2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Quality over Quantity: Demonstration Curation via Influence Functions for Data-Centric Robot Learning

Haeone Lee*[1], Taywon Min[1] Junsu Kim[1], Sinjae Kang[1], Fangchen Liu[2], Lerrel Pinto[3], and Kimin Lee[1]

Abstract— **Learning from demonstrations has emerged as a promising paradigm for end-to-end robot control, particularly when scaled to diverse and large datasets. However, the quality of demonstration data, often collected through human teleoperation, remains a critical bottleneck for effective data-driven robot learning. Human errors, operational constraints, and teleoperator variability introduce noise and suboptimal behaviors, making data curation essential yet largely manual and heuristic-driven. In this work, we propose Quality over Quantity (QoQ), a grounded and systematic approach to identifying high-quality data by defining data quality as the contribution of each training sample to reducing loss on validation demonstrations. To efficiently estimate this contribution, we leverage influence functions, which quantify the impact of individual training samples on model performance. We further introduce two key techniques to adapt influence functions for robot demonstrations: (i) using maximum influence across validation samples to capture the most relevant state-action pairs, and (ii) aggregating influence scores of state-action pairs within the same trajectory to reduce noise and improve data coverage. Experiments in both simulated and real-world settings show that QoQ consistently improves policy performances over prior data selection methods.**

I. INTRODUCTION

Learning from demonstrations has shown potential in end-to-end robot control, particularly when scaling both the diversity and quantity of demonstration data [1]–[7]. However, the quality of robot demonstration data, commonly collected through human teleoperation, significantly impacts performance when training with supervised learning methods like behavior cloning (BC) [8], [9]. Human errors, operational constraints, and varying skill levels of teleoperators introduce noise and suboptimal behaviors into these datasets, making effective curation critical for successful data-driven robot learning.

Despite its importance, data curation remains largely manual, expensive, and reliant on heuristic judgments. Previous approaches have attempted to address this challenge using proxy metrics, including similarity to expert demonstrations [10], [11] and mutual information between state and action distributions [12]. However, these metrics often fail to capture which training data truly contributes to improved policy performance.

In this work, we call for a data curation framework based on influence functions [13]. Influence functions can measure the contribution of training data to reducing loss on a small set of validation demonstrations that represent the target desired behavior (see Figure 1). We remark that performance on unseen validation samples serves as a useful metric for a policy's generalization capabilities, thereby allowing our definition to capture the complex relationship between training data and policy performance.

However, we find that naively applying influence functions to robot demonstrations yields noisy signals and tends to select redundant state-action pairs, resulting in poor coverage of the state space (see Section V-E). To address these issues, we propose **Q**uality **o**ver **Q**uantity (**QoQ**), which introduces two key techniques for effectively applying influence functions to robot demonstrations: First, we measure each state-action pair's influence by its maximum influence across validation samples, focusing only on the most relevant validation state-action pair rather than averaging across all validation data. Second, we implement trajectory-wise curation, which aggregates influence scores of state-action pairs within the same trajectory and then selects high-quality trajectories based on the aggregated scores. This approach significantly reduces noise in the influence signal while ensuring broad state coverage, capturing diverse and informative robot behaviors.

We evaluate QoQ on both Robomimic simulation [9] and multiple real-robot manipulation tasks. Our experiments show that QoQ successfully filters out low-quality demonstrations (*e.g.,* failure cases), significantly improving policy performance when trained on curated datasets. By considering the true effect to the policy in data curation, QoQ substantially outperforms the baseline methods that rely on state and action features [10], [11] up to 23.2% in Robomimic simulation and 30.0% in real robot success rate. We further demonstrate that QoQ can curate high-quality data from the DROID [3] dataset, which was collected in the wild and encompasses diverse environments and object locations.

II. RELATED WORK

A. Data valuation

Data valuation aims at quantifying the contribution of individual data points to the performance of machine learning models. One approach is Data Shapely [14]–[16], which leverages cooperative game theory to data valuation by computing the average marginal contribution of each datapoint to the models' performance across all possible subsets of training data. Influence functions [13] offer another approach by estimating the effect of upweighting or removing a training point on the models' parameters and loss, without requiring full retraining. This is achieved via first-order ap-

[1]KAIST [2]University of California, Berkeley [3]New York University.
∗ Correspondence to haeone.lee@kaist.ac.kr

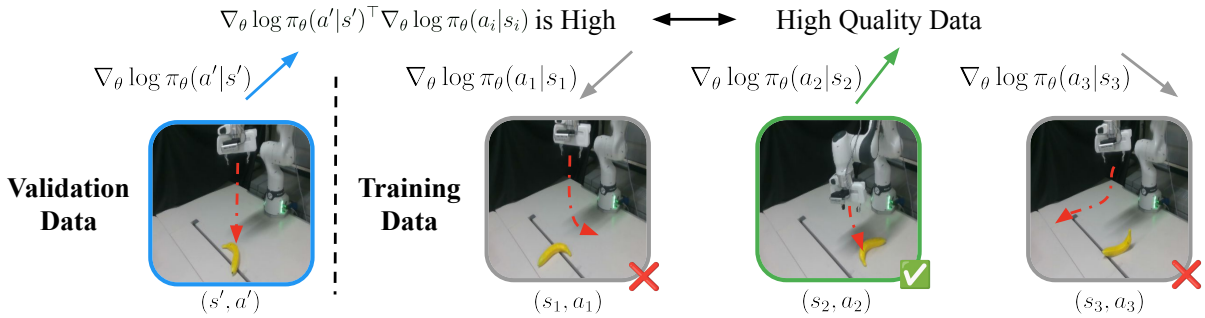

Fig. 1: **Illustration of high-quality data.** Our robot data curation method, QoQ, selects trajectories based on *their direct contribution to policy performance*, using influence functions [13] to quantify this impact. Specifically, we measure the similarity between gradients of validation data (blue) and those of training data; higher similarity (green) indicates that including a particular state-action pair will effectively reduce validation loss. By prioritizing these high-impact data points, we systematically identify and preserve the most valuable training data that drive performance improvement.

proximations involving gradients and inverse Hessian-vector products, enabling efficient estimation.

