure loss exceeds the reference, the repulsive gradient vanishes. Meanwhile, $\mathcal{L}_{\text{suc|fail}}$ typically decreases as training progresses, yielding an increasingly conservative repulsion margin. In this way, CARF pushes the policy away from failure-critical behaviors only when necessary, while avoiding unbounded separation and maintaining stable optimization.

IV. Experiments

A. Experimental Setup

Manipulation Tasks. As shown in Figure 3, we conduct experiments on three different tasks from RLBench [37] simulation and two tasks in the real world. In simulation, failure data are collected by perturbing successful trajectories at annotated keypoints, following [38]. In real world, failure data come from natural failures of human operators during teleoperation, which means we record both success and failure data in the data collection process.

Patterns of failure data in simulation and real world are intentionally different. Generated failed trajectories in simulation are allowed to continue after the failure, producing long failure sequences that may contain both misleading post-failure actions and partially useful motions. In contrast, real-world teleoperation failures terminate once the operator identifies an unrecoverable mistake. These two settings represent complementary types of imperfect robotic data: deployment failures with post-failure continuation, and human-collected failures with early termination. This is designed to show that CARF can improve performance in both settings, suggesting that the proposed attraction-repulsion mechanism is not tied to a specific failure collection protocol.

Besides, these tasks cover diverse manipulation modes for evaluating our method. *Pick and Lift* requires moving the target red block to a marked position (red sphere). *Put*

Fig. 3: Environment visualization of tasks in simulation and real world.

TABLE I: Capability requirements of designed robotics tasks.

Tasks	Basic Interaction	Visual Object Detection	Precise Manipulation	Long Horizon
pick and Lift	✓	✓	✗	✗
put groceries in cupboard	✓	✓	✓	✗
put item in drawer	✓	✗	✓	✓
mug hanging tree	✓	✗	✓	✗
cooking sequence	✓	✗	✗	✓

Groceries in Cupboard requires placing a cereal box from a cluttered tabletop into a cupboard shelf. *Put Item in Drawer* involves opening a drawer and placing an object inside. *Mug Hanging Tree* requires hanging a mug by its handle on a pole. *Cooking Sequence* evaluates long-horizon execution through an ordered sequence of opening the pot, placing corn inside, and putting broccoli on the plate. As shown in Table I, these tasks span different levels of dexterity and task complexity.

Baselines. We compared our method with three different baselines:

- **Success-only IL** [25]: We perform imitation learning using flow matching policy. With exactly the same network as our method, we only use success data to train and guide policy using normal MSE loss.
- **All-data IL** [25]: With everything else same as Success-only IL, we instead use both success and failure data to train imitation learning policy.
- **CGFM** [33]: We enhance flow matching policy with classifier guidance. Specifically, a binary (success/failure) classifier is trained based on trajectory-level labels in the mixed dataset. Then, it is used to guide flow matching policy during inference to lead policy to go to targeted area.

Each policy is evaluated for 100 independent rollouts per task for simulation, and 20 independent rollouts per task for real world. All methods use the same initial-condition distribution and success criterion.

Implementation Details. For dataset preparation, we set $\beta = 1$ and $t_0 = 30$ for all tasks. The progress prediction scorer is implemented as a three-layer MLP. Given the one-step observation feature and the corresponding sixteen-step action chunk, the scorer predicts sixteen scalar progress scores (one for each action) and is optimized using mean squared error (MSE) against the automatically generated progress labels. We train the scorer independently for 10 epochs using AdamW with a learning rate of 3e-4 and a batch size of 64. Once trained, its parameters are frozen and the scorer is used only to assign importance scores during policy training and not required during policy inference. For CARF policy learning, all methods use the same flow-matching architecture and training configuration for fair comparison. Policies are also optimized using AdamW with a learning rate of 1e-5, a batch size of 64, and 50 training epoches.

For CARF parameters, we set $\lambda_1 = 0.05$ and $\lambda_2 = 0.01$ for the attraction and repulsion objectives. These two hyperparameters are shared across all tasks and are not tuned separately for individual environments or manipulation tasks. The thresholds τ_1 and τ_2 are determined according to the scale of the progress scores induced by the number of subtasks. Since the overall progress-score range is divided across subtasks, tasks with more subtasks have smaller per-subtask score increments and consequently use smaller threshold values. Therefore, τ_1 and τ_2 are adjusted according to the number of subtasks rather than independently tuned based on policy performance.

