Robotics: Science and Systems 2025
Los Angeles, CA, USA, June 21–25, 2025

Robot Data Curation with Mutual Information Estimators

Joey Hejna[1,2,*], Suvir Mirchandani[2], Ashwin Balakrishna[1], Annie Xie[1], Ayzaan Wahid[1], Jonathan Tompson[1], Pannag Sanketi[1], Dhruv Shah[1], Coline Devin[1,†], Dorsa Sadigh[1,†]
[1]Google DeepMind Robotics, [2]Stanford University

***Abstract*—The performance of imitation learning policies often hinges on the datasets with which they are trained. Consequently, investment in data collection for robotics has grown across both industrial and academic labs. However, despite the marked increase in the quantity of demonstrations collected, little work has sought to assess the quality of said data despite mounting evidence of its importance in other areas such as vision and language. In this work, we take a critical step towards addressing the data quality in robotics. Given a dataset of demonstrations, we aim to estimate the relative quality of individual demonstrations in terms of both action diversity and predictability. To do so, we estimate the average contribution of a trajectory towards the mutual information between states and actions in the entire dataset, which captures both the entropy of the marginal action distribution and the state-conditioned action entropy. Though commonly used mutual information estimators require vast amounts of data often beyond the scale available in robotics, we introduce a novel technique based on k-nearest neighbor estimates of mutual information on top of simple VAE embeddings of states and actions. Empirically, we demonstrate that our approach is able to partition demonstration datasets by quality according to human expert scores across a diverse set of benchmarks spanning simulation and real world environments. Moreover, training policies based on data filtered by our method leads to a 5-10% improvement in RoboMimic and better performance on real ALOHA and Franka setups.**

I. Introduction

Supervised learning via maximum likelihood estimation, i.e., attempting to reproduce the training distribution, underpins several recent advancements in deep learning. Due to the broad availability of high-quality data on the internet, models in vision [62] and language [61] have continued to improve through supervised learning on larger and larger datasets [33, 74]. The observed trend of more data leading to more performance has inspired parts of robot learning community, spurring increased investment in data collection across both academia and industry [57, 24, 34, 69] in hopes of training better imitation learning policies, often with similar maximum likelihood objectives [55, 9]. However, MLE-based approaches benefit most from high-quality data, and as we have seen in vision and language, not all data is equal [72]. In other words, we should expect the performance of a large behavior cloning policies to mirror the quality of the collected data, and we may not be gathering the most optimal data. As an example, the recent DROID dataset [34] contains 76K demonstrations collected across 13 institutions. Despite being one of the largest and most diverse datasets in robotics, the DROID dataset was significantly down-weighted when training OpenVLA [36] as it was found to hurt performance rather than help. While many hypotheses surrounding this exist, e.g., insufficient operator supervision leading to an excessively broad data distribution, one conclusion we can draw is that we should pay more attention to data *quality*, not just quantity. This is particularly important in robotics, as every demonstration requires labor, time, and capital.

Even though data quality is critical to the performance of imitation learning algorithms, little work has sought to measure it [6] and unfortunately, techniques used for data curation in vision and language do not transfer well to robotics. For example, n-gram classifiers have been extremely effective for web text [2] but are unable to handle high-dimensional continuous states and actions. Pre-trained models have been used to curate vision datasets [64], but are incapable of reasoning about actions. In contrast, we believe metrics for imitation learning should be able to measure the relative predictability of the state-action distribution directly, which affects how well a policy is able to fit the expert [6]. In this work we explore how this desiderata can be captured by the mutual information between states and actions.

Mutual information, or the bits of information learned about one random variable by observing another, precisely measures the difference between the marginal entropy of one variable and the conditional entropy of another. In the context of states and actions, this means that high mutual information encourages a large diversity of actions (action entropy) but a predictable action distribution (low entropy of actions given a particular state). We thus propose using mutual information as a desirable metric for measuring data quality in imitation learning.

Unfortunately, estimating mutual information is particularly hard especially with low amounts of data. Common estimation techniques like InfoNCE [56] used for models such as CLIP [62] often require millions of data points from the same distribution. In robotics, we often do not have access to data at a similar scale due to the difficulty and cost of collection. Moreover, even if we assume access to more data, existing large robot datasets such as OpenX Embodiment [57] have sporadic support across a few highly varied environments likely containing little to no overlap with new data collected

* Work done as a Student Researcher at Google DeepMind
† Equal advising.
Videos and code at https://jhejna.github.io/demonstration-info.

across different labs, platforms, and tasks. To address this problem, we introduce Demonstration Information Estimation or DemInf for short. We design DemInf to work across both low- and high-data regimes in robotics. To do so, DemInf first learns a structured low-dimensional representation of the state and action space using variational autoencoders. Then, DemInf leverages mutual information estimators based on k-nearest neighbors to estimate the quality of state and action chunk pairs. Critically we found these non-parametric estimators to be more stable with datasets of 50-300 demonstrations commonly used in robotics. Finally, we average mutual information estimates across time to identify the highest and lowest quality demonstrations.

When applying DemInf to a number of different robot platforms and environments, we find that it is able to consistently partition high- and low-quality data as scored by expert human annotators, outperforming both contemporary baselines and alternative mutual information estimators like InfoNCE [56]. Furthermore, using DemInf to subsample demonstration data, we are able to attain higher performing imitation learning policies across the RoboMimic [50] benchmark and real ALOHA policies on RoboCrowd [52].

II. Related Work

As deep learning models have continued to scale, data quality estimation has become an area of increasing interest. Here we review works most relevant to our approach.

Data Quality in Vision and Language. Data quality has most often been studied in the vision and language domains, where modern training pipelines often include multiple steps of quality estimation and de-duplication [2, 16, 58]. For text data, this often consists of simple n-gram classifiers, or metadata filtering, which have been shown to have a large impact on performance [72]. Other more advanced techniques use unsupervised clustering [68, 67, 1, 8], most commonly for de-duplication and balancing across clusters. Though these methods improve the diversity of large datasets, they work mostly at scale and independent of label (or in our case action) quality. Methods in group mixing have been shown to increase learning efficiency by improving the dataset's distributional properties, but do so only at the coarse group level [12, 11, 71, 22]. This is problematic in robotics as we often are actively collecting data, and want to assess trajectories individually. Most related to our approach are techniques based on pre-trained models such as CLIP [23]. Though these methods do not explicitly make the connection, contrastive models precisely estimate a bound on mutual information [47, 56]. Due to the data requirements of contrastive learning, such techniques rely on large pre-trained models, e.g., CLIP [62], as priors for curation [23]. Unfortunately, such priors are useless for estimating the mutual information between states and actions. Moreover, training a similar contrastive model from scratch requires hundreds of millions of training examples [72]. This is rather unrealistic for the current behavior cloning paradigm where even the largest datasets have less than 100,000 demonstrations [34], and we do not have access to strong pre-trained action priors.

Data Quality in Robotics. Orthogonal to us, several works have focused on increasing the size of robotics datasets, through the development of tools [73, 15], human teleoperation [34, 57, 32, 20, 65, 49, 59, 7, 54], or automatic data augmentation [28, 51, 5, 48], with the aim of training large-scale robot policies [9, 77, 42, 55]. Through this process, data quality in the context of robot learning has come into question, but largely through the lens of inter-demonstration compositional generalization to new objects or scenes [10, 70, 26, 43]. Unlike our work, such approaches do not consider intra-demonstration transition quality, e.g., how good the action labels are which can ultimately determine the performance of imitation learning methods. Other works that consider action quality do so at an extremely coarse level. ReMix [30] learns group weights over large robot datasets using robust optimization. Such dataset mixing approaches require datasets to be partitioned into groups a priori, and are thus unable to determine the quality of individual demonstrations. Retrieval methods [53, 21, 44] use a target dataset to retrieve state-action pairs from unstructured data, but do not explicitly measure data quality, only similarity. Perhaps most related to our work, Kuhar et al. [41] directly estimate the quality of individual demonstrations using a latent space from temporal contrastive learning. However, to actually produce quality estimates they assume access to a dataset of human quality labels. Moreover, the choice of temporal contrastive learning means that the learnability of actions is not explicitly considered. DemInf on the other hand is completely unsupervised, and thus can be applied to broad amounts of robot data without any hand annotation.

Mutual Information Estimation. Mutual information estimation has been a long studied problem in both statistics and deep learning [60]. Direct mutual information objectives like InfoNCE [56] are often used for representation learning in vision [13] and language [75]. Other works have used the dual formulation [3]. Unfortunately, these parametric methods techniques often require on the order of a million samples for accurate estimation, but having access to this scale of data is rather uncommon when trying to measure data quality for imitation learning in a specific environment. Instead, DemInf uses non-parametric estimators based k-nearest-neighbors [39, 66], specifically the KSG estimator [40, 27]. Prior works have used k-nn estimators in unsupervised RL, but do so for maximizing state entropy [45, 35], not mutual information or data quality.

III. Preliminaries

A. Imitation Learning

Broadly, the objective of imitation learning is to learn a policy $\pi_\theta : \mathcal{S} \to \mathcal{A}$ parameterized by θ that is able to effectively reproduce the behavior of an expert π_E within an environment with state space $\mathcal{S}$, action space $\mathcal{A}$ and horizon T. Typically, we measure the similarity between the policy and expert using a divergence between their state visitation distributions:

$$\min_\theta D_{\text{KL}}\left(\rho_{\pi_\theta} || \rho_{\pi_E}\right), \tag{1}$$

where $\rho_\pi^t(s)$ is the probability that the policy visits state s at time t and $\rho_\pi(s) = \frac{1}{T}\sum_{t=1}^{T}\rho_\pi^t(s)$ is the average visitation across time. In essence, the above objective states that we want the learned policy to visit the same states as the expert. However, optimizing Eq. (1) is challenging as it requires sampling from the learned policy, which can usually only be done accurately by interacting with the environment.

Instead, the most common approach to imitation learning, Behavior Cloning (BC), reduces the problem to standard supervised learning [63]. Using the *opposite* direction of the KL divergence with respect to Eq. (1), π_θ can be learned purely offline.

$$\mathcal{L}_{\text{BC}}(\theta) = \mathbb{E}_{s\sim\rho_{\pi_E}}\left[D_{\text{KL}}(\pi_E(\cdot|s)||\pi_\theta(\cdot|s))\right] \tag{2}$$

In this case, we only need samples from π_E typically in the form of a dataset of N demonstrations $\mathcal{D}_N = \{\tau_1, \ldots, \tau_N\}$, where each demonstration $\tau_i = (s_1, a_1, s_2, a_2, \ldots, s_{T_i}, a_{T_i})$ of length T_i is a valid sequence through the state-action space according to the dynamics.

Demonstrations are assumed to be sampled from an absolute expert π_E—however, this assumption in practice is unrealistic. As an example, though we might only care about completing a "task" when learning robot policies, there are often several strategies of doing so, and even when using the same strategy, different demonstrators may be subtly different in how they complete the task, which ends up affecting our empirical estimate of the expert. In robot demonstration curation, we ask how we can better define the empirical expert.