B. Robot data curation

To filter low-quality trajectories, previous work proposed their own definition of the quality of robot data. For example, retrieval-based approaches [10], [17], [18] define it as similar points with expert data in feature space. Flow retrieval [11] extends this idea by representing trajectories using optical flow to better capture temporal dynamics. DemInf [12] curates the training dataset by selecting trajectories with high mutual information between states and actions as a proxy for demonstration quality. Demo-SCORE [19] curates trajectories that are reproducible for the robot to imitate; it filters them using a classifier trained on policy rollouts to identify trajectories that consistently lead to task success. Compared to prior work, we define robot data quality based on the *direct performance contribution* to the learned policy.

Related to our work, DataMIL [20] and CUPID [21] also attribute robotic data contributions using influence functions. CUPID uses returns estimated from policy rollouts to curate robot data. When using only successful trajectories of return 1.0 as a validation set, CUPID score is equivalent to measuring each state-action pair's influence by its sum over the entire validation transitions. Similarly, DataMIL uses the validation loss computed over the entire validation set. However, we found that this leads to unstable influence score estimations as shown in our experiment. This is because not all validation transitions are helpful to evaluate the quality of state-action pairs of interest, since they could correspond to different behaviors (e.g., pick-and-place behavior might not help identify useful behavior for screwing). Therefore, we measure each state-action pair's influence by its maximum influence across validation samples, focusing only on the most relevant validation state-action pair.

III. Preliminaries

A. Learning from Demonstrations

We model the robot learning problem as a sequential decision-making process, where a policy outputs an action a given a state s. We consider a setting in which a robot learns from a dataset of demonstrations using Behavior Cloning (BC) [22]. The dataset $\mathcal{D}$ consists of trajectories, each representing a sequence of states and actions:

$$\mathcal{D} = \{\tau_i\}_{i=1}^{N}, \quad \text{where} \quad \tau_i = (s_0, a_0, s_1, a_1, \dots).$$

Given this dataset $\mathcal{D}$, we train the policy π_θ to imitate the demonstrated behavior by minimizing the BC loss function, defined as the negative log-likelihood of the demonstrated actions:

$$\mathcal{L}_{\texttt{BC}}(\mathcal{D}; \theta) = \mathbb{E}_{(s,a)\sim\mathcal{D}} \left[-\log \pi_\theta(a|s) \right],$$

where θ denotes the parameters of the policy.

B. Influence Functions

Influence functions [13] approximates how the model parameters, or a function of the model parameters (*e.g.*, validation loss) changes regarding a specific training point $(x_i, y_i) \in \mathcal{D}_{\texttt{tr}}$.

Specifically, given a loss function $\mathcal{L}$ and a training dataset $\mathcal{D}_{\texttt{tr}}$, the ε-weighted risk minimizer for a single training data point (x_i, y_i) is defined as $\theta^{(i)}(\varepsilon) := \arg\min_{\theta\in\Theta} \frac{1}{|\mathcal{D}_{\texttt{tr}}|} \sum_{(x,y)\in\mathcal{D}_{\texttt{tr}}} \mathcal{L}(f_\theta(x), y) + \varepsilon \mathcal{L}(f_\theta(x_i), y_i)$.

This indicates that a single training sample, (x_i, y_i), has been up-weighted by a small perturbation ε. The influence function $\mathcal{I}_\theta(x_i, y_i)$ is defined as the derivative of $\theta^{(i)}(\varepsilon)$ at $\varepsilon = 0$, quantifying how the model parameters change due to the data point (x_i, y_i):

$$\mathcal{I}_\theta(x_i, y_i) := \left. \frac{d\theta^{(i)}(\varepsilon)}{d\varepsilon} \right|_{\varepsilon=0}. \tag{1}$$

While this describes how model parameters shift, our primary goal is to quantify how each training sample influences validation loss. This can be achieved by applying the chain rule:

$$\mathcal{I}_{\texttt{val}}(x_i, y_i) := \nabla_\theta \mathcal{L}(\mathcal{D}_{\texttt{val}}; \theta)^\top \left. \frac{d\theta^{(i)}(\varepsilon)}{d\varepsilon} \right|_{\varepsilon=0}, \tag{2}$$

where $\mathcal{D}_{\texttt{val}}$ denotes the validation dataset, and $\mathcal{L}(\mathcal{D}_{\texttt{val}}; \theta)$ is the validation loss. The influence function $\mathcal{I}_{\texttt{val}}(x_i, y_i)$

estimate the degree of change in validation loss when a training sample (x_i, y_i) is up-weighted. Under standard assumptions (twice-differentiability and strong convexity), this can be approximated as:

$$\mathcal{I}_{\texttt{val}}(x_i, y_i) := -\nabla_\theta \mathcal{L}(\mathcal{D}_{\texttt{val}}; \theta)^\top H(\mathcal{D}_{\texttt{tr}}; \theta)^{-1} \nabla_\theta \mathcal{L}(x_i, y_i; \theta) \quad (3)$$

$H(\mathcal{D}_{\texttt{tr}}; \theta) := \nabla^2_\theta \mathcal{L}(\mathcal{D}_{\texttt{tr}}; \theta)$ is the Hessian matrix of the training loss, and θ is the minimizer of the empirical risk over $\mathcal{D}_{\texttt{tr}}$. A lower value of $\mathcal{I}_{\texttt{val}}(x_i, y_i)$ indicates the sample decreases the validation loss (potentially beneficial) and vice versa.

