Combining Unsupervised Data Collection with Expert Demonstrations for Offline Multi-Task Robot Learning

CS 330 Fall 2023 - Project Report [code][data]

George Hu, Jared Watrous, Jianhao Zheng

October 24, 2024

ABSTRACT

Offline reinforcement learning (RL) is the family of reinforcement learning approaches that uses a fixed dataset of trajectories and learns a policy without needing to interact with the environment. By separating the dataset and the learning algorithm, offline RL is an effective and interpretable method that can benefit from both data and inference improvements independently. In previous works, datasets used for offline RL have often relied upon online RL trajectories or expert demonstrations, but this approach can be expensive or fail to generalize to broader tasks. In this paper, we address these limitations by combining expert demonstrations with unsupervised data collection methods to explore the benefits of data scaling and data diversity in a multi-task and multi-goal setting.

We compare five different unsupervised data collection methods: random actions, Random Network Distillation (RND), Hindsight Goal-Conditioned Reinforcement Learning (GCRL), Active Pretraining with Successor Features (APS), and Contrastic Intrinsic Control (CIC). Using Meta-World, an open-source multi-task and meta-reinforcement learning robot arm simulator, we use each of these five methods to collect data on multiple task environments and goal positions. We then combine data from these methods with varying amounts of expert demonstration data and train an offline RL Implicit Q-Learning (IQL) model, evaluating the success rates on the corresponding environments. We further investigate how the training dataset composition, such as metrics relating to extrinsic rewards and state diversity affects the performance of downstream models.

Our results show that in the completely unsupervised setting, where offline RL datasets do not contain expert demonstrations, APS and CIC tend to collect more diverse and useful data compared to other methods. However, this is somewhat dependent on the environment, and overall the best data collection method, CIC, outperforms the random baseline by a 2.6% success rate in the multi-goal setting.

We also find that combining unsupervised data collection with sufficient expert demonstration data tends to significantly benefit the downstream offline model, with the increase in success rate correlating positively with the number of expert demonstrations used. However, in some scenarios, incorporating only very few expert demonstration trajectories actually reduced the performance of downstream models, and we believe a possible reason for this is unstable gradient updates when encountering insufficient expert demonstrations. But with just a modest amount of expert data we find our approach very useful in retaining information on both task generalization and specific demonstrations, achieving significantly higher performance than unsupervised-only baselines.

Moreover, we explore training a single multi-goal, multi-task model on all 10 environments using only the unsupervised data. While performance on all tasks is lower than when trained independently by environment, we find that performance degeneration varies among different data collection methods, with APS retaining the most performance.

By combining various unsupervised data collection methods and few-shot expert demonstrations for downstream Offline RL in a multi-task setting, we illustrate a novel method for using offline learning effectively in a variety of applications. With only a modest number of expert demonstrations and autonomously collected data that does not require supervised environment rewards, we demonstrate a path forward full of new possibilities for offline RL.

1 INTRODUCTION

Offline reinforcement learning (RL) is a method that differs from standard online RL by not allowing interaction with the environment, and instead assuming the existence of a dataset of trajectories and learning a policy based only on these trajectories. This technique has shown promising results, allowing models to leverage efficient training schema to learn complex policies Yarats et al. (2022), and it benefits from data scaling laws in line with supervised learning methods, along with inference completely disjoint from exploration, which is easier to analyze compared to online RL. However, an under-specified area in offline RL is how to create these datasets in the first place, and the performance of these models is highly dependent on the features of the data, including the collection method, state space coverage, and the sparsity of the rewards.

To overcome the challenges associated with this data dependency, many models have historically relied on expert demonstrations, training episodes in which an external controller entity demonstrates a successful completion of the task Yarats et al. (2022). Especially in the offline case, this helps fully decouple the exploration/exploitation trade-off, as the data collector no longer needs to concern itself with ensuring a high sample density near the goal state. Although this tends to work well in practice, it requires a manual data collection phase involving the influence of an (often human) expert, which may be difficult or impossible when many demonstrations are needed. This is especially true in a multi-task or multi-goal setting, where collecting multiple expert demonstrations for every possible task or goal may be infeasible or resource inefficient.