B. Experimental Results

Table II shows the main results of our methods compared to the aforementioned baselines. Simply incorporating all failed trajectories into imitation learning consistently degrades performance compared with training on successful demonstrations only. This result indicates that increasing the amount of training data does not necessarily benefit policy learning when the additional demonstrations contain heterogeneous and conflicting behaviors. In particular, failed trajectories should not be uniformly treated as additional expert demonstrations, motivating a more selective use of their informative segments. This also supports our treatment of Indeterminate Segments: behaviors with ambiguous contribution to task progress may introduce unreliable supervision if they are directly included in policy optimization. It is interesting to find out that CGFM actually reaches worse result than pure imitation learning. This reflects that classifier guidance is not suitable for this task which requires specific and precise distinguishment between different categories, especially with small dataset. In this case, the boundary between different categories is hard to recognize, which leads to wrong gradient during policy inference.

TABLE II: Quantitative result comparison between CARF and other baselines.

Dataset & Methods	Simulation			Real World	
	pick and lift	*put groceries in cupboard*	*put item in drawer*	*mug hanging tree*	*cooking sequence*
Success data	200	200	200	100	50
Failure data	56	110	68	26	29
Success-only IL [25]	0.38	0.23	0.62	0.50	0.55
All-data IL [25]	0.32 (-15.8%)	0.20 (-13.0%)	0.53 (-14.5%)	0.35 (-30.0%)	0.45 (-18.2%)
CGFM [33]	0.24 (-36.8%)	0.14 (-39.1%)	0.47 (-24.2%)	0.45 (-10.0%)	0.40 (-27.3%)
Ours	**0.48 (+26.3%)**	**0.28 (+21.7%)**	**0.69 (+11.3%)**	**0.85 (+70.0%)**	**0.70 (+27.3%)**

TABLE III: Ablation study on attraction-repulsion items.

Task	*pick and lift*	*put grocery in cupboard*	*mug hanging tree*
Success-only IL	0.38	0.23	0.50
IL *w/* only positive terms	0.39	0.21	0.70
IL *w/* only negative terms	0.43	0.26	0.60
CARF (ours)	**0.48**	**0.28**	**0.85**

On the other hand, instead of directly fitting the limited training distribution or relying on unstable classifier gradients, our method explicitly exploits the usefulness of failed trajectories and selectively incorporates informative samples into policy learning. Therefore, it can improve data efficiency while reducing the risk of introducing misleading supervision from low-quality trajectories. More importantly, our results show that meaningful improvement can still be achieved by leveraging only a limited amount of imperfect data, instead of relying on large-scale demonstrations to learn an explicit boundary between correct and incorrect behaviors.

Table III presents ablation results to see the actual effect on attraction and repulsion design. From the table, we can see that for most of the time, attraction and repulsion will solely take relatively smaller effect on the result, but combine them together will improve the final results most for both failure setups. It demonstrate that attraction and repulsion play complementary roles: attraction extracts useful behavior from imperfect trajectories, while repulsion prevents the policy from imitating failure-critical actions. Their combination therefore enables more effective utilization of failed demonstrations than either component alone.

C. Scorer performance on failure trajectories

To investigate whether our scorer can identify different trajectory types, we visualize the sequence-level importance scores derived from the scorer predictions within a failed trajectory, as shown in Figure 4. We observe that the importance scores soar upon the completion of a subtask, while they drop below zero when obvious errors occur or the trajectory starts deviating. This phenomenon is fully consistent with our expectation. Notably, in some segments, although the action is incorrect when considered as a whole trajectory, it may still contain partially correct information. This can lead to a neutralized importance score close to zero, similar to the value observed when the agent is not close enough but still approaching to the target object. This explains why we only use the high-value and low-value regions for training. Such cases are inevitable in simulation tasks: even if a previous subtask fails, the policy continues to execute subsequent subtasks. In contrast, this situation does not happen in real-world tasks, where execution ends immediately when a failure is detected. This may also explain why our real-world results are slightly better since these misleading actions, mixed with correct motion patterns but actually caused by earlier failures, do not appear in real-world executions. This finding is consistent with this interpretation: early-terminated failures contain fewer misleading post-failure actions, making the separation between useful and harmful segments cleaner for the scorer.