However, computing the inverse Hessian in eq. (3) is often computationally prohibitive. To address this, [23] proposed a first-order approximation that omits the Hessian. Furthermore, they found that normalizing gradients improves the stability of influence estimation. With this modification, influence functions become:

$$\mathcal{I}_{\texttt{val}}(x_i, y_i) := -\nabla'_\theta \mathcal{L}(\mathcal{D}_{\texttt{val}}; \theta)^\top \nabla'_\theta \mathcal{L}(x_i, y_i; \theta), \quad (4)$$

where the normalized gradient is defined as $\nabla'_\theta \mathcal{L} := \frac{\nabla_\theta \mathcal{L}}{\|\nabla_\theta \mathcal{L}\|_2}$. In our work, we utilize eq. (4) to estimate influence values on robot datasets.

IV. Quality over Quantity (QoQ)

In this section, we introduce **Q**uality **o**ver **Q**uantity (**QoQ**), a method for curating high-quality robot data using influence functions. In Section IV-A, we define what constitutes high-quality data. In Section IV-B, we describe how we systematically identify such data using influence functions.

A. What Counts as High-Quality Robot Data?

Defining high-quality demonstration data is a challenging problem in data-driven robot learning. Existing approaches have adopted varying criteria: some researchers prioritize optimal behaviors (*e.g.,* shortest path to target objects) [9], while others emphasize diversity [3], [8], robustness [24]. However, these predefined notions of quality often fail to capture the complex relationship between training data characteristics and final policy performance.

We propose a more grounded definition: robot data quality should be measured by its *direct performance contribution to the learned policy*. Specifically, we quantify a demonstration's quality through its contribution to reducing loss on a small set of validation set consisting of desirable behavior.[1] To calculate this contribution efficiently, we employ influence functions [13] (detailed in Section III-B), which estimate how validation loss would change if individual training samples were removed, without the computational cost of retraining. Intuitively, understanding how the removal of a training sample affects validation loss allows us to precisely quantify the value of that sample. We note that from a machine learning perspective, performance on unseen validation samples serves as a natural proxy for a policy's generalization capabilities. This performance-based definition inherently captures the complex relationship between training data characteristics and policy effectiveness, addressing the limitations of predefined quality criteria.

[1]This small validation set can consist of held-out teleoperation data or actual policy rollouts from trained models.

B. Curating High-Quality Data using Influence Functions

Now, we present **Q**uality **o**ver **Q**uantity (**QoQ**), a data curation method based on influence functions. Formally, given a training dataset $\mathcal{D}_{\texttt{tr}}$ and a trained policy model $\pi_{\theta_{\texttt{tr}}}$, we curate high-quality data using a small validation dataset $\mathcal{D}_{\texttt{val}}$ through the following steps:

- *Step 1 (Contribution estimation)*: For each state-action pair $(s, a) \in \mathcal{D}_{\texttt{tr}}$, we quantify its contribution to reducing loss on the validation dataset $\mathcal{D}_{\texttt{val}}$ based on influence functions.
- *Step 2 (Trajectory-wise curation)*: We aggregate the influence scores of state-action pairs within each trajectory and select the top N trajectories based on the aggregated scores.

We describe the details of each step in the following paragraphs.

Step 1: Contribution estimation For each state-action pair $(s, a) \in \mathcal{D}_{\text{tr}}$, we define

$$g(s, a) := \nabla_\theta \log \pi_{\theta_{\texttt{tr}}}(a|s) / \|\nabla_\theta \log \pi_{\theta_{\texttt{tr}}}(a|s)\|, \quad (5)$$

and compute the following score to measure its contribution to reducing loss on the validation dataset $\mathcal{D}_{\texttt{val}}$:

$$\texttt{QoQ-score}(s, a) := \max_{(s', a') \in \mathcal{D}_{\texttt{val}}} g(s', a')^\top g(s, a). \quad (6)$$

where $\pi_{\theta_{\texttt{tr}}}$ denotes a policy model trained via BC and $g(s, a)$ is the normalized gradient on log-likelihood. This score, based on the negative of influence functions in eq. (4), measures gradient similarity between validation data and training state-action pairs. This similarity indicates the contribution of training state-action pairs to reducing validation loss, where a high score implies helpful training data.

Unlike the original formulation of influence functions that averages gradient products over all validation samples (*i.e.,* $\sum_{(s', a') \in \mathcal{D}_{\texttt{val}}} \nabla_\theta \log \pi_{\theta_{\texttt{tr}}}(a'|s')^\top \nabla'_\theta \log \pi_{\theta_{\texttt{tr}}}(a|s)$), we instead take the maximum gradient product across validation samples, which we call *maximum influence scoring*. Because each state-action pair in the validation trajectory represents different behaviors, we focus on the most relevant pair by taking the maximum, which helps reduce noise. Our experiments confirm that the maximum influence scoring enhances the reliability of influence estimation, resulting in better performance (see Section V-E for supporting results).