B. Demonstration Curation

While theoretical analyses fix the expert distribution, in practice it is empirically defined by the users and practitioners who collect data. In turn, choices made during data collection can affect the performance of a policy trained with behavior cloning. For example, a novice data collector may produce less predictable actions than an experienced one, and pooling together the data from multiple demonstrators may lead to a more complex action distribution. Moreover, choices made within individual demonstrations τ, such as using differing strategies or varied approaches to complete a task, might make learning from the overall dataset $\mathcal{D}_N$ more difficult. Thus, the problem of demonstration curation in imitation learning is concerned with how we can shape the expert policy distribution ρ_{π_E} such that we can attain the highest performance at a given task. Mathematically, we do so by adjusting the empirical expert distribution, $\hat{\rho}_{\pi_E}(s) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{T}\sum_{t=1}^{T}\mathbb{1}(s = \tau_{i,t})$ of the dataset, where $\tau_{i,t}$ is the tth state of the ith demonstration.

We consider the general problem of shaping the empirical expert distribution $\hat{\rho}_{\pi_E}(s)$ tabula rasa at the demonstration level, assuming all demonstrations are successful at completing the desired task. Specifically, our goal is to determine a score function $S(\tau)$ in a purely offline fashion that is able to predict the *quality* of demonstrations, where quality is determined by the performance of a policy trained with behavior cloning on the score-filtered demonstration dataset

$$\mathcal{D}_N(\kappa, S) = \{\tau_i \mid S(\tau_i) > \kappa, \forall i = 1, \ldots, n\} \tag{3}$$

for some quality threshold value κ. This is a more difficult problem than considered in prior work. Data mixing approaches Hejna et al. [30] only modify $\hat{\rho}_{\pi_E}(s)$ at the mixture level, i.e. adjusting coarse coefficients α over groups of demonstrations. Instead, considering data curation at the individual demonstration allows us to have a fine-grained understanding of what strategies and expert distributions lead to the best performing policy downstream. Works in interactive data curation necessitate both online access to the environment and expensive oracle feedback [31, 18] for curation. Our setting is purely offline and unsupervised, allowing methods we develop to be applied to virtually any robotics dataset available. However, given we have no explicit signal from the environment in the demonstration curation setting, we aim to define S according to unsupervised objectives, namely mutual information. In the next section, we discuss why mutual information between states and actions can be a valuable scoring function for behavior cloning.

IV. Mutual Information as a Quality Metric

Mutual information captures the bits of knowledge one gains about one random variable by observing another, in essence measuring predictability. In BC, we want to train a policy π_θ to predict the action a from the state s. Thus the mutual information between states and actions is a rather natural choice for a quality metric. In this section we interpret the following factorization of mutual information in the context of robot data curation:

$$I(S; A) = H(A) - H(A \mid S) \tag{4}$$

where S and A represent random variables for the state and action. First, we will discuss why minimizing the conditional action entropy allows for more accurate policies. Second, we discuss why maximizing the marginal action entropy is required to learn a good policy.

A. Minimizing Conditional Action Entropy

Our overall objective is to align the distribution of the learned policy with that of expert data (Eq. (1)). Following Theorem 4.1 of Belkhale et al. [6], we can bound the distribution matching objective from Eq. (1) using the log-sum inequality in terms of the divergence between the learned policy and expert policy at each time step:

$$D_{\text{KL}}\left(\rho_{\pi_\theta}||\rho_{\pi_E}\right) \leq \frac{1}{T}\sum_{t=1}^{T}(T-t)\mathbb{E}_{s\sim\rho_{\pi_\theta}^t}\left[D_{\text{KL}}\left(\pi_\theta(\cdot|s)||\pi_E(\cdot|s)\right)\right].$$

Intuitively, if we can keep the policies close enough to each other at every state, then we should be able to better reproduce the desired state distribution. Below, we use this fact to argue why low conditional action entropy $H(A \mid S)$ (term 2 in Eq. (4)) leads to better BC performance [6].

Ease of Fit. Lower entropy distributions are generally simpler, possibly making them easier to match. For example, an action distribution that can only take on a single value has zero conditional entropy. Note that BC (Eq. (2)) optimizes the

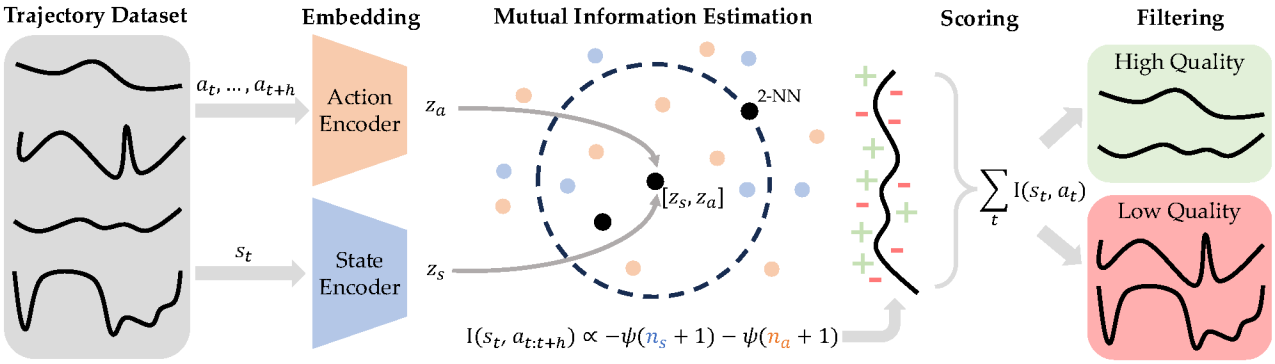

Fig. 1. A graphical depiction of the DemInf method. First, we begin by learning VAEs for states and action chunks to produce latent representations z_a and z_s. Using these latent representations, we apply the KSG k-nearest-neighbor based mutual information estimator. Finally, we filter demonstrations based on their estimated mutual information.

opposite direction of the KL-divergence with respect to the above abound. The forward and reverse KL-divergences are only equal when π_E and π_θ are the same. This is more likely to happen for simple distributions, allowing us to make progress towards the true state matching objective in Eq. (1).

Multimodality. Lower entropy distributions often have fewer modes or peaks. Given the forward and reverse KL-divergences have different behaviors around modes, e.g., mode-seeking versus mode-covering, they are more likely to exhibit similar behaviors on unimodal datasets.

Privileged Information. It can be difficult for a policy to fit demonstrations when the data collector has access to information unavailable to the policy. For example, a data collector may have extra sensory information–such as direct line-of-sight to observe objects that are occluded in the robot's camera views. The resulting actions might only be predictable when given access to the unobserved variable Z. Mathematically, we can bound the mutual information between the unobserved Z and actions A by $H(A|S) \geq I(A; Z|S)$ [17]. Thus, by minimizing the entropy of the action distribution we ensure that unobserved factors have a smaller effect on the data.

B. Maximizing Marginal Action Entropy

In addition to minimizing conditional action entropy, mutual information encourages high entropy in the marginal action distribution $H(A)$ (the first term of Eq. (4)). While this might be puzzling at first, it has an important regularizing affect. Without considering the marginal entropy of actions $H(A)$, the conditional entropy $H(A|S)$ could be trivially minimized by distributions that have constant actions, e.g. taking the same action regardless of the state. Having a high marginal action entropy avoids this pitfall, forcing the learned policy to pay attention to the state when making predictions, which is desirable for closed-loop control.

We find mutual information to be a useful metric for data quality. However, practitioners in robotics have often considered simpler metrics as heuristics for data quality. For example, initial state diversity is often chosen as a convenient proxy for enabling generalization. Mutual information $I(S; A)$ can equivalently be decomposed as $H(S) - H(S|A)$ to uncover state entropy. However, the latter term is less interpretable in the context of imitation learning. As we will later average mutual information estimates across entire trajectories, DemInf will not explicitly optimize for initial state diversity, but instead the underlying predictability of the data.

V. Method

Though mutual information is perhaps a natural metric for data curation, it can be practically difficult to estimate [19]. In this section we propose the Demonstration Information Estimation (DemInf) method for computationally estimating mutual information for demonstration data. Though mutual information is usually considered at the distribution or dataset level, we are interested in scoring individual demonstrations for data curation. Thus, we measure the contribution of individual episodes to the overall mutual information of the dataset. Fortunately, this can easily be done as the majority of of empirical mutual information estimators can be decomposed into an average of sample-wise estimators.

$$\hat{I}(S; A) = \frac{1}{|\mathcal{D}_N|} \sum_{(s_i, a_i) \in \mathcal{D}_N} \hat{I}(s_i, a_i; \mathcal{D}_N) \tag{5}$$

As previously outlined in Section II, there are several possible neural estimators of mutual information which can be applied to high dimensional robotics data. However, the majority of existing methods like InfoNCE [56] and MINE [4] have extremely high sample requirements for effective estimation which are unrealistic for real world BC datasets. To overcome this challenge we propose Demonstration Information Estimation, which uses k-nearest-neighbor (k-NN) estimates of mutual information. Our method involves three steps – representation learning, mutual information estimation, and scoring – which we outline below.

A. Representation Learning

As k-NN estimators of mutual information do not require training a deep neural network, they have been found to be more sample efficient than other estimators. However, they are typically applied to low-dimensional datasets in contrast with robotics datasets which often contain multiple images and sensors. Directly applying k-NN estimators to raw image data may suffer poor performance as distances as become meaningless due to the curse of dimensionality [38]. To

remedy this problem and provide a space suitable for non-parametric estimation we train separate Variational Auto-Encoders (VAEs) [37] to embed both the states and actions into low-dimensional representations.

We denote embedded states as $z_{s,i} = f_s(s_i)$ and embedded actions as $z_{a,i} = f_a(a_i)$. Though other techniques for representation learning exist, we choose to learn VAEs because they enforce an isotropic Gaussian constraint onto the latent distribution $p(z)$. This is particularly desirable for k-NN based mutual information estimators for two reasons. First, enforcing a prior over the latent distribution ensures that distances between embedded states and actions are meaningful – a necessary prerequisite for statistics based on k-NN. Second, k-NN based mutual information estimators are commonly assessed on Gaussian distributed data, where they are known to perform well [19]. When training the VAEs f_s and f_a we try to select the smallest latent dimension that we believe can sufficiently capture the variable.

B. k-NN MI Estimation

Given a latent representation of the states and actions, we can estimate the contribution of an individual state-action pair to the overall mutual information of the dataset using k-NN based estimators. The general intuition behind these estimators is that the probability density function around a sample is proportional to how many other data points are near it, which can be measured with the nearest neighbors. If the density function is high near a data point, then we expect there to be many samples around it and thus have a small k-NN distance. Conversely, if the density function is low we expect a large k-NN distance. Averaging these density estimates allows us to estimate entropy [39], which can be extended to mutual information. In particular, we use the KSG estimator from Kraskov et al. [40], which we outline below.