D. Visualization about failure improvement

In this section, we compare the performance differences between our method and the Success-only IL baseline, as shown in Figure 5. The results show that our method addresses several key weaknesses of imitation learning. **1) Accuracy.** Success-only IL can be confused by the scene and fail to perform precise motions, while our method correctly identifies, lifts and places the object into right place. **2) Smoothness.** Success-only IL often trembles during inference, preventing precise grasp execution. In contrast, our method produces smoother trajectories and completes the task successfully. **3) Speed.** Due to repeated grasping attempts, Success-only IL may fail to finish the task within a reasonable number of steps, while our method directly reaches and fetches the object. **4) Stability.** In continuous-contact tasks, Success-only IL can produce unstable motions that cause the target object to slip, whereas our method maintains stable contact until reaching the final destination.

Overall, these observations show that our method improves both effectiveness and execution quality. Compared with Success-only IL, our method is better at resolving ambiguous observations, generating smooth and stable motions, and completing tasks within fewer interaction steps. This indicates that scorer-guided learning can provide more robust demonstrations for complex manipulation, leading to policies that are better aligned with real execution requirements.

V. Conclusion and Limitation

Overall, CARF provides a practical way to use imperfect robotic data by separating *Progressive Segments* and

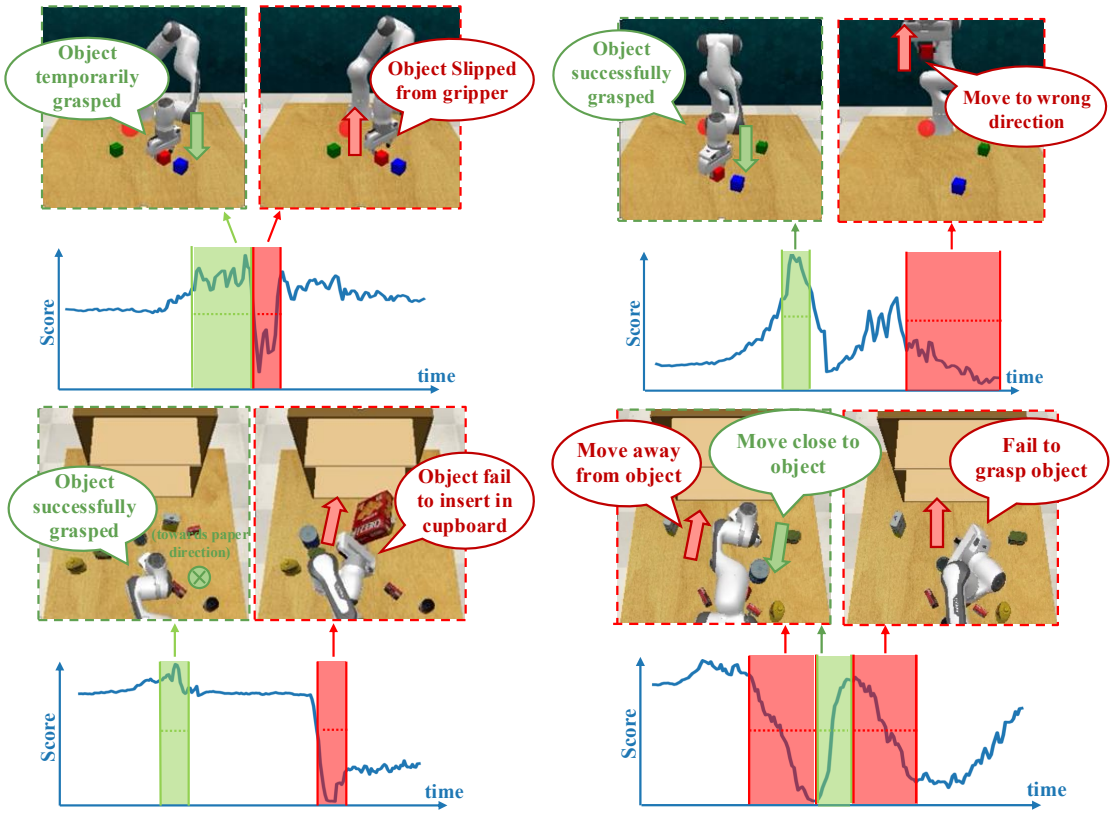

Fig. 4: Scorer performance on representative failure trajectories. Green and Red respectively mean *Progressive Segments* and *Failure-Critical Segments*, and arrows in pictures mean current action moving direction of gripper end effector.