However, computing and storing gradients in eq. (6) for each sample poses substantial computational challenges for modern robot foundation models with billions of parameters [4]–[6]. To address this, we implement two complementary efficiency strategies: First, we selectively compute gradients for only a subset of network layers, specifically excluding parameter-dense components such as vision

encoders. Second, we employ the one-permutation one-random-projection (OPORP) technique [25] to compress gradient vectors while preserving their dot product relationships. This compression approach, inspired by recent advances in compute-efficient influence calculation [26], [27], significantly reduces storage requirements without compromising the accuracy of our scoring mechanism.

Step 2: Trajectory-wise curation After computing the proposed metric $\texttt{QoQ-score}(s, a)$ in eq. (6), we aggregate these scores within each trajectory τ by taking the mean: $\frac{1}{|\tau|}\sum_{(s,a)\in\tau}\texttt{QoQ-score}(s, a)$. We then select the top N trajectories based on these aggregated scores. We adopt trajectory-wise curation specifically to mitigate issues that arise from naively selecting individual state-action pairs with high scores. In our preliminary experiments, we observed that state-action-wise curation often results in redundant state-action pairs. For instance, specific behaviors like grasping moments were disproportionately selected, while other behaviors (such as reaching motions or diverse, multi-modal successful strategies) were filtered out. This leads to poor state coverage, which is undesirable for robust policy learning. By selecting entire trajectories instead, we ensure that the curated dataset maintains diverse state distributions and captures complete behavior sequences. Our empirical results in Section V-E confirm that trajectory-wise curation significantly outperforms state-action-wise curation across multiple benchmarks.

V. Experiments

We design our experiments to answer the following questions:

- (1) Can QoQ data curation improve policy success rates? (Section V-B)
- (2) How robust is QoQ to in-the-wild robot data under diverse domains? (Section V-C)
- (3) Can QoQ leverage policy rollout as validation set? (Section V-D)
- (3) How does each QoQ component impact performance? (Section V-E)

A. Setups

Environments We evaluate QoQ in both simulation and real-robot setups spanning multiple environments and tasks. For the simulation experiments, we use the Robomimic benchmark [9], where a Franka Research 3 robot arm is tasked with placing a Coke can into the correct bin. For the real-robot experiments, we use a Franka Research 3 robot arm operated via teleoperation to collect training datasets for three tasks: banana grasping, multi-object pick-and-place, and cabinet opening, as shown in Figure 3. We describe each task setup in detail sections below.

Baselines For comparison, we mainly consider the following baselines: (1) **All data**: utilizing original demonstrations without any curation. (2) **Behavior Retrieval** [10]: a curation method that retrieves state-action pairs from the training dataset based on their similarity to state-action pairs in the validation dataset. Specifically, it maps state-action pairs from both datasets into a latent space using a Variational Autoencoder (VAE) [28], and computes similarity based on distances in the latent space. (3) **Flow Retrieval** [11]: a curation method similar to Behavior Retrieval that maps optical flow derived from image sequences into a latent space using a VAE, and computes similarity based on distances. Unlike Behavior Retrieval, Flow Retrieval relies solely on optical flow information and does not utilize explicit state-action pairs.

Metrics For evaluation, we measure **curation accuracy**, defined as the proportion of successful trajectories in the curated dataset. We also report the **success rate** of policies trained on the curated datasets.

Experiment details For simulated experiments, we use the Transformer policy [29] with two self-attention layers with a pretrained ResNet encoder [30]. We train for 1k epochs using a batch size of 100. For real robot experiments, we use GR00T N1 [5], a high-performing vision-language-action model, and train for 20k steps using LoRA [31] with $\alpha = 64$ and rank $r = 128$. GR00T N1 model is trained via flow matching loss [32] and we use the gradient of flow matching loss in place of log-likelihood when computing QoQ scores. This approach has previously been adopted for data attribution in flow matching model classes, as it can be seen as a variational lower bound of the likelihood [33], [34]. Also, we use gradients from the Transformer blocks in the simulation experiment and from the action-head layers of GR00T N1 in the real robot experiment. In Section V-E, we examine the impact that varying the curation budget and gradient calculation layers has on the curation accuracy and policy success rates. Finally, we construct the curated dataset by selecting the top N trajectories with the highest $\texttt{QoQ-score}$ from the training dataset. The curation budget N is set to match the number of successful trajectories in the training dataset. If the number of successful trajectories is unknown, we set the curation budget to half the size of the training dataset. For all experiments, validation trajectories are randomly selected.

B. Can QoQ Data Curation Improve Policy Success Rates?

Single task curation We first evaluate QoQ and baseline curation methods by curation accuracy in the single task of pick and place of a Coke can in simulation, and the banana grasping experiment in a real robot.

In simulation, similar to Behavior Retrieval [10], we use the "Can-paired" dataset [9] as our train dataset, consisting of 100 successful and 100 failed trajectories, where successful trajectories are characterized by placing the Coke can into the correct bin. By using 10 successful trajectories as a validation set, we aim to curate a dataset that leads to successful task completion by filtering failures.[2]

[2]The trajectories in the validation dataset involve grasping and moving the can, but do not include the reaching behavior, as both successful and failed trajectories share the same reaching behavior. For the baseline methods, however, we retain the reaching behavior in the validation set, as these methods retrieve individual state-action pairs and perform poorly when the reaching behavior is excluded.