Let $\rho_{k,i}$ be the k-NN distance of the ith state action pair $[z_s, z_a]$ in the joint space $\mathcal{Z}_\mathcal{S} \times \mathcal{Z}_\mathcal{A}$ defined using the metric

$$||[z_s, z_a] - [z'_s, z'_a]|| = \max\{||z_s - z'_s||_2, ||z_a - z'_a||_2\}.$$

The L2 norm between individual latents follows the Gaussian distribution learned by the VAEs. The infinity norm between the $\mathcal{Z}_\mathcal{S}$ and $\mathcal{Z}_\mathcal{A}$ spaces allows the errors from estimates of S and A to cancel in the final KSG estimator. Then, in the context of Eq. (5), the KSG estimator is given by:

$$\hat{I}(s_i, a_i; \mathcal{D}_N) \propto -\psi(n(z_{s,i}) + 1) - \psi(n(z_{a,i}) + 1)$$

where ψ is the di-gamma function and $n(z_{s,i})$ is

$$n(z_{s,i}) = \sum_{j \neq i} \mathbb{1}\{||z_{s,i} - z_{s,j}||_2 \leq \rho_{k,i}\}$$

or the number of latent states z_s less than or equal to the k-nearest-neighbor distance $\rho_{k,i}$ in $\mathcal{Z}_\mathcal{S} \times \mathcal{Z}_\mathcal{A}$. The same quantity is analogously defined for actions. We omit constant terms that do not affect the relative contribution of different state-action pairs to the mutual information. We refer the reader to Fig. 1 for a pictoral example.

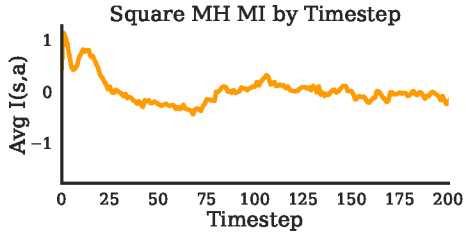

Fig. 2. The average estimated $\hat{I}(s; a)$ per timestep for high quality data ("better" demonstrators) in "Square MH" from RoboMimic [50]. Notice that at the start of the trajectory and after the grasp (75-100 steps), $\hat{I}$ is highest, while it is low during the grasp period (50-75 steps).

As computing k-NN is computationally prohibitive as the dataset size increases, we take a randomized approach. Using a large batch size, we iterate over the dataset multiple times, each time with a distinct shuffling order. We then compute the mutual information contribution $\hat{I}(s, a; B)$ within each batch B for multiple values of k and average.

C. Scoring

Given a set of mutual information estimates, our goal is to determine a scoring function S for each episode τ. Intuitively, we can then define the scoring function for each demonstration as the average contribution of that demonstration τ to the overall mutual information estimator $\hat{I}(S, A)$.

$$S(\tau) = \frac{1}{T} \sum_{t=1}^{T} \hat{I}(s_t, a_t; \mathcal{D}_N)$$

Since we are filtering datasets by the score, we primarily care about the relative ordering of mutual information estimates rather than their absolute values. In practice, we standardize the dataset by first clipping state-action estimates $\hat{I}(s, a)$ to lie between the 1st and 99th percentiles to prevent excessive influence of outliers.

Note that even though we have scores for each state-action pair $\hat{I}(s, a)$, we do not use them to directly filter the data. Such an approach would not only be noisier, but also remove all parts of a task that are inherently harder to predict, but necessary for success. For example, free motion towards an object is likely easy to predict, but the exact time-step at which the gripper should close is hard to predict. We show this in Fig. 2 for the high quality demonstrations in one dataset, where $\hat{I}(s, a)$ is significantly higher at the start during free motion and lowest when grasping the object ($\sim$ 50–75 steps). Filtering data by mutual information at the state-action level thus might drop data for crucial parts of a task that inherently have lower mutual information in favor of easily predictable motion.

Using the score function S, we can subset the dataset to include only demonstrations that contribute positively towards the average mutual information estimate of the dataset.

VI. Experiments

We aim to answer the following questions: (1) How well does DemInf curate robot data? (2) How do different mutual information estimators affect performance? (3) Can data curation via mutual information improve performance on down-

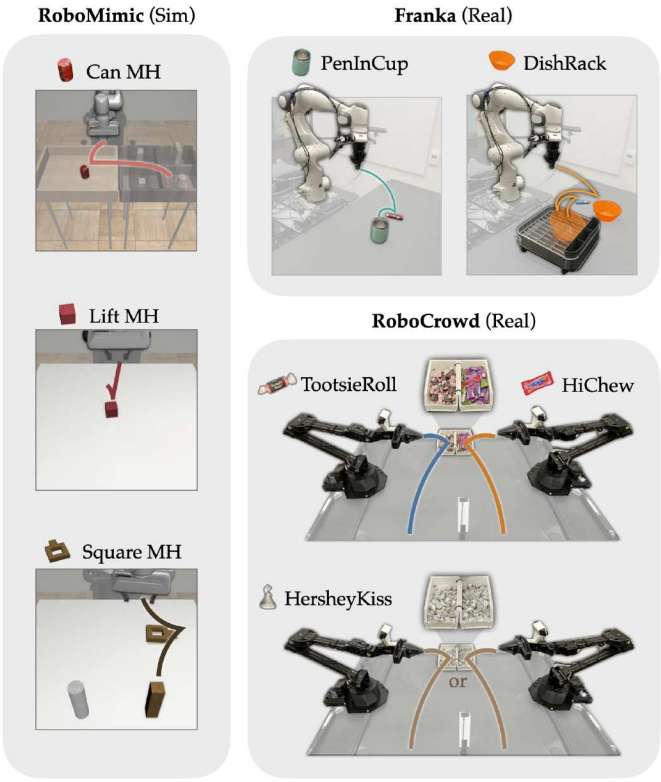

Fig. 3. Visualization of the tasks represented in the datasets we use in this work, including the Can MH, Lift MH, and Square MH datasets from RoboMimic; real-world PenInCup and DishRack datasets collected on a Franka robot; and the real-world TootsieRoll, HiChew, and HersheyKiss datasets from RoboCrowd for the ALOHA robot.

stream policy learning? and (4) What is important to DemInf's performance? Additional results are presented in Appendix A.

A. Experimental Setup

1) Datasets: To assess the performance of different robot demonstration curation techniques, we perform experiments on a broad set of datasets spanning simulated, real single-arm, and real bi-arm robots with varying levels of data quality as depicted in Fig. 3. Notably, we use datasets where human experts have provided quality labels, allowing us to easily assess different demonstration curation metrics:

RoboMimic. The multi-human datasets from the RoboMimic benchmark [50] include 100 demonstrations from each of three robot operators for three tasks in increasing difficulty: "Lift" where the robot simply lifts a cube, "Can" where the robot moves a can from one bin to another, and "Square" where the robot places a nut onto a peg. RoboMimic provides quality labels for each operator, which we use to assign quality scores (with scores of 1, 2, and 3 for the "worse", "okay", and "better" demonstrations respectively). We measure the performance of different data curation methods from both state, in which ground truth object information is provided, as well as third-person images.

Franka. Using the setup from Khazatsky et al. [34] with a Franka Panda robot we collect 60 and 80 demonstrations for each of two tasks, "PenInCup" and "DishRack" respectively. Within each task, we collect 50% expert demonstrations (quality 1) and 50% poor demonstrations (quality 0), where the operator intentionally makes a mistake (e.g. dropping an object, taking a long inefficient path, jerky motion). We use a single third person camera and a wrist camera to train policies and action chunks of size 4.

RoboCrowd. The RoboCrowd benchmark from Mirchandani et al. [52] contains crowdsourced robot data on the bimanual ALOHA [76] platform from real, novice users in a natural environment. Data in RoboCrowd varies widely in quality – many trajectories contain suboptimal data or sequences of "play" data that are irrelevant to the target task. RoboCrowd serves as a suitable platform to study data curation as it has a small number of expert demonstrations for each task and human expert quality labels ranging from 0 to 3 for all crowd-sourced data. Specifically, we use the "HiChew Play," "TootsieRoll Play," and "HersheyKiss Play" datasets which contain both expert demonstrations and crowdsourced demonstrations for candy bin-picking tasks. Every demonstration contains some amount of task-relevant data, with the potential of irrelevant play data in the crowdsourced demonstrations as well. We additionally evaluate on versions of these datasets ("HiChew", "TootsieRoll', "HersheyKiss") where the unstructured play data has been removed, but where demonstrations still contain task-relevant data of varying quality. The HiChew and TootsieRoll datasets contain 40 demonstrations each and the HersheyKiss dataset contains 100 demonstrations, half of which are expert demonstrations. We use the wrist cameras, overhead camera, and action chunks of size 10 for data curation.

2) Baselines: We compare against a number of different data quality estimators from prior work in addition to a number of alternative mutual information estimators, which we label with "(MI)".

Uncertainty. Following prior works in active learning for imitation learning [18, 31], we select data based on the uncertainty of an ensemble of 5 policies. Note that while this metric makes sense for active learning, it does not necessarily make sense in the offline setting, and in some ways may be inversely correlated with quality if the ensemble converges better on high quality data.

Compatibility. Following Gandhi et al. [25], we use a measure of demonstration "compatibility" to score data. Namely, a demonstration is compatible with respect to a policy if it has either high "novelty" as measured by the prediction variance of an ensemble, or low novelty and high likelihood as measured by the average loss. Though this method was originally designed to be used in the online setting, we adopt it to the offline setting by training a policy on all data, then estimating the "compatibility" for each demonstration with respect to the overall policy.

VIP. Value Implicit Pre-training [46] is an action-free method that leverages the dual formulation of the goal-conditioned RL problem to learn a "universal" value function. We use VIP to estimate data quality by considering the total predicted reward over a demonstration.

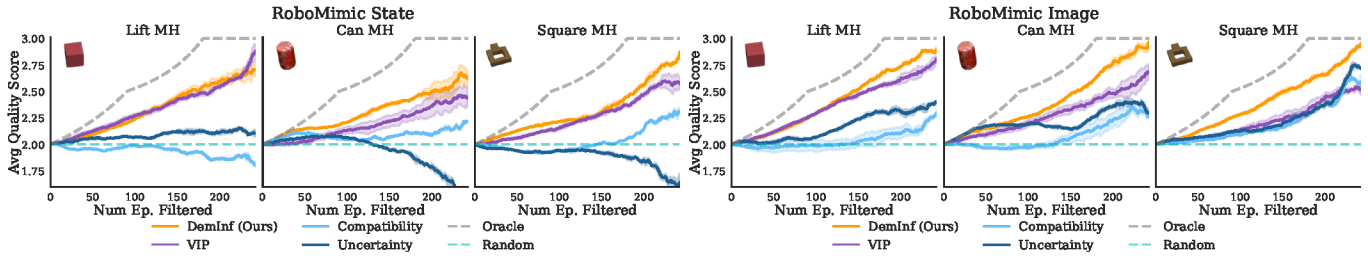

Fig. 4. Average quality of demonstrations remaining in datasets after filtering with different choices of S on the Lift, Can, and Square Multi-Human (Mh) datasets from the Robomimic benchmark with states (Left) and images (right). Results are shown as an average of 3 seeds.

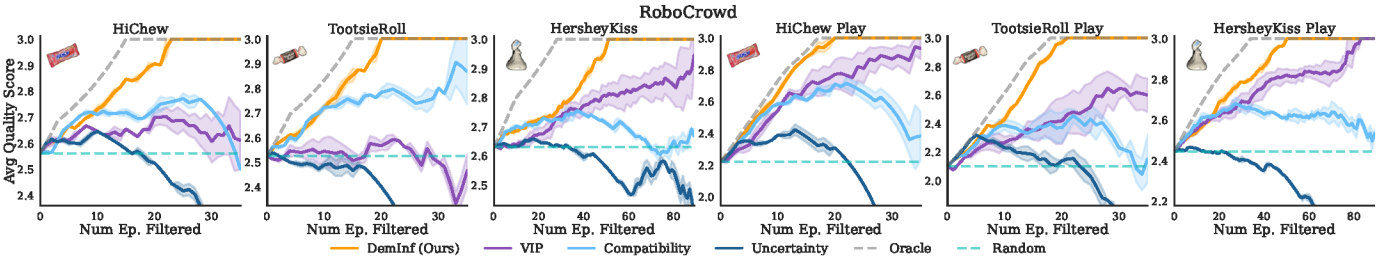

Fig. 5. Average quality of demonstrations remaining in datasets after filtering with different choices of S on the Hi-Chew, Tootsie-Roll, and Hershey-Kiss crowdsourced datasets from the RoboCrowd benchmark. We include results for datasets with a combination of expert and only task-relevant data (left), and a version of the data that contains additional unstructured play data (right). Results are shown as an average of 3 seeds.