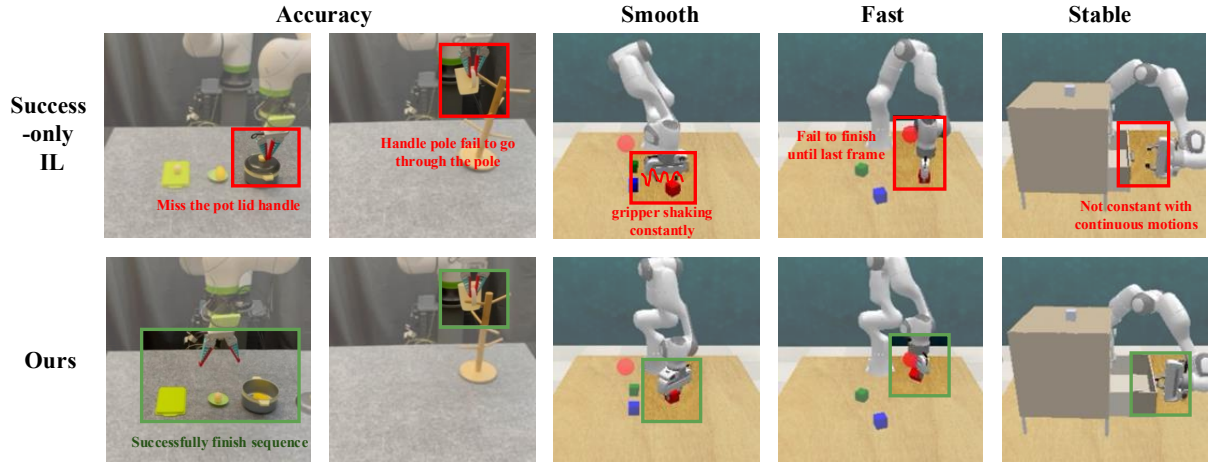

Fig. 5: Visualization of failure improvement, where the improvement parts are annotated in the pictures using highlight rectangles with text descriptions.

Failure-Critical Segments in failed trajectories. Rather than relying on value bootstrapping or selectively reusing only progressive segments from failed trajectories, CARF introduces a progress-based scorer and a bounded attraction–repulsion objective to shape flow-matching policy learning. Across simulated and real-world manipulation tasks, CARF improves over imitation learning baselines, suggesting that segment-level failure guidance can improve data efficiency and execution robustness.

CARF still has several limitations. First, the current scorer uses gripper-state transitions for subtask segmentation, which works well for grasp-centric tasks but may be less suitable for continuous-contact manipulation. Future work can explore more general segmentation cues based on contact, force, or visual progress. Second, CARF adopts a conservative strategy that only utilizes high-confidence Progressive and Failure-Critical Segments, while discarding Indeterminate Segments with ambiguous supervision. Although this reduces

the risk of introducing misleading training signals, potentially useful information contained in these intermediate segments remains underutilized. Future work could incorporate uncertainty-aware or soft weighting mechanisms to make more complete use of imperfect demonstrations. Finally, the current threshold selection still depends on the task-specific scale of the progress representation, which requires lightweight calibration according to the subtask structure. Although the attraction and repulsion weights are shared across all tasks, developing an adaptive or task-agnostic thresholding strategy could further reduce manual design.

REFERENCES

[1] C. Chi, Z. Xu, C. Pan, E. Cousineau, B. Burchfiel, S. Feng, R. Tedrake, and S. Song, "Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots," *arXiv preprint arXiv:2402.10329*, 2024.

[2] M. Xu, H. Zhang, Y. Hou, Z. Xu, L. Fan, M. Veloso, and S. Song, "Dexumi: Using human hand as the universal manipulation interface for dexterous manipulation," *arXiv preprint arXiv:2505.21864*, 2025.