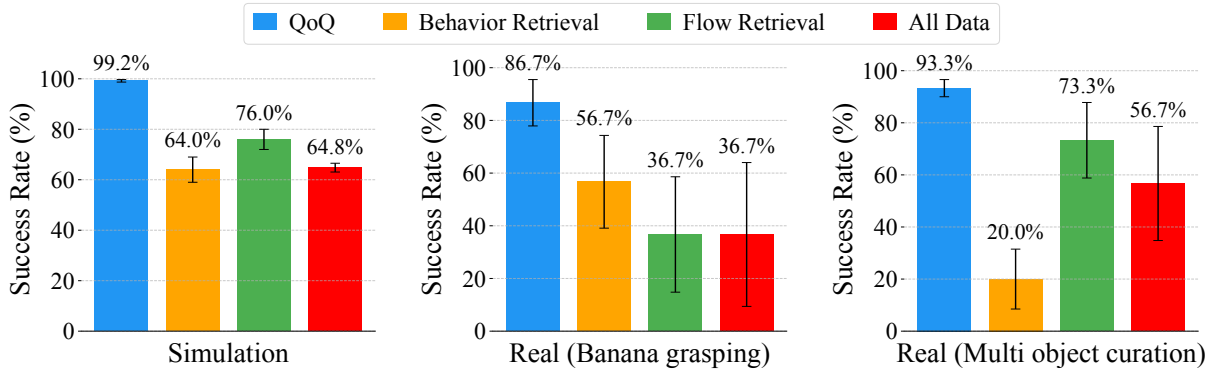

Fig. 2: **Success rate for simulation and real robot experiments.** QoQ outperforms all baselines in simulation and real robot experiments by detecting helpful trajectories. We report the mean and standard deviation across 5 runs (simulation) and 3 runs (real robot).

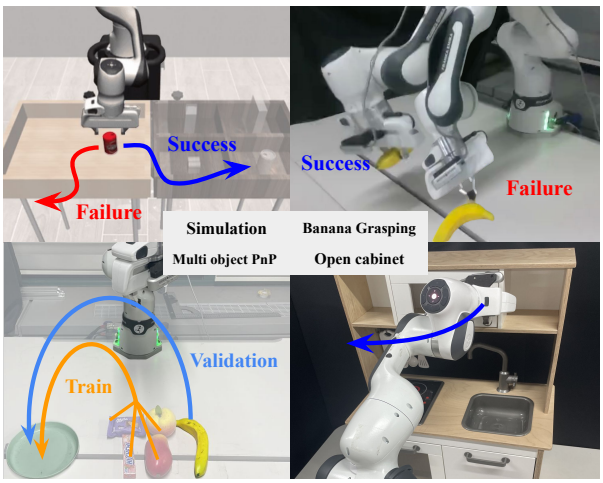

Fig. 3: **Visualization of experiment environments** across simulation and real robot setups, including a single task of grasping a banana and multi-object pick-and-place, and open cabinet task.

In the banana grasping experiment, the train dataset consists of 100 real robot trajectories of grasping a banana, with 60 successful and 40 failure trajectories, where failures are characterized by missed grasping points (*e.g.,* attempting to grasp at unsuitable positions or with improper gripper orientations). The validation set consists of 10 successful trajectories. By taking a curation budget that matches the actual number of successful trajectories in the train data, QoQ achieves substantially high curation accuracy across both domains, outperforming the best baseline by 31.6% in simulation and 16.3% in real robot experiment in the banana grasping experiment as shown in Table I.

We also assess whether these curated datasets translate to improved policy performance. We fine-tune policies using datasets curated by QoQ and baseline methods, then evaluate their success rates during deployment in both simulation and real robot environments. As shown in Figure 2, policies trained with QoQ-curated data achieve a 99.2% success rate in simulation, significantly outperforming Flow Retrieval, the best baseline, which reaches 76.0%. In real robot experiments, QoQ-trained policies achieve 86.7% success compared to 56.7% of the best baseline, Behavior Retrieval.

Method	Simulation	Real (Banana grasping)
QoQ (Ours)	**99.4** $\pm$ **0.3**	**83.6** $\pm$ **0.8**
Behavior Retrieval	67.8 $\pm$ 0.7	67.3 $\pm$ 0.7
Flow Retrieval	56.9 $\pm$ 0.2	57.7 $\pm$ 0.1
All Data	55.4	58.7

TABLE I: **Curation accuracy (%).** Percentage of state-action pairs from successful trajectories among all pairs in the curated dataset.

Multi object curation In this experiment, we aim to identify helpful trajectories for the pick-and-place of a banana, from a training dataset composed of the pick-and-place of different objects. Specifically, the training dataset includes 80 trajectories, with 20 each for a peach, mango, snack, and gum. The validation set consists of 10 successful banana pick-and-place trajectories, and is used to identify the most helpful trajectories for this held-out object. Unlike the single-task curation experiment described above, this train dataset does not contain explicitly failed trajectories. Rather, we aim to find trajectories that contain helpful information for the pick-and-place of a banana from various tasks. By constructing the curated set that takes the top 50% trajectories with the highest QoQ scores, the downstream policy success rate greatly improves over policy trained in all data, reaching 93.3% as shown in Figure 2. Meanwhile, Behavior Retrieval completely fails to capture relevant information to help pick-and-place a banana, resulting in the lowest success rate of 20%[3]. We suspect state-action pair representation from VAEs in Behavior Retrieval is distracted by multiple different objects. Flow Retrieval, meanwhile, performs relatively better by focusing on robot motion captured by optical flow, but still lags behind QoQ-curated policy. In Appendix VI-A, we also show that QoQ's data selection is most consistent across different seeds compared to baselines, which clearly

[3]Note that it is reasonable to achieve a lower success rate than using all data due to the use of fewer trajectories from curation.