InfoNCE (MI). We use the symmetric InfoNCE [56] objective used to train CLIP [62] which converges to an estimate of mutual information. We compare to InfoNCE as CLIP is commonly used to curate datasets in vision and language [64].
MINE (MI). MINE [4] leverages the dual form of the KL divergence to estimate the mutual information using a learned critic function.

3) Architectures: For all state-based experiments we use MLPs with two hidden layers of size 512. For image-based experiments we use ResNet-18 Encoders [29] with spatial softmax activations following Mandlekar et al. [50], which are concatenated with state information as input to a MLP with two hidden layers of size 1024. When training VAEs from images we use matching ResNet-18 Decoder networks for each view. For each dataset we use the same architecture for all methods, where the latent z dimension is set to be consistent across both DemInf and baselines. Images resized to 84×84 for RoboMimic and 128×128 otherwise. For all experiments we use the Adam optimizer with learning rate 0.0001 and a batch size of 256. State-based models are trained for 50,000 steps and image based models are trained for 100,000 steps using VMs provided by a Google TPU Research Cloud Grant. We run three seeds for all methods. More details and hyperparameters can be found in Appendix C.

B. How well does DemInf curate data?

To assess how well DemInf can curate data, we plot the number of episodes filtered from each dataset against the average resulting expert quality label. This amounts to considering every possible dataset generated by sweeping the threshold level κ in the sub-setted dataset according to scoring function S (see Eq. (3)). Doing so allows us to simultaneously assess how well each method would does at every threshold level. The closer the curve is to the "oracle" or curating directly by the expert labels, the better. Note that one should not over-index on the right-hand size of the plot as with a typical learning curve, as that represents performance only as the dataset reaches 10% of its original size.

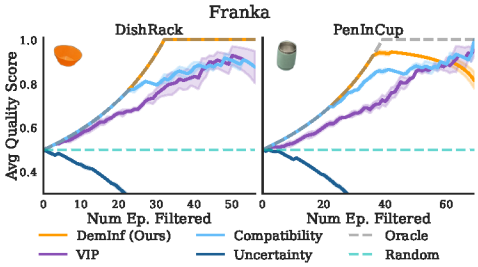

Fig. 6. Average quality of demonstrations remaining in datasets after filtering with different choices of S on the Franka Datasets. Average of 3 seeds.

State-Based Results. We depict results on the state-based RoboMimic benchmark on the left side of Fig. 4. DemInf performs as well or better than baselines in all environments, though there is not a particularly large gap with VIP.
Image-Based Results. In the image-based settings we find that DemInf performs even better, surpassing all methods on both RoboMimic (Fig. 4) and Franka (Fig. 6). On the "DishRack" task, DemInf is able to exactly match the oracle. VIP performs comparably worse in this setting, likely because its bootstrapping-based RL objective is more difficult to optimize in higher dimensions. Conversely, the uncertainty based metrics perform better in RoboMimic. The compatibility metric performs quite well on the Franka tasks, likely because the low quality data was explicitly collected with higher

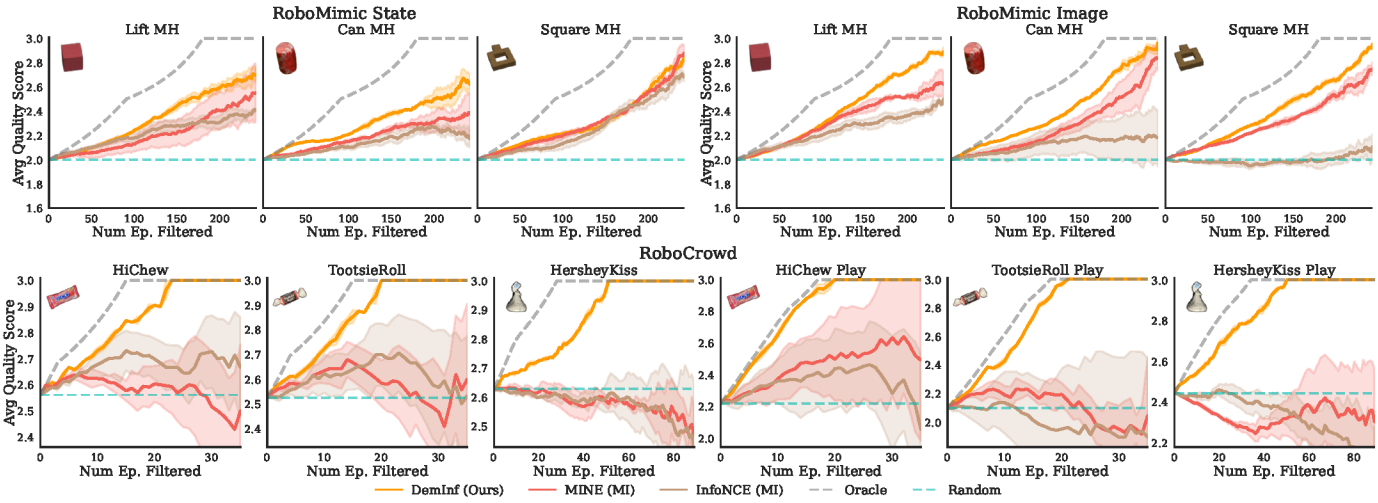

Fig. 7. Average quality of demonstrations remaining in datasets across RoboMimic and RoboCrowd after filtering with different mutual information estimators. Again, all experiments are averaged over 3 seeds. We found InfoNCE and MINE to exhibit higher variance than DemInf and struggle with higher dimensional inputs, especially with lower amounts of data.

entropy, making it easy to distinguish with policy loss alone. **Crowdsourced Data.** To assess DemInf's ability to filter data from a wide variety of operators, styles, and quality levels we turn to the RoboCrowd benchmark. We again find that DemInf most consistently filters out low-quality data with respect to the expert labels. The extreme diversity of these datasets, combined with the limited number of demonstrations available (40-100) proves extremely challenging for all baselines, which often provide only a small edge over random sampling. Selecting based on uncertainty performs quite poorly here – demonstrating that when learning offline, uncertainty is a poor metric, and certainty (its inverse) may perform better. These results suggest that methods designed for active learning and interactive data collection are not sufficient for the problem of offline data curation. When comparing the left side of Fig. 5 with the right side, we see that VIP is able to perform better on the "Play" datasets can contain task-irrelevant sequences in the demonstrations. While this might be counter-intuitive at first, VIP is goal-conditioned and scores state, next-state tuples based on perceived progress towards the goal. Thus, data with large amounts of irrelevant data extending the length of trajectories, VIP has an easier time filtering.

C. Mutual Information Estimators

Fig. 7 shows the performance of different mutual information estimators across RoboMimic and RoboCrowd. While InfoNCE and MINE perform acceptably in state-based settings, they begin to perform significantly worse in image based settings as the dimensionality of the data increases. InfoNCE in particular performs far worse in RoboMimic, underscoring the raw amount of data needed to train a high quality contrastive representation as documented by prior work. Both MINE and InfoNCE perform poorly in the more data-limited regime in RoboCrowd while DemInf, which uses non-parametric estimation no top of representation learning, is able to retain performance. Moreover, we find that DemInf exhibits far lower

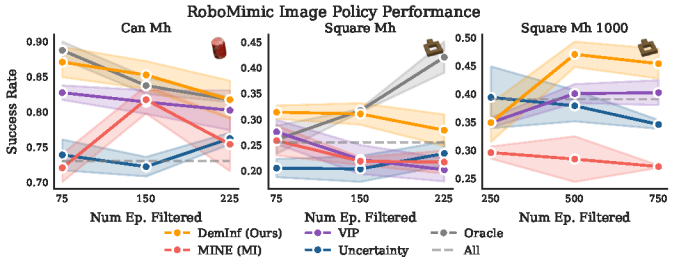

Fig. 8. Performance of ResNet18 + MLP BC Policies trained on filtered subsets of RoboMimic from Images. Evaluations are averaged over 200 trials for each of 3 seeds (600 total) after 100K training steps. Each dataset begins with 300 demonstrations, except for "Square Mh 1000" which has 1000.

variance across seeds, while the parametric estimators were more unstable and had one or two runs that performed far worse than the others. This is particularly problematic for downstream data curation, as one often does not have ground truth labels to check the quality of the scoring function.

D. Does demonstration curation affect policy performance?

While comparing to ground truth labels allows us to assess the quality of different approaches to filtering, we ultimately care about the performance of downstream BC policies. In Fig. 8 we train BC policies on RoboMimic Can MH and Square MH from images when filtering different numbers of episodes from the dataset $\mathcal{D}_N$ according to the best baselines for the tasks. Overall, we find that at all data scales DemInf performs better than baselines, which exhibit far less consistent performance trends overall. For example, uncertainty often shows little improvement until the majority of the dataset is filtered. Crucially, we see that filtering data with DemInf performs *better* than training on all of the data by over 10% in Can and is the only method to improve upon training on all of the data in Square. To further assess performance on larger datasets, we add an additional 700 successful rollouts from Diffusion Policy [14] to Square MH to bring the total to 1000 demos for "Square MH 1000". Even with larger

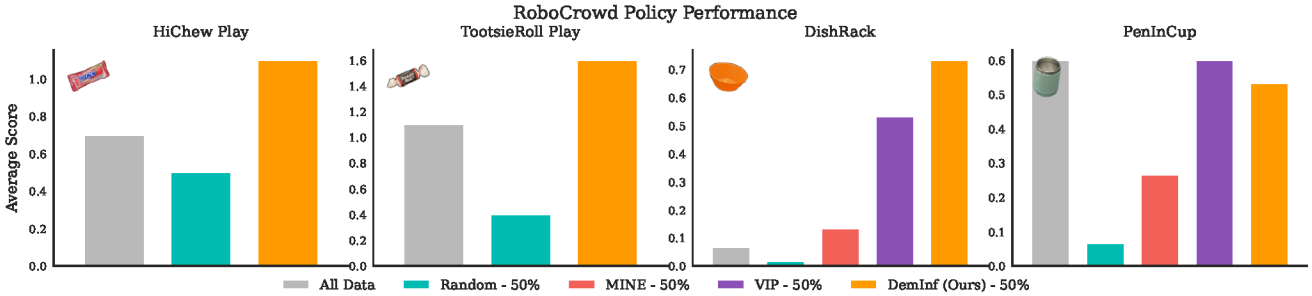

Fig. 9. Performance of ACT trained on filtered versus non-filtered RoboCrowd (HiChew and TootsieRoll) and Diffusion Policy trained on filtered versus non-filtered data for the Franka tasks (DishRack and PenInCup). Evaluations for RoboCrowd are shown using the same scoring procedure from Mirchandani et al. [52] over 10 trials, and evaluations for Franka are computed using binary success over 15 trials.