Various autonomous data collection approaches have been attempted Endrawis et al. (2021) and show success for generating useful datasets in offline RL, but a drawback of these approaches has been the apparent intractability for particularly difficult environments, such as robotics tasks combining multiple complex actions. To achieve the best of both worlds, we combine unsupervised data collection with few-shot expert demonstrations with the goal of data efficiency and generalization to multiple tasks. In this regard, we (1) experiment with different autonomous data collection methods across multiple environments in the multi-goal setting, and (2) examine the performance of each method when combined with few-shot expert demonstrations. We evaluate the quality of a collected dataset primarily by the success rate of an offline model trained on it, while also inspecting other intuitive metrics such as reward density.

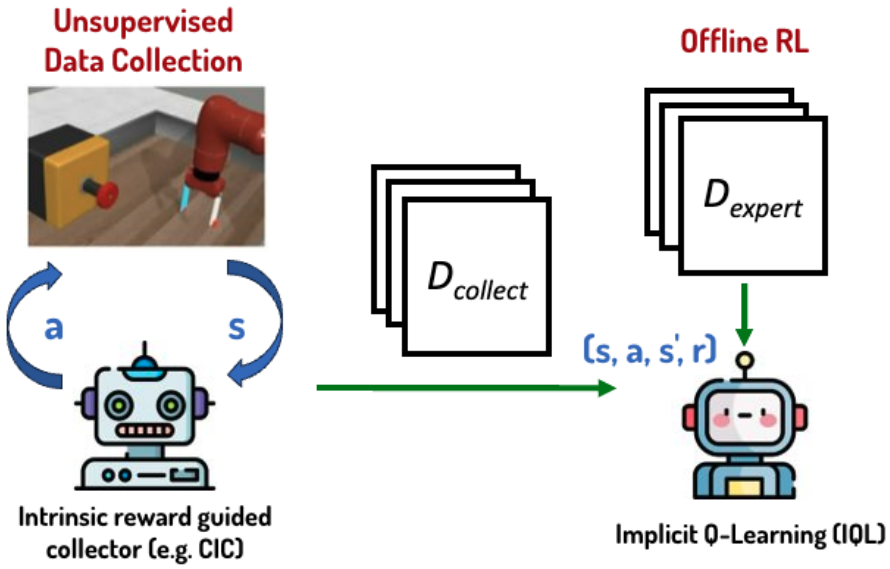

Figure 1: Illustration of our method for employing unsupervised data collection and expert demonstrations in robot learning tasks.

2 Related Work

2.1 Offline RL Challenges

Given that offline RL involves no environment interaction, the separation of data collection and inference upon such data is an explicit separation between exploration and exploitation, a common trade-off in online RL Yarats et al. (2022). Historically offline RL theory and application has focused only on the inference side, leaving data collection to be done ad hoc and aribtrarily from expert demonstrations or using the traces of online RL algorithms, but such an approach leaves a significant gap in dataset understanding Rashidinejad et al. (2023).

In particular, Rashidinejad et al. (2023) note how attributes of both expert data, trajectories that are collected from direct task demonstrations by experts, and uniform coverage data, data that deliberately covers all parts of the observation space, are important for offline RL algorithms. They propose theoretical frameworks to bridge this *data composition range*, and develop an offline RL algorithm that adapts to the type of data composition.

Another significant related challenge in data collection for offline RL is the multi-task and/or multi-goal setting. In many environments it is common to be able to collect expert demonstrations for a finite set of tasks and/or goals, but to ensure the data collected generalizes to any relevant task or goal is more difficult. Lambert et al. (2022) call the family of exploration methods in this setting as *task-agnostic exploration*, moddling their approach as online planning with exploration. The resulting Intrinsic Model Predictive Control method with Random Network Distllation (RND) Burda et al. (2018) and Dynamics Dissimilarity (DD) to measure curiosity in the data collection process. They then evaluate with offline RL inference on out of distribution tasks and goals, showing how task-agnostic methods can almost match the performance as task-aware methods.