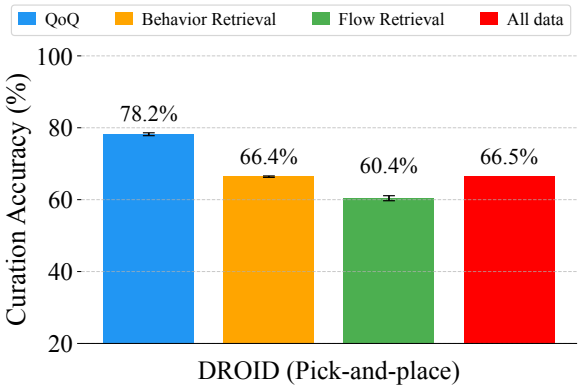

Fig. 4: **Droid dataset curation accuracy (%).** Compared to baselines, QoQ maintains high curation accuracy in DROID dataset, which consists of different domains and object locations.

shows that QoQ well captures relevant information for data curation.

C. How Robust Is QoQ to In-the-wild Robot Data under Diverse Domains?

We evaluate the ability of QoQ to curate high-quality in-the-wild robot data. Specifically, we construct a training dataset using DROID [3]. We sample 200 trajectories of pick-and-place of "pen/pencil" tasks, which involve 133 successful trajectories and 67 failed trajectories. For the validation set, we use 20 successful trajectories in the same tasks. Both successful trajectories and failed trajectories are very challenging to distinguish, as they differ in domains and behavior by varying environments, object location, and camera viewpoints. For this experiment, we train the GR00T N1 model for 50k steps without LoRA to account for the dataset's heterogeneity. To compare different algorithms in this setup, we curate the actual number of successful trajectories and compare the curation accuracy. Figure 4 displays the accuracy of the curation in QoQ, Behavior Retrieval, Flow Retrieval. Our results demonstrate that QoQ shows the highest curation accuracy in the presence of multiple domains, whereas Behavior Retrieval, Flow Retrieval suffer from training VAE encoders from heterogeneous visual input and diverse behaviors.

D. Can QoQ Use Policy Rollout as Validation Set?

Instead of assuming a separate validation set in addition to the train data, we experiment with using trajectories acquired by policy rollout as a validation set when curating train data. For example, a policy trained on an uncurated dataset can be used to collect a validation set for data curation. However, policy rollouts often involve failures due to poor initial performance, while we define validation to contain only desirable behavior. To make use of the failed trajectories as a validation set, we use the negative value of QoQ score after calculating the score using the failed validation trajectories. This intuitively makes sense because for a specific trajectory in the train dataset, if QoQ score is high when using failed trajectories as a validation set, it implies that the trajectory encourages such failed behavior, so it should be discouraged.

We calculate QoQ score the same as before for successful policy rollout and subtract the QoQ score calculated from failed trajectories. To balance between two scores, we weight each score by the size of the validation set used, as it can account for the uncertainty of scores coming from different sizes of validation sets. To verify this setup, we collect a training dataset on the "Open cabinet" task, including 100 successful and 50 failure trajectories. As shown in Figure 3, this task involves grasping and pulling the cabinet door handle to open it. We also vary the cabinet location to 5 different positions to maximize task difficulty. After training our policy on the training dataset, we conducted 20 rollouts from the policy and acquired 5 successful and 15 failure trajectories.

From the training dataset, we take 100 trajectories where such weighted QoQ scores are the highest to construct a curated dataset. Then, we fine-tune policy using this curated set and evaluate policy success rate in real robot deployment. As shown in Figure 5, the policy trained using the curated dataset achieves a higher policy success rate compared to the policy trained on all data. This shows that QoQ can also improve the performance of the pretrained policy without assuming initial availability for the validation set.

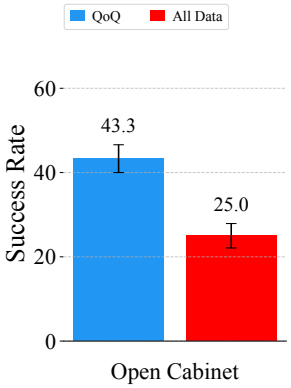

Fig. 5: **Open cabinet policy success rate.**

E. How Does Each Component in QoQ Impact Performance?

In this section, we present the ablation study of QoQ, analyzing the contribution of each component to the overall performance. All experiments are conducted on the banana grasping experiment task using real robots.

Maximum influence scoring. As discussed in Step 1 of Section IV-B, QoQ computes scores by taking the maximum gradient product across validation samples, instead of averaging gradients over all validation samples. To demonstrate the effectiveness of this approach, we compare both curation accuracy and the success rate of policies trained on datasets curated using our maximum influence scoring against those curated by an alternative approach that averages the gradient products. As shown in Figure 6, our method achieves higher accuracy and success rates. This improvement arises from its ability to ignore irrelevant validation samples when scoring each training sample.

Trajectory-wise curation. Similarly, we compare the success rates between trajectory-wise curation (as discussed in Step 2 of Section IV-B) and state-action-wise curation.