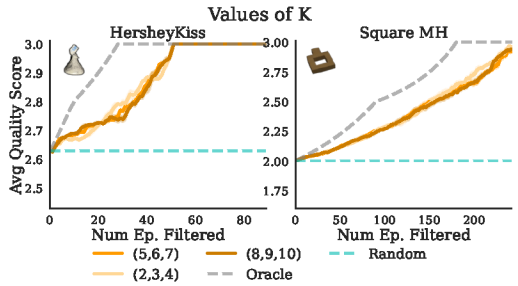

Fig. 10. Performance of DemInf with different ranges of k for k-NN

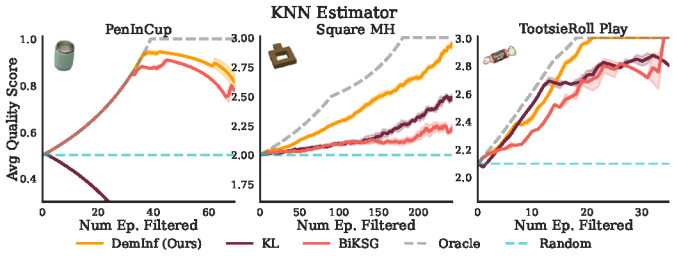

Fig. 11. Performance of DemInf with different k-NN mutual information estimators

datasets, DemInf maintains performance. We include results for robomimic state in Appendix A.

This trend continues in real world evals for both RoboCrowd and Franka where we compare training policies on all of the demonstrations (no filter), training policies on a random 50% subset of the demonstrations, and filtering 50% of the demonstrations with DemInf in Fig. 9. We train ACT [76] for RoboCrowd and Diffusion Policy [14] for Franka. In RoboCrowd, after scoring trials according to the methodology in Mirchandani et al. [52] (1 for grasping any number of candies, 2 for returning it to the user, and 3 for returning only one as in the demonstrations), we find that the DemInf policy not only more commonly successfully completes the task, but also exhibits better motion when compared to training on all of the data. When considering the same number of demonstrations randomly selected from the dataset, we find that the gap in score is larger, indicating that better performance can be attained by collecting only good data. On the Franka tasks DemInf always outperforms the random subset, but training on all data does slightly better than filtering with DemInf on "PenInCup". This is likely because "PenInCup", being the harder task, requires more data for representation learning even if the quality is poor. However, when dataset size is kept at parity (50% random subset) we see a huge difference in performance, indicating that in all tasks data quality matters. VIP does well on the Franka tasks as well, which can be attributed to the fact that the purposefully suboptimal demonstrations we collected were significantly longer than the expert demonstrations, making filtering via a value function particularly effect.

E. Ablations

Finally, we consider which design choices of DemInf affect its ability to curate demonstration data. Here, we consider the value of k used for k-NN and the type of non-parameteric mutual information estimator. In Appendix A we include additional ablations over the size of the latent dimensions of z_s and z_a and the value of β for the VAEs. Fig. 10 shows the performance of DemInf with different ranges of k used to compute the mutual information. We average final predictions over this range. DemInf's performance is generally robust to this parameter, with no substantial change in performance in both HersheyKiss and Square MH. However, the story is different for the choice of k-NN estimator. As shown in Fig. 11, we found the KSG estimator from Kraskov et al. [40] to be superior to both the BiKSG estimator from Belghazi et al. [3] and the naïve application of the differential entropy estimator from Kozachenko and Leonenko [39] (KL). This indicates that the quality of the latent space, as well as the quality of the estimator, are important for downstream performance.

VII. Conclusion

In this work, we propose the Demonstration Information Estimation (DemInf) procedure as a method for data curation in robot imitation learning. Specifically, we motivate mutual information as a useful basis for measuring the quality of individual demonstrations, and instantiate mutual information estimators as a way to rank and select demonstrations. Across several datasets of human-teleoperated demonstrations in both the real-world and simulation, we find that the DemInf outperforms several prior methods at measuring the quality of demonstrations.

A. Limitations

Time Evolution. DemInf considers the aggregate mutual information between states and actions within the dataset. Our analyses, however, largely ignores the fact that these quantities are linked in the sequential setting via the environment dynamics. Concretely, we can increase the marginal action entropy with a more random policy, which will likely increase the state entropy as well through the dynamics. However, this comes at the cost of increasing $H(A|S)$, and it is unclear how DemInf balances these factors. An approach to disentangle the effects of dynamics might consider measuring $I(S_1; A_1, \ldots, A_T)$, or the mutual information between initial states and action sequences. However, doing so reduces the amount of data available for estimation by a factor of T, which is typically 100-1000, making this approach more challenging in practice.
Pauses. Because DemInf considers the average estimated $\hat{I}$ across a trajectory, it is susceptible to preferring data that is predictable, but might not make progress towards completing the task. For example, if a robot pauses for an extended amount of time at a particular state, the action distribution is very predictable. However, this behavior is not desired in practice. To mitigate this effect, we recommend ensuring that all data completes the task and pauses are filtered.
Greediness. Note that DemInf's curation procedure is greedy and not globally optimal – once we remove an episode we have changed the data distribution, which in turn affects the true mutual information. However, re-running the mutual information estimator on the entire dataset for each filtered demonstration would be far more computationally expensive.
Success. DemInf assumes that all provided demonstrations are successful at completing the desired task.

B. Future Work

Exciting avenues for future work remain. For instance, extending DemInf to the multi-task setting will require disentangling task conditioning from mutual information estimation as to not retain only the easiest tasks. This problem becomes harder in settings where task definitions are not enumerable, like natural language. Other directions include scaling DemInf to larger datasets such as Open X-Embodiment Collaboration et al. [57] and Khazatsky et al. [34] to curate better subsets for the robot learning community. Finally, integrating DemInf into an online data collection interface could improve data collection efficiency. Though there is more work to do, we believe DemInf is a step towards addressing the data problem in robotics.

Acknowledgments

Compute for this research was provided by a Google TPU Research Cloud Grant. This work was supported by ONR project N00014-22-1-2293. We would also like to thank the Stanford IRIS Lab for sharing their ALOHA robot for our final real world experiments and Kaylee Burns for providing feedback and help with set up.

References

[1] Amro Abbas, Kushal Tirumala, Dániel Simig, Surya Ganguli, and Ari S. Morcos. Semdedup: Data-efficient learning at web-scale through semantic deduplication, 2023.

[2] Alon Albalak, Yanai Elazar, Sang Michael Xie, Shayne Longpre, Nathan Lambert, Xinyi Wang, Niklas Muennighoff, Bairu Hou, Liangming Pan, Haewon Jeong, Colin Raffel, Shiyu Chang, Tatsunori Hashimoto, and William Yang Wang. A survey on data selection for language models, 2024.

[3] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. In *International conference on machine learning*, pages 531–540. PMLR, 2018.

[4] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeswar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and R Devon Hjelm. Mine: mutual information neural estimation. *arXiv preprint arXiv:1801.04062*, 2018.

[5] Suneel Belkhale, Yuchen Cui, and Dorsa Sadigh. Hydra: Hybrid robot actions for imitation learning. In *Conference on Robot Learning*, pages 2113–2133. PMLR, 2023.

[6] Suneel Belkhale, Yuchen Cui, and Dorsa Sadigh. Data quality in imitation learning. *Advances in Neural Information Processing Systems*, 36, 2024.

[7] Homanga Bharadhwaj, Jay Vakil, Mohit Sharma, Abhinav Gupta, Shubham Tulsiani, and Vikash Kumar. Roboagent: Generalization and efficiency in robot manipulation via semantic augmentations and action chunking. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4788–4795. IEEE, 2024.

[8] Vighnesh Birodkar, Hossein Mobahi, and Samy Bengio. Semantic redundancies in image-classification datasets: The 10% you don't need. *arXiv preprint arXiv:1901.11409*, 2019.

[9] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. Rt-1: Robotics transformer for real-world control at scale. In *arXiv preprint arXiv:2212.06817*, 2022.

[10] Kaylee Burns, Zach Witzel, Jubayer Ibn Hamid, Tianhe Yu, Chelsea Finn, and Karol Hausman. What makes

pre-trained visual representations successful for robust manipulation? *arXiv preprint arXiv:2312.12444*, 2023.

[11] Mayee Chen, Nicholas Roberts, Kush Bhatia, Jue Wang, Ce Zhang, Frederic Sala, and Christopher Ré. Skill-it! a data-driven skills framework for understanding and training language models. *Advances in Neural Information Processing Systems*, 36, 2024.

[12] Mayee F Chen, Michael Y Hu, Nicholas Lourie, Kyunghyun Cho, and Christopher Ré. Aioli: A unified optimization framework for language model data mixing. *arXiv preprint arXiv:2411.05735*, 2024.

[13] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[14] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, page 02783649241273668, 2023.

[15] Cheng Chi, Zhenjia Xu, Chuer Pan, Eric Cousineau, Benjamin Burchfiel, Siyuan Feng, Russ Tedrake, and Shuran Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. In *Proceedings of Robotics: Science and Systems (RSS)*, 2024.

[16] Together Computer. Redpajama: an open dataset for training large language models, October 2023. URL https://github.com/togethercomputer/RedPajama-Data.

[17] Thomas M Cover. *Elements of information theory*. John Wiley & Sons, 1999.

[18] Yuchen Cui, David Isele, Scott Niekum, and Kikuo Fujimura. Uncertainty-aware data aggregation for deep imitation learning. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 761–767. IEEE, 2019.

[19] Paweł Czyż, Frederic Grabowski, Julia Vogt, Niko Beerenwinkel, and Alexander Marx. Beyond normal: On the evaluation of mutual information estimators. In A. Oh, T. Neumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, 2023.

[20] Sudeep Dasari, Frederik Ebert, Stephen Tian, Suraj Nair, Bernadette Bucher, Karl Schmeckpeper, Siddharth Singh, Sergey Levine, and Chelsea Finn. Robonet: Large-scale multi-robot learning. *arXiv preprint arXiv:1910.11215*, 2019.

[21] Maximilian Du, Suraj Nair, Dorsa Sadigh, and Chelsea Finn. Behavior retrieval: Few-shot imitation learning by querying unlabeled datasets. *arXiv preprint arXiv:2304.08742*, 2023.

[22] Simin Fan, Matteo Pagliardini, and Martin Jaggi. Doge: Domain reweighting with generalization estimation. In *Forty-first International Conference on Machine Learning*, 2023.

[23] Alex Fang, Albin Madappally Jose, Amit Jain, Ludwig Schmidt, Alexander T Toshev, and Vaishaal Shankar. Data filtering networks. In *The Twelfth International Conference on Learning Representations*, 2024.

[24] Hao-Shu Fang, Hongjie Fang, Zhenyu Tang, Jirong Liu, Junbo Wang, Haoyi Zhu, and Cewu Lu. Rh20t: A robotic dataset for learning diverse skills in one-shot. In *RSS 2023 Workshop on Learning for Task and Motion Planning*, 2023.

[25] Kanishk Gandhi, Siddharth Karamcheti, Madeline Liao, and Dorsa Sadigh. Eliciting compatible demonstrations for multi-human imitation learning. In Karen Liu, Dana Kulic, and Jeff Ichnowski, editors, *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pages 1981–1991. PMLR, 14–18 Dec 2023.