2.2 Unsupervised RL

Unsupervised reinforcement learning encompasses the umbrella of all applications where an agent learns without using extrinsic environment rewards. Instead, a measure of intrinsic reward is passed into an online algorithm to guide the agent's control Laskin et al. (2021). Yarats et al. (2022) define three broad categories for intrinsic reward schema: **knowledge-based**, which minimizes the uncertainty of a predictive model, **data-based**, which maximizes the coverage of the observation space, and **skill-based**, which attempts to learn self-generated skill representations. Note that the more common application of unsupervised RL is to then finetune the same network in a supervised way using the extrinsic reward, and Laskin et al. (2021) find this approach of unsupervised RL as pretraining to be very successful.

Knowledge-Based Unsupervised RL Burda et al. (2018) introduce Random Network Distillation (RND), which uses a randomly initialized frozen target network $\hat{f}$ and student network f that encode states into some embedding dimension d. The intrinsic motivation for a state s is measured as

$$I(s) = ||f(s) - \hat{f}(x)||_2^2$$

The agent takes actions to maximize the sum of the extrinsic reward and I. The student network is trained with loss the same as the intrinsic motivation so that it emulates the target for states already visited.

Data-Based Unsupervised RL Endrawis et al. (2021) use goal-conditioned reinforcement learning (GCRL) as a form of intrinsic motivation, learning a goal conditioned policy $\phi(s, g)$ that attempts to bring state s to goal g. Goals are explicitly determined during learning by sampling a replay buffer and determining the highest novelty observation from RND. Endrawis et al. (2021) show that their method explores the whole state space more effectively than knowledge-based exploration policies.

By decoupling the process of evaluating novelty and locating states, their approach permits a more consistent training environment and yields more diverse data collection. Thus, they find that data-based unsupervised RL can provide uniform coverage data that can be applied effectively for downstream offline RL.

Skill-Based Unsupervised RL Skill-Based unsupervised RL methods generally involve maximizing mutual information between states visited and skills. In Active Pretraining with Successor Features (APS), Liu & Abbeel (2021) sample successor features w from a uniform random ball. These successor features are intended to represent a parameterization of latent skills, and are used in the Q function representation $Q(s, a, w) = \psi(s, a, w)^T w$ for critic network ψ. The intrinsic reward in APS is the mutual information objective

$$\max I(s; z) = \max H(s) - H(s|z)$$

The entropy of the states $H(s)$ is computed empirically through a particle filter, and the conditional entropy term $-H(s|z)$ is maximized by maximizing the lower bound $Q(s, a, w)$.

Laskin et al. (2022) do a similar approach in Contrastive Intrinsic Control (CIC) and instead maximize $I(\tau, z) = H(\tau) - H(\tau|z)$ where $\tau = [\phi(s), \phi(s')]$ is the concatenation of state encodings from ϕ. CIC also slightly differs from APS by directly sampling z from a uniform random ball, and they use the fact that the negative conditional entropy $-H(\tau|z)$ can be bounded from the cross entropy between τ and z, which can easily be calculated. Both APS and CIC are methods that thus allow policies to learn latent skills developed through environment interactions.

3 METHODS

3.1 DATA COLLECTION

We compare 5 different data collection methods, evaluating both the quality of the data using intuitive metrics, and the success rates of offline RL models trained on the data.

Random Actions As a baseline, our first method of data collection is taking random actions. In this method, we query our simulator environment for a random sample from the current action space and perform the sampled action. This approach involves no online learning during the data collection process.

Random Network Distillation We also construct a baseline for simple intrinsic motivation using Random Network Distillation (RND). Inspired by the approach outlined by Burda et al. (2018), we randomly initialize two neural networks that take a hindsight observation as input, and train one network to match the outputs of the other using a mean-squared-error loss. Actions are selected using Proximal Policy Optimization (PPO) as outlined in Schulman et al. (2017), trained on the loss between the RND models. We slightly modify the original implementation from Burda et al. (2018) by discarding extrinsic rewards and training only on the RND rewards, as well as removing the normalizer for the RND rewards. For this reason, we refer to this intrinsic motivation baseline as "Vanilla RND" in this paper.