[3] H.-S. Fang, B. Romero, Y. Xie, A. Hu, B.-R. Huang, J. Alvarez, M. Kim, G. Margolis, K. Anbarasu, M. Tomizuka *et al.*, "Dexop: A device for robotic transfer of dexterous human manipulation," *arXiv preprint arXiv:2509.04441*, 2025.

[4] S. Zhao, X. Zhu, Y. Chen, C. Li, Y. Xie, X. Zhang, M. Ding, and M. Tomizuka, "Dexh2r: Task-oriented dexterous manipulation from human to robots," *IEEE/ASME Transactions on Mechatronics*, 2025.

[5] Y. Mao, J. Fu, R. Zhang, H. Xie, and M. Yao, "Beyond success: Refining elegant robot manipulation from mixed-quality data via just-in-time intervention," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026, pp. 13 508–13 518.

[6] G. Zheng, S. Seenivasan, M. Johnson-Roberson, and W. Zhi, "Rewind-il: Online failure detection and state respawning for imitation learning," *arXiv preprint arXiv:2604.16683*, 2026.

[7] Y. Xie, Y. Wang, S. Zhao, C.-E. Wu, M. Tomizuka, J. Xie, and H.-S. Fang, "Multi-camera view scaling for data-efficient robot imitation learning," *arXiv preprint arXiv:2604.00557*, 2026.

[8] J. Chen, I. Liu, G. Sukhatme, D. Seita *et al.*, "Craft: Video diffusion for bimanual robot data generation," *arXiv preprint arXiv:2604.03552*, 2026.

[9] G.-H. Kim, S. Seo, J. Lee, W. Jeon, H. Hwang, H. Yang, and K.-E. Kim, "Demodice: Offline imitation learning with supplementary imperfect demonstrations," in *International Conference on Learning Representations*, 2021.

[10] Y. Huang, N. Alvina, M. D. Shanthi, and T. Hermans, "Fail2progress: Learning from real-world robot failures with stein variational inference," *arXiv preprint arXiv:2509.01746*, 2025.

[11] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín, "What matters in learning from offline human demonstrations for robot manipulation," in *Conference on Robot Learning (CoRL)*, 2021.

[12] K. Huang, R. Scalise, C. Winston, A. Agrawal, Y. Zhang, R. Baijal, M. Grotz, B. Boots, B. Burchfiel, M. Itkina *et al.*, "Using non-expert data to robustify imitation learning via offline reinforcement learning," *arXiv preprint arXiv:2510.19495*, 2025.

[13] P. Intelligence, A. Amin, R. Aniceto, A. Balakrishna, K. Black, K. Conley, G. Connors, J. Darpinian, K. Dhabalia, J. DiCarlo *et al.*, "$\pi^*_{0.6}$: a vla that learns from experience," *arXiv preprint arXiv:2511.14759*, 2025.

[14] V. Giridhar, A. Khandelwal, J. A. Collins, I. Georgiev, and A. Garg, "Beyond imitation: Self-improving robot policies via off-policy q-planning," *arXiv preprint arXiv:2608.21204*, 2026.

[15] K. Wu, N. Liu, Z. Zhao, D. Qiu, J. Li, Z. Che, Z. Xu, and J. Tang, "Learning from imperfect demonstrations with self-supervision for robotic manipulation," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 16 899–16 906.

[16] L.-H. Lin, Y. Cui, A. Xie, T. Hua, and D. Sadigh, "Flowretrieval: Flow-guided data retrieval for few-shot imitation learning," *arXiv preprint arXiv:2408.16944*, 2024.

[17] M. Du, S. Nair, D. Sadigh, and C. Finn, "Behavior retrieval: Few-shot imitation learning by querying unlabeled datasets," *arXiv preprint arXiv:2304.08742*, 2023.

[18] Y. Zheng, P. Yang, Z. Xia, Q. Zhang, Y. Zheng, S. Gu, B. Jin, T. Zhang, B. Lu, C. Han *et al.*, "Data scaling laws for imitation learning-based end-to-end autonomous driving," *arXiv preprint arXiv:2412.02689*, 2024.

[19] S. Xia, H. Fang, C. Lu, and H.-S. Fang, "Cage: Causal attention enables data-efficient generalizable robotic manipulation," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 13 242–13 249.