[26] Jensen Gao, Annie Xie, Ted Xiao, Chelsea Finn, and Dorsa Sadigh. Efficient data collection for robotic manipulation via compositional generalization, 2024.

[27] Weihao Gao, Sewoong Oh, and Pramod Viswanath. Demystifying fixed k-nearest neighbor information estimators. *IEEE Transactions on Information Theory*, 64(8): 5629–5661, 2018.

[28] Huy Ha, Pete Florence, and Shuran Song. Scaling up and distilling down: Language-guided robot skill acquisition. In *Proceedings of the 2023 Conference on Robot Learning*, 2023.

[29] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[30] Joey Hejna, Chethan Anand Bhateja, Yichen Jiang, Karl Pertsch, and Dorsa Sadigh. Remix: Optimizing data mixtures for large scale imitation learning. In *8th Annual Conference on Robot Learning*, 2024.

[31] Ryan Hoque, Ashwin Balakrishna, Ellen Novoseller, Albert Wilcox, Daniel S. Brown, and Ken Goldberg. ThriftyDAgger: Budget-aware novelty and risk gating for interactive imitation learning. In *5th Annual Conference on Robot Learning*, 2021.

[32] Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pages 991–1002. PMLR, 2022.

[33] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[34] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. In *Robotics: Science and Systems*, 2024.

[35] Dongyoung Kim, Jinwoo Shin, Pieter Abbeel, and Younggyo Seo. Accelerating reinforcement learning with value-conditional state entropy exploration. *Advances in Neural Information Processing Systems*, 36, 2024.

[36] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An open-source vision-language-action model. In *8th Annual Conference on Robot Learning*, 2024.

[37] Diederik P Kingma. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[38] Nikolaos Kouiroukidis and Georgios Evangelidis. The effects of dimensionality curse in high dimensional knn search. In *2011 15th Panhellenic Conference on Informatics*, pages 41–45, 2011. doi: 10.1109/PCI.2011.45.

[39] Lyudmyla F Kozachenko and Nikolai N Leonenko. Sample estimate of the entropy of a random vector. *Problemy Peredachi Informatsii*, 23(2):9–16, 1987.

[40] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics*, 69(6): 066138, 2004.

[41] Sachit Kuhar, Shuo Cheng, Shivang Chopra, Matthew Bronars, and Danfei Xu. Learning to discern: Imitating heterogeneous human demonstrations with preference and representation learning. In *7th Annual Conference on Robot Learning*, 2023.

[42] Sergey Levine, Peter Pastor, Alex Krizhevsky, Julian Ibarz, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International journal of robotics research*, 37(4-5):421–436, 2018.

[43] Fanqi Lin, Yingdong Hu, Pingyue Sheng, Chuan Wen, Jiacheng You, and Yang Gao. Data scaling laws in imitation learning for robotic manipulation, 2024.

[44] Li-Heng Lin, Yuchen Cui, Amber Xie, Tianyu Hua, and Dorsa Sadigh. Flowretrieval: Flow-guided data retrieval for few-shot imitation learning. In *8th Annual Conference on Robot Learning*, 2024.

[45] Hao Liu and Pieter Abbeel. Behavior from the void: Unsupervised active pre-training. *Advances in Neural Information Processing Systems*, 34:18459–18473, 2021.

[46] Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. VIP: towards universal visual reward and representation via value-implicit pre-training. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[47] Zhuang Ma and Michael Collins. Noise contrastive estimation and negative sampling for conditional models: Consistency and statistical efficiency. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3698–3707, 2018.

[48] Zhao Mandi, Homanga Bharadhwaj, Vincent Moens, Shuran Song, Aravind Rajeswaran, and Vikash Kumar. Cacti: A framework for scalable multi-task multi-scene visual imitation learning. *arXiv preprint arXiv:2212.05711*, 2022.

[49] Ajay Mandlekar, Yuke Zhu, Animesh Garg, Jonathan Booher, Max Spero, Albert Tung, Julian Gao, John Emmons, Anchit Gupta, Emre Orbay, et al. Roboturk: A crowdsourcing platform for robotic skill learning through imitation. In *Conference on Robot Learning*, pages 879–893. PMLR, 2018.

[50] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *5th Annual Conference on Robot Learning*, 2021.

[51] Ajay Mandlekar, Soroush Nasiriany, Bowen Wen, Iretiayo Akinola, Yashraj Narang, Linxi Fan, Yuke Zhu, and Dieter Fox. Mimicgen: A data generation system for scalable robot learning using human demonstrations. In *7th Annual Conference on Robot Learning*, 2023.

[52] Suvir Mirchandani, David D. Yuan, Kaylee Burns, Md Sazzad Islam, Tony Z. Zhao, Chelsea Finn, and Dorsa Sadigh. Robocrowd: Scaling robot data collection through crowdsourcing. *arXiv*, 2024.

[53] Soroush Nasiriany, Tian Gao, Ajay Mandlekar, and Yuke Zhu. Learning and retrieval from prior data for skill-based imitation learning. In *Conference on Robot Learning (CoRL)*, 2022.

[54] Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. Robocasa: Large-scale simulation of everyday tasks for generalist robots. In *Robotics: Science and Systems (RSS)*, 2024.

[55] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Charles Xu, Jianlan Luo, Tobias Kreiman, You Liang Tan, Lawrence Yunliang Chen, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, 2024.

[56] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[57] Open X-Embodiment Collaboration, Abhishek Padalkar, Acorn Pooley, Ajinkya Jain, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anikait Singh, Anthony Brohan, Antonin Raffin, Ayzaan Wahid, Ben Burgess-Limerick, Beomjoon Kim, Bernhard Schölkopf, Brian Ichter, Cewu Lu, Charles Xu, Chelsea Finn, Chenfeng Xu, Cheng Chi, Chenguang Huang, Christine Chan, Chuer Pan, Chuyuan Fu, Coline Devin, Danny Driess, Deepak Pathak, Dhruv Shah, Dieter Büchler, Dmitry Kalashnikov, Dorsa Sadigh, Edward Johns, Federico Ceola, Fei Xia, Freek Stulp, Gaoyue

Zhou, Gaurav S. Sukhatme, Gautam Salhotra, Ge Yan, Giulio Schiavi, Hao Su, Hao-Shu Fang, Haochen Shi, Heni Ben Amor, Henrik I Christensen, Hiroki Furuta, Homer Walke, Hongjie Fang, Igor Mordatch, Ilija Radosavovic, Isabel Leal, Jacky Liang, Jaehyung Kim, Jan Schneider, Jasmine Hsu, Jeannette Bohg, Jeffrey Bingham, Jiajun Wu, Jialin Wu, Jianlan Luo, Jiayuan Gu, Jie Tan, Jihoon Oh, Jitendra Malik, Jonathan Tompson, Jonathan Yang, Joseph J. Lim, João Silvério, Junhyek Han, Kanishka Rao, Karl Pertsch, Karol Hausman, Keegan Go, Keerthana Gopalakrishnan, Ken Goldberg, Kendra Byrne, Kenneth Oslund, Kento Kawaharazuka, Kevin Zhang, Keyvan Majd, Krishan Rana, Krishnan Srinivasan, Lawrence Yunliang Chen, Lerrel Pinto, Liam Tan, Lionel Ott, Lisa Lee, Masayoshi Tomizuka, Maximilian Du, Michael Ahn, Mingtong Zhang, Mingyu Ding, Mohan Kumar Srirama, Mohit Sharma, Moo Jin Kim, Naoaki Kanazawa, Nicklas Hansen, Nicolas Heess, Nikhil J Joshi, Niko Suenderhauf, Norman Di Palo, Nur Muhammad Mahi Shafiullah, Oier Mees, Oliver Kroemer, Pannag R Sanketi, Paul Wohlhart, Peng Xu, Pierre Sermanet, Priya Sundaresan, Quan Vuong, Rafael Rafailov, Ran Tian, Ria Doshi, Roberto Martín-Martín, Russell Mendonca, Rutav Shah, Ryan Hoque, Ryan Julian, Samuel Bustamante, Sean Kirmani, Sergey Levine, Sherry Moore, Shikhar Bahl, Shivin Dass, Shuran Song, Sichun Xu, Siddhant Haldar, Simeon Adebola, Simon Guist, Soroush Nasiriany, Stefan Schaal, Stefan Welker, Stephen Tian, Sudeep Dasari, Suneel Belkhale, Takayuki Osa, Tatsuya Harada, Tatsuya Matsushima, Ted Xiao, Tianhe Yu, Tianli Ding, Todor Davchev, Tony Z. Zhao, Travis Armstrong, Trevor Darrell, Vidhi Jain, Vincent Vanhoucke, Wei Zhan, Wenxuan Zhou, Wolfram Burgard, Xi Chen, Xiaolong Wang, Xinghao Zhu, Xuanlin Li, Yao Lu, Yevgen Chebotar, Yifan Zhou, Yifeng Zhu, Ying Xu, Yixuan Wang, Yonatan Bisk, Yoonyoung Cho, Youngwoon Lee, Yuchen Cui, Yueh hua Wu, Yujin Tang, Yuke Zhu, Yunzhu Li, Yusuke Iwasawa, Yutaka Matsuo, Zhuo Xu, and Zichen Jeff Cui. Open X-Embodiment: Robotic learning datasets and RT-X models, 2023.

[58] Guilherme Penedo, Hynek Kydlíček, Leandro von Werra, and Thomas Wolf. Fineweb, April 2024. URL https://huggingface.co/datasets/HuggingFaceFW/fineweb.

[59] Lerrel Pinto and Abhinav Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 3406–3413. IEEE, 2016.

[60] Ben Poole, Sherjil Ozair, Aaron Van Den Oord, Alex Alemi, and George Tucker. On variational bounds of mutual information. In *International Conference on Machine Learning*, pages 5171–5180. PMLR, 2019.

[61] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8): 9, 2019.

[62] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[63] Stéphane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 661–668. JMLR Workshop and Conference Proceedings, 2010.

[64] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems*, 35:25278–25294, 2022.

[65] Pratyusha Sharma, Lekha Mohan, Lerrel Pinto, and Abhinav Gupta. Multiple interactions made easy (mime): Large scale demonstrations data for imitation. In *Conference on robot learning*, pages 906–915. PMLR, 2018.

[66] Harshinder Singh, Neeraj Misra, Vladimir Hnizdo, Adam Fedorowicz, and Eugene Demchuk. Nearest neighbor estimates of entropy. *American journal of mathematical and management sciences*, 23(3-4):301–321, 2003.

[67] Kushal Tirumala, Daniel Simig, Armen Aghajanyan, and Ari Morcos. D4: Improving llm pretraining via document de-duplication and diversification. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, 2023.

[68] Huy V Vo, Vasil Khalidov, Timothée Darcet, Théo Moutakanni, Nikita Smetanin, Marc Szafraniec, Hugo Touvron, Camille Couprie, Maxime Oquab, Armand Joulin, et al. Automatic data curation for self-supervised learning: A clustering-based approach. *arXiv preprint arXiv:2405.15613*, 2024.

[69] Homer Walke, Kevin Black, Abraham Lee, Moo Jin Kim, Max Du, Chongyi Zheng, Tony Zhao, Philippe Hansen-Estruch, Quan Vuong, Andre He, Vivek Myers, Kuan Fang, Chelsea Finn, and Sergey Levine. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning (CoRL)*, 2023.