Hindsight Goal-Conditioned Reinforcement Learning We implement the Hindsight Goal-Conditioned Reinforcement Learning (GCRL) data collection approach introduced by Endrawis et al. (2021). Empirically we found that the TD3 backend led to very poor exploration policies, so we differ from their method by instead using the Soft Actor-Critic (SAC) algorithm defined by Haarnoja et al. (2018). Moreover, whereas Endrawis et al. (2021) only uses being exactly in the same state as a boolean reward for the GCRL agent, we found that the exploration benefited from reward shaping based on the euclidean distance to the goal. That is, we have $r_{\text{intrinsic}}(s, g) = \mathbb{1}_{\{s=g\}} + \lambda||s - g||_2$ for some coefficient λ.

Active Pretraining with Successor Features We employ the approach for APS used in Liu & Abbeel (2021) closely, using the same DDPG Lillicrap et al. (2019) backbone with additional encoders for sucessor features. We empirically found that the best training schema was to train on batches sample uniformly from a small replay buffer. This mode of hindsight replay is so that the training is relevant to the current agent but not too uniform from the concurrent skill composition.

Contrastive Intrinsic Control For CIC Laskin et al. (2022), we use the same DDPG backbone as in the described approach, and do the same modification to the training schema as in APS.

3.2 Offline Learning

Implicit Q-Learning Our offline learning phase is identical for all training datasets, regardless of task or amount of expert data. We use Implicit Q-Learning (IQL), proposed by Kostrikov et al. (2021), which uses expectile regression to estimate Q-values without evaluating any state-action pairs outside the dataset. Specifically, we use the IQL implementation provided in d3rlpy, an open-source offline reinforcement learning library by Seno & Imai (2022).

4 Experiments

4.1 Meta-World Environment

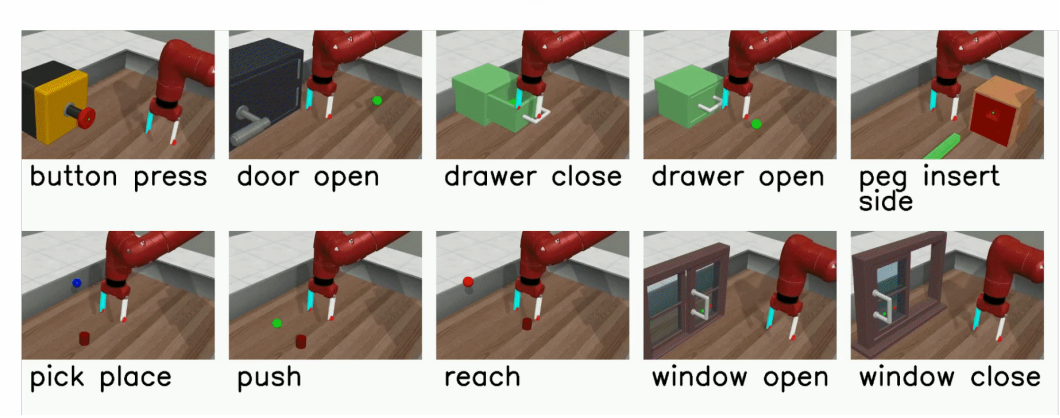

Figure 2: Sample renderings of the MT10 task environments from Meta-World Yu et al. (2019). These environments are intended for multi-task, mult-goal learning, with each task class having 50 variants with different goal positions.

We conduct all our experiments in the Meta-World simulator Yu et al. (2019). Meta-World is an open-source simulated benchmark for multi-task and meta-reinforcement learning consisting of 50 distinct robot arm manipulation environments. We focus our work on the MT10 subset of these environments, which consists of 10 task classes with 50 variants of each task class. Each task variant involves a unique goal state. See Figure 2 for sample a rendering from each task environment.