[20] S. Zhao, K. Yang, Y. Chen, C. Li, Y. Xie, X. Zhang, C. Wang, and M. Tomizuka, "Dexctrl: Towards sim-to-real dexterity with adaptive controller learning," *arXiv preprint arXiv:2505.00991*, 2025.

[21] R. Zheng, D. Niu, Y. Xie, J. Wang, M. Xu, Y. Jiang, F. Castañeda, F. Hu, Y. L. Tan, L. Fu *et al.*, "Egoscale: Scaling dexterous manipulation with diverse egocentric human data," *arXiv preprint arXiv:2602.16710*, 2026.

[22] K. Xu, Z. Zhu, A. Chen, S. Zhao, Q. Huang, Y. Yang, H. Lu, R. Xiong, M. Tomizuka, and Y. Wang, "Seeing to act, prompting to specify: A bayesian factorization of vision language action policy," *arXiv preprint arXiv:2512.11218*, 2025.

[23] 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.

[24] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," *The International Journal of Robotics Research*, vol. 44, no. 10-11, pp. 1684–1704, 2025.

[25] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow matching for generative modeling," *arXiv preprint arXiv:2210.02747*, 2022.

[26] G. Stoica, V. Ramanujan, X. Fan, A. Farhadi, R. Krishna, and J. Hoffman, "Contrastive flow matching," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025, pp. 1185–1194.

[27] H. Xu, X. Zhan, H. Yin, and H. Qin, "Discriminator-weighted offline imitation learning from suboptimal demonstrations," in *International Conference on Machine Learning*. PMLR, 2022, pp. 24 725–24 742.

[28] S. Dass, A. Khaddaj, L. Engstrom, A. Madry, A. Ilyas, and R. Martín-Martín, "Datamil: Selecting data for robot imitation learning with datamodels," *arXiv preprint arXiv:2505.09603*, 2025.

[29] A. Wei, N. Pfaff, T. Cohn, A. K. Dayı, C. Daskalakis, G. Daras, and R. Tedrake, "Ambient diffusion policy: Imitation learning from suboptimal data in robotics," *arXiv preprint arXiv:2606.12365*, 2026.

[30] M. Zheng, S. Marri, A. Choudhuri, B. Planche, Z. Gao, V. N. Nguyen, T. Chen, G. Chowdhary, and Z. Wu, "Failing forward: Adaptive failure-informed learning for vision-language-action models," *arXiv preprint arXiv:2605.08434*, 2026.

[31] H. Hao, S. N. Syed, J. Ichnowski, and J. Schneider, "Far: Failure-aware retry for test-time recovery and continual policy improvement," *arXiv preprint arXiv:2607.01111*, 2026.

[32] H. Yan, Q. Li, J. Yang, and Y. Mu, "Progressvla: Progress-guided diffusion policy for vision-language robotic manipulation," *arXiv preprint arXiv:2603.27670*, 2026.

[33] P. Dhariwal and A. Nichol, "Diffusion models beat gans on image synthesis," *Advances in neural information processing systems*, vol. 34, pp. 8780–8794, 2021.

[34] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative q-learning for offline reinforcement learning," *Advances in neural information processing systems*, vol. 33, pp. 1179–1191, 2020.

[35] I. Kostrikov, A. Nair, and S. Levine, "Offline reinforcement learning with implicit q-learning," *arXiv preprint arXiv:2110.06169*, 2021.

[36] P. Hansen-Estruch, I. Kostrikov, M. Janner, J. G. Kuba, and S. Levine, "Idql: Implicit q-learning as an actor-critic method with diffusion policies," *arXiv preprint arXiv:2304.10573*, 2023.

[37] S. James, Z. Ma, D. R. Arrojo, and A. J. Davison, "Rlbench: The robot learning benchmark & learning environment," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3019–3026, 2020.

[38] J. Duan, W. Pumacay, N. Kumar, Y. R. Wang, S. Tian, W. Yuan, R. Krishna, D. Fox, A. Mandlekar, and Y. Guo, "Aha: A vision-language-model for detecting and reasoning over failures in robotic manipulation," *arXiv preprint arXiv:2410.00371*, 2024.