[70] Annie Xie, Lisa Lee, Ted Xiao, and Chelsea Finn. Decomposing the generalization gap in imitation learning for visual robotic manipulation. *arXiv preprint arXiv:2307.03659*, 2023.

[71] Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy S Liang, Quoc V Le, Tengyu Ma, and Adams Wei Yu. Doremi: Optimizing data mixtures speeds up language model pretraining. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, 2023.

[72] Hu Xu, Saining Xie, Xiaoqing Tan, Po-Yao Huang,

Russell Howes, Vasu Sharma, Shang-Wen Li, Gargi Ghosh, Luke Zettlemoyer, and Christoph Feichtenhofer. Demystifying CLIP data. In *The Twelfth International Conference on Learning Representations*, 2024.

[73] Sarah Young, Dhiraj Gandhi, Shubham Tulsiani, Abhinav Gupta, Pieter Abbeel, and Lerrel Pinto. Visual imitation made easy. In Jens Kober, Fabio Ramos, and Claire Tomlin, editors, *Proceedings of the 2020 Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pages 1992–2005. PMLR, 16–18 Nov 2021.

[74] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *CVPR*, 2022.

[75] Yan Zhang, Ruidan He, Zuozhu Liu, Kwan Hui Lim, and Lidong Bing. An unsupervised sentence embedding method by mutual information maximization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1601–1610, 2020.

[76] Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In Kostas E. Bekris, Kris Hauser, Sylvia L. Herbert, and Jingjin Yu, editors, *Robotics: Science and Systems XIX, Daegu, Republic of Korea, July 10-14, 2023*, 2023. doi: 10.15607/RSS.2023.XIX.016.

[77] Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pages 2165–2183. PMLR, 2023.

APPENDIX

A. *Extended Results*

Here we provide results and ablations that could not fit in the main text.

Additional Results

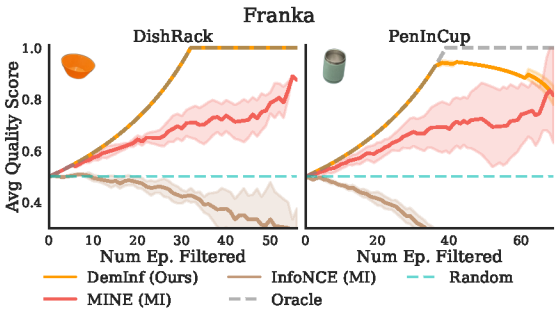

Fig. 12. The performance of different mutual information estimators on the Franka Datasets, cut from the main text due to space.

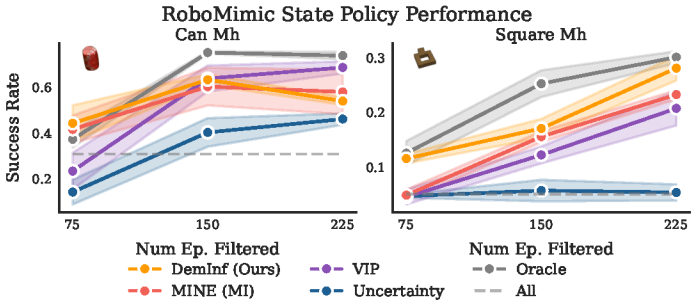

Fig. 13. RoboMimic Policy learning performance from state.

Additional Ablations

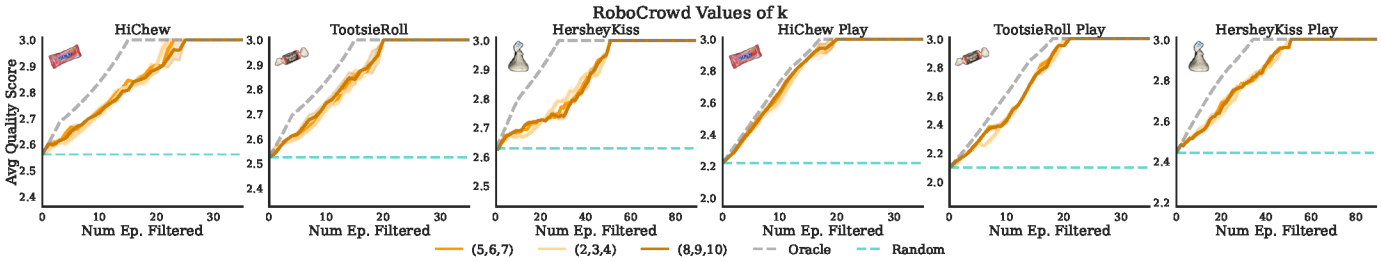

Fig. 14. The effect of different values of k on RoboCrowd

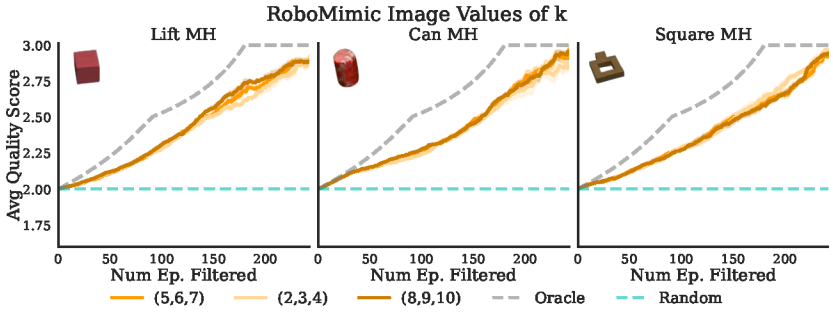

Fig. 15. The effect of different values of k on RoboMimic Image

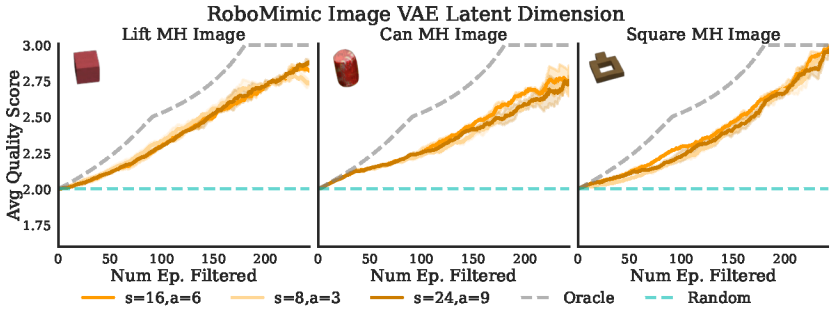

Fig. 16. The effect of different latent dimension sizes for z_s and z_a on RoboMimic Image. we find that performance is relatively robust to this parameter. Unlike all others, this experiment was run over 2, not 3, seeds.

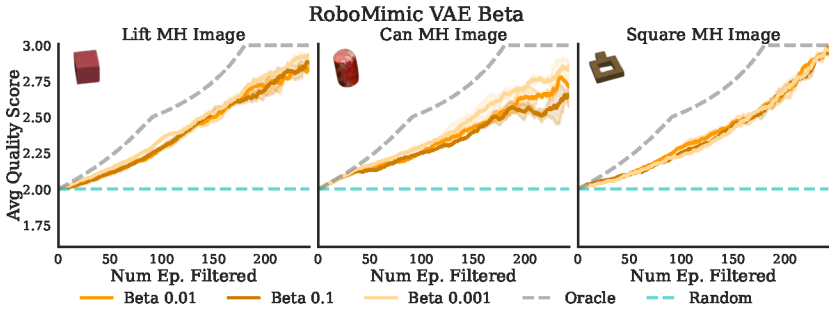

Fig. 17. The effect of different values of VAE β on RoboMimic Image. We find that performance is relatively robust to this parameter for RoboMimic. Unlike all others, this experiment was run over 2, not 3, seeds.

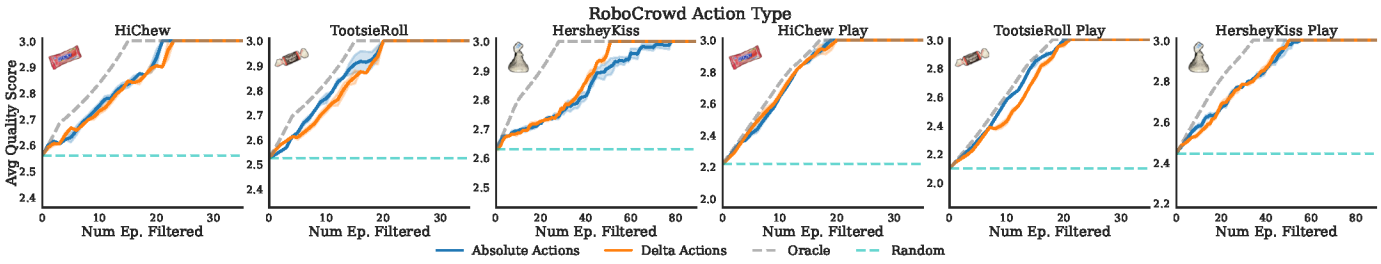

Fig. 18. The effect of using different action spaces for the RoboCrowd dataset.

Plots with All Baselines and Estimators. The below plots show the performance of all methods on the same exact plot, allowing for direct comparison. We additionally consider another baseline "Policy Loss", which simply measures the loss of a BC policy.

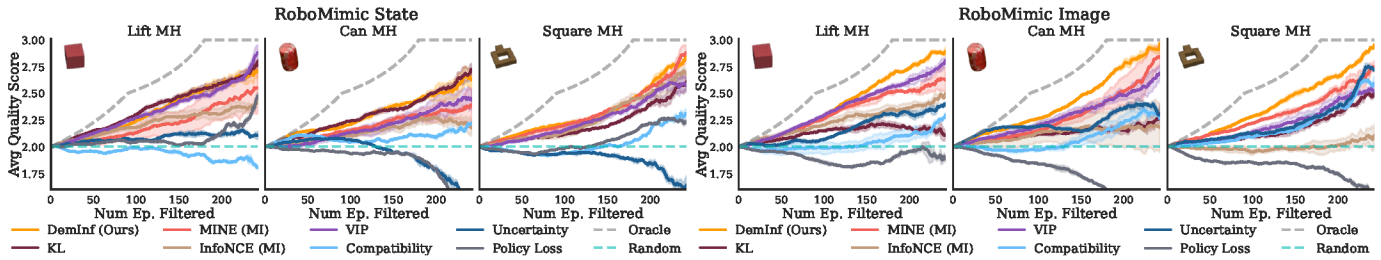

Fig. 19. RoboMimic results for all methods.

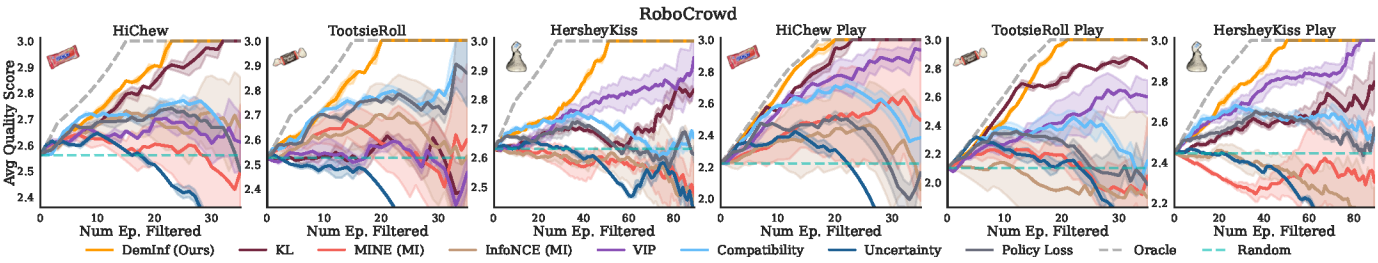

Fig. 20. RoboCrowd results for all methods.