In Meta-World, the observation space $\mathbf{o}^t$ is represented as a vector of 39 components, arranged as

$$\mathbf{o}^t = [\mathbf{s}_e^t, \mathbf{s}_{o_1}^t, \mathbf{s}_{o_2}^t, \mathbf{s}_e^{t-1}, \mathbf{s}_{o_1}^{t-1}, \mathbf{s}_{o_2}^{t-1}, \mathbf{x}_{\text{goal}}] \in \mathbb{R}^{39}$$

where $\mathbf{s}_e^t$ denotes the state of the end effector at timestep t; $\mathbf{s}_{o_i}^t$ denotes the state of the ith object in the environment at timestep t; and x_{goal} is a task-dependent representation of the goal state, such as the target position. The detailed definition is as the following:

$$\begin{aligned}
\mathbf{s}_e^t &= [x_e^t, y_e^t, z_e^t, \delta_e^t] \in \mathbb{R}^4 \\
\mathbf{s}_{o_i}^t &= [x_{o_i}^t, y_{o_i}^t, z_{o_i}^t, \mathbf{q}_{o_i}^t] \in \mathbb{R}^7 \\
\mathbf{x}_{\text{goal}} &= [x_{\text{goal}}, y_{\text{goal}}, z_{\text{goal}}] \in \mathbb{R}^3
\end{aligned}$$

where x_k^t, y_k^t, z_k^t are the 3D Cartesian coordinates of object k at timestep t; δ_e^t measures how open the gripper is; and $\mathbf{q}_{o_i}^t$ is the Quaternion of the ith object at timestep t. If only one object is relevant to the current task, $\mathbf{s}_{o2}^t$ consists of all zeros.

To solicit changes in state, the controller model can perform an action with 4 numerical components, arranged as

$$\mathbf{a} = [\nabla \mathbf{x}_e, \tau]$$

where $\nabla \mathbf{x}_e \in \mathbb{R}^3$ is the gradient of the position of the end effector, and $\tau \in \mathbb{R}$ is the torque applied to the gripper.

Lastly, for each $(\mathbf{o}^t, \mathbf{a}^t)$ pair, the model receives a real-time reward signal $r \in [0, 10]$, which is uniquely defined for each task.

4.2 UNSUPERVISED DATA COLLECTION

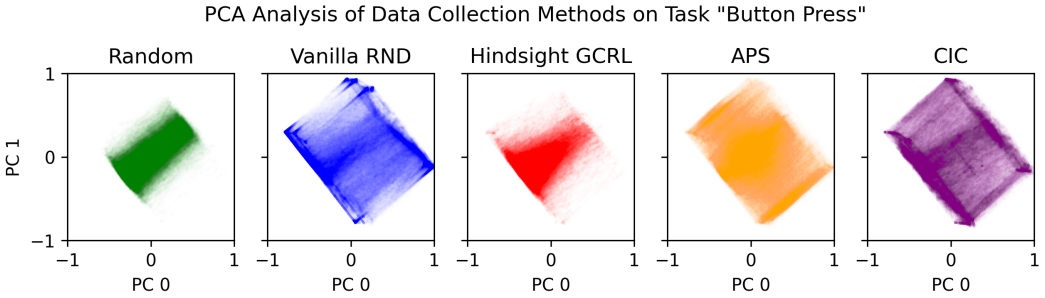

Figure 3: A PCA analysis of data collected by the 5 methods on the task "button press." For this visualization, we compute the principal components of the entire set of observations $\mathbf{o}^t \in \mathbb{R}^{39}$ from all data collection methods. We then project the observations from each method independently onto the two largest principal components, labeled here as "PC 0" and "PC 1." We postulate that the 3-dimensional components corresponding to spatial positions contribute to the apparent cubic shape of the projected observations, as the physical bounds of the simulator form a rectangular prism.

For each Meta-World task, we collect training data using the unsupervised data collection methods introduced in the previous sections. For each of the 50 possible goal locations, 40 trajectories of 500 steps are collected, for a total of 1 million data samples per environment. A visualization of the collected data can be found in Figure 3.

Because the plotted observation vectors $\mathbf{o}^t$ include 3-dimensional position attributes, we anticipate that the locations of points in the visualization reasonably correspond to physical locations within the spatial bounds of the Meta-World simulator, and that the apparent density of points in Figure 3 correlates with the density of samples at that physical location.