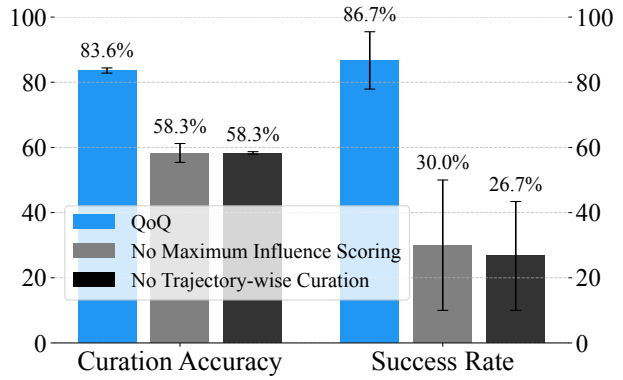

Fig. 6: **Ablation study of QoQ components** in banana grasping experiment. Removing maximum influence scoring and trajectory-wise curation reduces curation accuracy and policy success rate. Error bars show mean±stderr over 3 runs.

Figure 6 shows that trajectory-wise curation results in higher success rates. This gain is due to its ability to mitigate distributional bias in the curated dataset, which could otherwise degrade the performance of the BC policy during training.

Gradient computation layer We compare curation accuracy when computing the QoQ score from different network layers of the GROOT N1 model [5]. Table II shows that applying QoQ to only a subset of network modules yields results that are consistent with those obtained from the full parameters. This suggests that effective influence estimation does not require computing gradients across all layers. As a result, QoQ is particularly well-suited for scaling to modern VLAs [4]–[6], which involve billions of parameters.

Layer Part	Backbone	Action Head	All
Curation Accuracy	82.7 ± 0.6	**83.6** ± **1.3**	82.1 ± 1.2

TABLE II: **Curation accuracy across different influence computation layer parts** in banana grasping experiment. Backbone denotes the VLM and vision encoder part used in GROOT N1 model.

Curation budget We present experimental results on how varying the curation budget affects both curation accuracy and downstream policy performance. When the number of curated trajectories is set below or near the proportion of successful trajectories in the training dataset, we find that curated policies consistently achieve substantially higher task success rates compared to policies trained without curation. Across a wide range of curation budgets, our method also outperforms the strongest baseline, Behavior Retrieval, demonstrating its effectiveness (Table III). However, the curation budget still has a non-negligible impact on final policy performance, as smaller budgets reduce data coverage. Thus, in practice, we recommend exploring multiple curated set sizes to maximize policy performance.

Curation Budget	10	20	40	60	All data	Best baseline
Success Rate (%)	36.7 ± 17.6	63.3 ± 17.6	60.0 ± 10.0	**86.7** ± 8.9	36.7 ± 27.3	56.7 ± 17.6

TABLE III: **Success rates across curation budgets.** QoQ consistently achieves higher success rates across a wide range of budgets compared to baselines. Best baseline refers to the Behavior Retrieval method.

VI. CONCLUSIONS

In this work, we propose QoQ, a method that curates robotic datasets based on direct performance contribution to the learned policy. We define a quality scoring mechanism for state-action pairs derived from influence functions. To enhance curation effectiveness and policy performance, we introduce two key components: (1) maximum influence scoring and (2) trajectory-wise curation. Our experiments demonstrate that QoQ effectively identifies high-quality robotic trajectories, resulting in significantly improved policy success rates compared to baseline curation algorithms across both simulation and real-world robot experiments. Also, QoQ reliably detects success and failed trajectories and curates helpful trajectories from the in-the-wild DROID datasets. We believe that our scoring method offers a promising approach for data-driven robot learning, enabling more efficient use of demonstration data to achieve high-performing robotic policies.

Limitations and Future Work Our approach still has several limitations that suggest promising directions for future work. While trajectory-level curation ensures broad coverage, it cannot selectively use high-quality segments within a trajectory, motivating finer-grained sub-trajectory curation. Influence function computation remains costly and approximate, despite layer restrictions and methods like TracIn [23] or OPORP [25], calling for more accurate yet efficient estimators. Moreover, our setup assumes shared embodiments between training and validation, whereas extending to cross-embodiment scenarios (e.g., Open X-Embodiment [2]) would broaden applicability. Although we focus on behavioral cloning, the method can generalize to other policy objectives, such as offline RL.

APPENDIX

A. Data Selection Consistency Experiment

In this section, we compare the data selection consistency between QoQ and baselines across from multi-object curation experiment across three different seeds. Our hypothesis is that more consistent rankings across seeds indicate that the algorithm provides more reliable trajectory scores. Specifically, we use Kendall's W coefficient [35] to compare the correlation of ranking.

Metric	QoQ	Behavior Retrieval	Flow Retrieval
Kendall's W	0.7713	0.3287	0.7026

TABLE IV: Kendall's W values across different methods. Higher indicates more consistent trajectory rankings.

Table IV shows that QoQ shows the most consistent ranking across seeds, followed by Flow Retrieval and Behavior Retrieval algorithms. This also matches the success rates of downstream policies shown in Section V-B, demonstrating that QoQ well captures relevant information for data curation.

ACKNOWLEDGMENT

The authors would like to thank Changyeon Kim, Juyong Lee and Dongjun Lee for providing helpful comments for improving the work. This work was supported by Institute for Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (RS-2019-II190075, Artificial Intelligence Graduate School Program(KAIST)), the Korea government(MSIT) (No. RS-202400509279, Global AI Frontier Lab), and Institute of Information & Communications Technology Planning & Evaluation(IITP) grant (RS-2025-02304967, AI Star Fellowship(KAIST)) funded by the Korea government(MSIT).