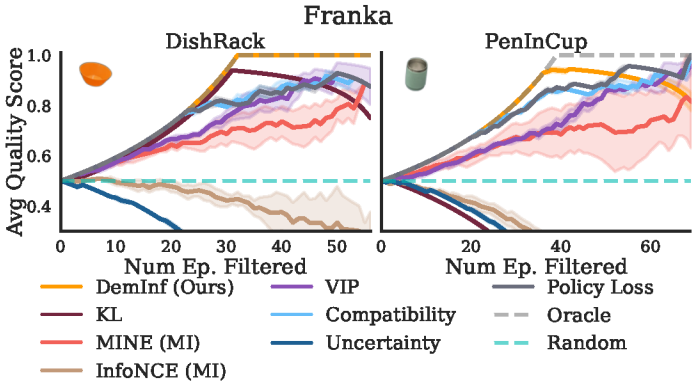

Fig. 21. Franka results for all methods.

B. Method Details

Here we provide more details on each of the different methods used to for data curation. For each method we train the requisite model(s), then run inference over the whole dataset. As we filter at the demonstration level, we aggregate scores for all methods over each demonstration. Mathematically, all scoring functions take the form:

$$S(\tau) = \frac{1}{T}\sum_{t=1}^{T} h(s_t, a_t; \mathcal{D}_n)$$

where h is some function of the state-action pairs in a set of data $\mathcal{D}$ comprised of trajectories τ. We use the subscript t to denote that we index over the steps of a trajectory τ. In practice we clip state-action scores from h at the 1st and 99th percentile. Below we provide the scoring function used for all methods.

Demonstration Information Estimation (DemInf). We first fit an action VAE $z_a = f_a(a)$ and state VAE $z_s = f_s(s)$. Then, we iterate over the entire dataset 4 times, computing scores in random batches of size 1024. The score function is then

$$h(s_i, a_i; B) = \hat{I}(s_i, a_i; B) \propto -\psi(n(z_{s,i}) + 1) - \psi(n(z_{a,i}) + 1)$$

where B is a random batch and n is defined as Section V.

BiKSG. We follow the same approach as in DemInf, except the mutual information is estimated as

$$h(s_i, a_i; B) = \hat{I}(s_i, a_i; B) \propto -\log n(z_{s,i}) - \log n(z_{a,i})$$

and we use the $l2$ distance metric over $\mathcal{Z}_\mathcal{S} \times \mathcal{Z}_\mathcal{A}$ without the l_∞ norm:

$$||[z_s, z_a] - [z'_s, z'_a]|| = |||[z_s, z_a] - [z'_s, z'_a]||_2$$

KL. We follow the same approach as in DemInf, except the mutual information is estimated using separate terms for $H(S)$, $H(A)$ and $H(S, A)$ where each term is given by the differentiable entropy estimator from Kozachenko and Leonenko [39]. Let $z^k_{s,i}$ be $z_{s,i}$'s k-nearest-neighbor. Then, the estimator is given as

$$h(s_i, a_i; B) = \hat{I}(s_i, a_i; B) \propto \log ||z_{s,i} - z^k_{s,i}||_2^{|\mathcal{Z}_\mathcal{S}|} + \log ||z_{a,i} - z^k_{a,i}||_2^{|\mathcal{Z}_\mathcal{A}|} - \log |||[z_s, z_a]_i - [z_s, z_a]^k_i||_2^{|\mathcal{Z}_\mathcal{S}|+|\mathcal{Z}_\mathcal{A}|}$$

where $|\mathcal{Z}_\mathcal{S}|$ is the dimension of the latent space.

MINE. MINE optimizes a critic function $f_\theta(s, a)$ to predict the mutual information using the objective

$$\max_\theta \mathbb{E}_{(s,a)\sim\mathcal{D}}[f_\theta(s,a)] - \log\left(\mathbb{E}_{s\sim\mathcal{D},a\sim D}[e^{f_\theta(s,a)}\right)$$

where the first term is sampled from the joint and the second is sampled from the marginals. The scoring function is then simply:

$$h(s_i, a_i; B) = f_\theta(s_i, a_i)$$

In practice MINE uses an exponential moving average of gradient's denominator to un-bias the estimator. We refer to this parameter as α as in the original paper and leave it at 0.9.

InfoNCE. We optimize the symmetric InfoNCE objective from CLIP, which converges to the mutual information up to a constant [47]. To do so, we train a state encoder f_s and an action encoder f_a. After training, the scoring function becomes:

$$h(s_i, a_i; B) = f_s(s_i) \cdot f_a(a_i)$$

or simply the dot product between the two representations.

VIP. VIP [46] uses the dual form of the goal-conditioned RL problem, with the negative L2 distance between encoded states as a proxy for the value function $V(s, g) = -||f(s) - f(g)||_2$. The VIP training objective is

$$\min_\theta \mathbb{E}_{s_1\sim\rho_1,g\sim D}[||f_\theta(s_1) - f_\theta(g)||_2] + \log \mathbb{E}_{s_t,s_{t+1},g\sim\mathcal{D}}[\exp\left(||f_\theta(s_t) - f_\theta(g)||_2 - \mathbb{1}\{s_t = g\} - \gamma||f_\theta(s_{t+1}) - f_\theta(g)||_2\right)]$$

Then, using the learned value function we estimate the "reward" of each transition by

$$h(s_t, s_{t+1}, g; B) = -||f_\theta(s_{t+1}) - f_\theta(g)||_2 + ||f_\theta(s_t) - f_\theta(g)||_2$$

which captures the progress of the transition towards the goal. During training we sample goals uniformly from the future, but during quality estimation we set the goals to be the final state in each demonstration.

Compatibility. Following Gandhi et al. [25] we train an ensemble of 5 policies. Then, the compatibility score is estimated as:

$$h(s_i, a_i; B) = \begin{cases} 1 - \min\left(\text{L2Loss}(\pi_\theta(s_i), a_i)/\lambda, 1\right) & \text{if std}(\pi_\theta(s_i)) < \eta \\ 1 & \text{otherwise} \end{cases}$$

where L2Loss is the average L2 loss of the ensemble and std is the standard deviation of the predictions.

Uncertainty. The uncertainty score is estimated from the same ensemble of 5 policies by the standard deviation of the predictions:

$$h(s_i, a_i; B) = \text{std}(\pi_\theta(s_i))$$

Policy Loss. The Policy Loss metric is simply the negative L2 Loss of the network, such that demonstrations with lower loss have a higher score.

$$h(s_i, a_i; B) = -\text{L2Loss}(\pi_\theta(s_i), a_i)$$

C. Implementation Details

Architectures. We use the same architectures for all methods whenever possible. For state-based experiments we simply use MLPs with two hidden layers of size 512 with ReLU activations. When training BC policies, we add dropout of 0.5 as we found it to be important to performance. For VAEs we use a symmetric decoder.

For Image experiments, we use ResNet18 architectures followed by a spatial softmax layer, similar to the original setup in Mandlekar et al. [50]. We concatenate representations from all cameras along with the state information, and then feed that to information to an MLP. For RoboMimic we use a three layer MLP with hidden dimension of size 512. For Franka and RoboCrowd we use an MLP with two hidden layers of size 1024. For all methods using a state encoder, we use this architecture. For BC policies we ensemble the MLP, add dropout and use the L2 Loss function for training. MINE additionally concatenates the action before the MLP and InfoNCE trains a separate action encoder using just the MLP architecture. For action encoders and decoders, we use the same architecture as for state. For training VAEs on images, we use the same architecture but in reverse, with ResNet18 Decoders.

We trained all models on TPU v4-8 VMs provided by the Google TPU Research Cloud. Training for 100K steps took approximately 30 minutes for VAEs and 1 hour for other methods.

Method	Parameter	RoboMimic State	RoboMimic Image	Franka	RoboCrowd
All	Optimizer		Adam		
	Learning Rate		0.0001		
	Batch Size		256		
	Training Steps	50,000		100,000	
	Action Chunk	1	1	4	10
	Image Resolution	–	(84, 84)	(128, 128)	(128, 128)
DemInf	Augmentations	–	Random Scale and Crop (0.9, 0.95)		
	β	0.05		0.01	
	Image Recon Weight	–		0.005	
	z_s	12	16	24	16
	z_a	6	6	16	12
	k		(5,6,7)		
VIP	z	8	16	24	16
	γ		0.98		
InfoNCE	z	8	16	24	16
MINE	z	8	16	24	16
	α		0.9		
Compatibility	Ensemble Size		5		
	Dropout		0.5		
	η	0.025	0.05	0.1	0.05
	λ	8	4	2	4
Uncertainty	Ensemble Size		5		
	Dropout		0.5		

TABLE I
HYPERPARAMETERS FOR ALL METHODS.

Hyperparameters We set hyper-parameters consistently across settings, e.g. RoboCrowd and try to choose the same parameters for all methods when possible. Hyperparameters for all methods are shown in Table I.

Randomized k-NN Estimation. We estimate the mutual information using random batches for k-NN estimators. When doing so, we use a batch size of 1024 and iterate over the entire dataset 4 times.

Checkpoint Selection. We train all state-based models for 50K timesteps and all image-based models for 100K timesteps. For VAEs, BC policies, and VIP we select final checkpoints (e.g. 50K or 100K steps). We found that InfoNCE and MINE tended to overfit quite fast. For InfoNCE we used checkpoints after 20K for state and 40K for images. For MINE we used 50K for state and 60K for images.

D. Evaluation Details

RoboMimic For training policies in robomimic, we use the same architecture as in the image-based data quality experiments with an MLP action head using L2 loss. We train for 100K timesteps before running 200 evaluation episodes. Episodes are truncated after 400 timesteps.

RoboCrowd. We use an ALOHA robot setup to evaluate performance on the RoboCrowd benchmark, with ten trials per method. As in Mirchandani et al. [52] each trial assigned one of the following scores: 1 point for successfully grasping any number of candies, 2 points for returning any number of candies, and 3 points for returning exactly one candy. 0 points are given otherwise. Policies are trained for 200K timesteps using the same architecture ad hyperparameters from ACT, e.g. encoder-decoder transformer with L1 loss and action chunks of size 100.

Franka. We use the Franka robot setup from DROID [34], and run 15 trials per method. We score trails of "DishRack" as fully successful (1) if the robot puts both items in the dish rack, and a failure other wise (0). We score trials of "PenInCup" as fully successful (1) if the pen ends up completely in the cup a failure in any other scenario (0). We use the same architectures and hyperparameters as in Khazatsky et al. [34] for evaluation, e.g. Diffusion Policy [14] with action chunks of size 16 and execution size of 8 with a few differences. Instead of using pre-trained ResNet-50s as in Khazatsky et al. [34], we use ResNet-34s initialized from scratch with GroupNorm instead of BatchNorm. To compensate, we train for 200K steps as opposed to 50K.