REFERENCES

[1] H. R. Walke, K. Black, T. Z. Zhao, Q. Vuong, C. Zheng, P. Hansen-Estruch, A. W. He, V. Myers, M. J. Kim, M. Du, *et al.*, "Bridgedata v2: A dataset for robot learning at scale," in *Conference on Robot Learning*, 2023.

[2] A. O'Neill, A. Rehman, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, A. Jain, *et al.*, "Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration," in *IEEE International Conference on Robotics and Automation*, 2024.

[3] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis, *et al.*, "Droid: A large-scale in-the-wild robot manipulation dataset," *arXiv preprint arXiv:2403.12945*, 2024.

[4] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, *et al.*, "π0: A vision-language-action flow model for general robot control," *arXiv preprint arXiv:2410.24164*, 2024.

[5] J. Bjorck, F. Castañeda, N. Cherniadev, X. Da, R. Ding, L. Fan, Y. Fang, D. Fox, F. Hu, S. Huang, *et al.*, "Gr00t n1: An open foundation model for generalist humanoid robots," *arXiv preprint arXiv:2503.14734*, 2025.

[6] M. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, Q. Vuong, T. Kollar, B. Burchfiel, R. Tedrake, D. Sadigh, S. Levine, P. Liang, and C. Finn, "Openvla: An open-source vision-language-action model," *arXiv preprint arXiv:2406.09246*, 2024.

[7] O. M. Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, T. Kreiman, C. Xu, *et al.*, "Octo: An open-source generalist robot policy," *arXiv preprint arXiv:2405.12213*, 2024.

[8] S. Belkhale, Y. Cui, and D. Sadigh, "Data quality in imitation learning," in *Advances in Neural Information Processing Systems*, 2023.

[9] 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*, 2022.

[10] 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.

[11] L.-H. Lin, Y. Cui, A. Xie, T. Hua, and D. Sadigh, "Flowretrieval: Flow-guided data retrieval for few-shot imitation learning," in *Conference on Robot Learning*, 2024.

[12] J. Hejna, S. Mirchandani, A. Balakrishna, A. Xie, A. Wahid, J. Tompson, P. Sanketi, D. Shah, C. Devin, and D. Sadigh, "Robot data curation with mutual information estimators," *arXiv preprint arXiv:2502.08623*, 2025.

[13] P. W. Koh and P. Liang, "Understanding black-box predictions via influence functions," in *International Conference on Machine Learning*, 2017.

[14] A. Ghorbani and J. Zou, "Data shapley: Equitable valuation of data for machine learning," in *International Conference on Machine Learning*, 2019.

[15] R. Jia, D. Dao, B. Wang, F. A. Hubis, N. M. Gurel, B. Li, C. Zhang, C. J. Spanos, and D. Song, "Efficient task-specific data valuation for nearest neighbor algorithms," *arXiv preprint arXiv:1908.08619*, 2019.

[16] Y. Kwon and J. Zou, "Beta shapley: a unified and noise-reduced data valuation framework for machine learning," *arXiv preprint arXiv:2110.14049*, 2021.

[17] S. Nasiriany, T. Gao, A. Mandlekar, and Y. Zhu, "Learning and retrieval from prior data for skill-based imitation learning," in *Conference on Robot Learning*, 2022.

[18] M. Memmel, J. Berg, B. Chen, A. Gupta, and J. Francis, "STRAP: Robot sub-trajectory retrieval for augmented policy learning," in *International Conference on Learning Representations*, 2025.

[19] A. S. Chen, A. M. Lessing, Y. Liu, and C. Finn, "Curating demonstrations using online experience," *arXiv preprint arXiv:2503.03707*, 2025.

[20] 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.

[21] C. Agia, R. Sinha, J. Yang, R. Antonova, M. Pavone, H. Nishimura, M. Itkina, and J. Bohg, "Cupid: Curating data your robot loves with influence functions," *arXiv preprint arXiv:2506.19121*, 2025.

[22] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," in *Advances in Neural Information Processing Systems*, 1988.

[23] G. Pruthi, F. Liu, S. Kale, and M. Sundararajan, "Estimating training data influence by tracing gradient descent," in *Advances in Neural Information Processing Systems*, 2020.

[24] M. Laskey, J. Lee, R. Fox, A. Dragan, and K. Goldberg, "Dart: Noise injection for robust imitation learning," in *Conference on Robot Learning*, 2017.

[25] P. Li and X. Li, "Oporp: One permutation+ one random projection," in *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023.

[26] Y. Kwon, E. Wu, K. Wu, and J. Zou, "Datainf: Efficiently estimating data influence in loRA-tuned LLMs and diffusion models," in *International Conference on Learning Representations*, 2024.

[27] T. Min, H. Lee, H. Ryu, Y. Kwon, and K. Lee, "Understanding impact of human feedback via influence functions," *arXiv preprint arXiv:2501.05790*, 2025.

[28] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2014.

[29] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[31] E. J. Hu, yelong shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "LoRA: Low-rank adaptation of large language models," in *International Conference on Learning Representations*, 2022.

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

[33] K. Georgiev, J. Vendrow, H. Salman, S. M. Park, and A. Madry, "The journey, not the destination: How data guides diffusion models," *arXiv preprint arXiv:2312.06205*, 2023.

[34] D. McAllister, S. Ge, B. Yi, C. M. Kim, E. Weber, H. Choi, H. Feng, and A. Kanazawa, "Flow matching policy gradients," *arXiv preprint arXiv:2507.21053*, 2025.

[35] H. Abdi, "The kendall rank correlation coefficient," *Encyclopedia of measurement and statistics*, vol. 2, pp. 508–510, 2007.