Under the above assumption, we observe that Hindsight GCRL collects state samples that somewhat resemble random actions and are much less uniformly distributed than Vanilla RND, APS, or CIC. We also note that Vanilla RND tends to concentrate in the corners of the state space, which may indicate policy collapse due to the continuously changing reward landscape. By qualitative observation alone, APS and CIC appear to yield the most uniformly distributed datasets, with only slight bias towards the center of the state space, where the button is often located. It is difficult and unreasonable to determine the quality of the data from visualizations alone, so we turn to more fundamental metrics for a more comprehensive evaluation of each data collection method.

4.3 PERFORMANCE OF OFFLINE RL

We use IQL Kostrikov et al. (2021) to train an offline RL model on the collected data. For each task and the collected data, we train the model for 5×10^5 steps with a mini-batch size of 256, actor learning rate of 3×10^{-4} and critic learning rate of 3×10^{-4}.

4.3.1 UNSUPERVISED COLLECTED DATA

	Random	Vanilla RND	Hindsight GCRL	APS	CIC
Avg Success Rate	58.4%	50.8%	49.2%	60.2%	**61%**

Table 1: Success rate of offline IQL models trained on data collected by each method, averaged across the 10 task environments.

As a baseline for our experiments, we first train offline IQL models for each task environment with only the data collected by our five unsupervised methods. We evaluate these models by their success rates across all 50 variants of task environments, see Figure 4. Although models trained by data collected by all five methods can hardly succeed in some complex tasks such as "pick place", "peg

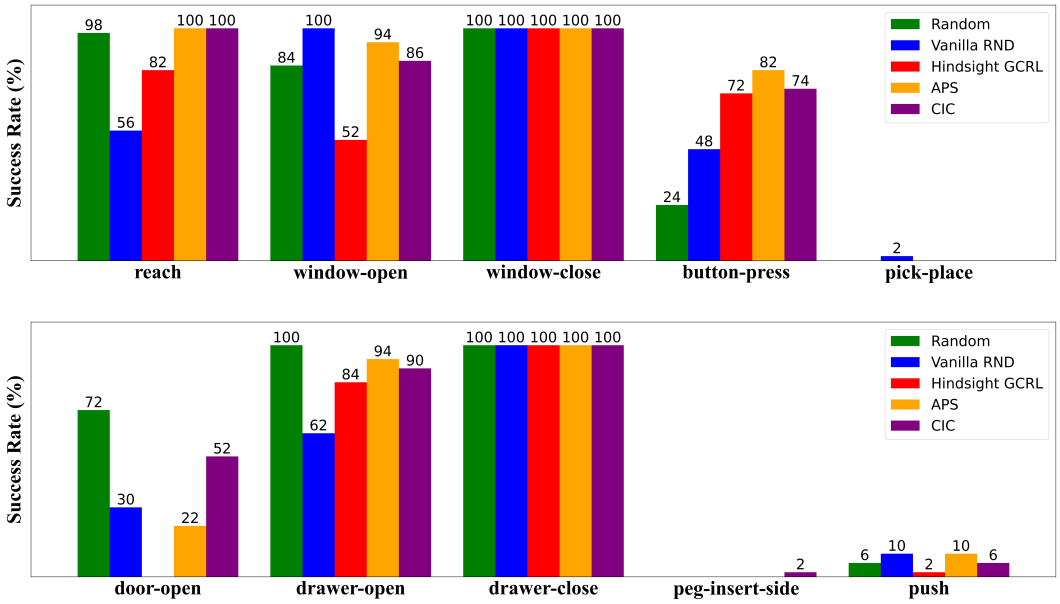

Figure 4: Success rate of model trained with unsupervised data collection.

insert side", and "push", data collected by CIC and APS can produce a higher success rate in most of the remaining tasks. In addition, CIC achieves the most stable performance on the 7 easier tasks, while the other 4 methods produce a significantly low success rate on specific tasks. For example, Random only achieves 24% success rate on "button press".

In Table 1, we list the average success rate of each method in the 10 task environments. CIC achieves the best performance overall. APS is slightly worse than CIC, but still performs better than Random exploration. Vanilla RND and Hindsight GCRL have the lowest overall success rates, below even our random exploration baseline.

4.3.2 Incorporating Expert Demonstrations

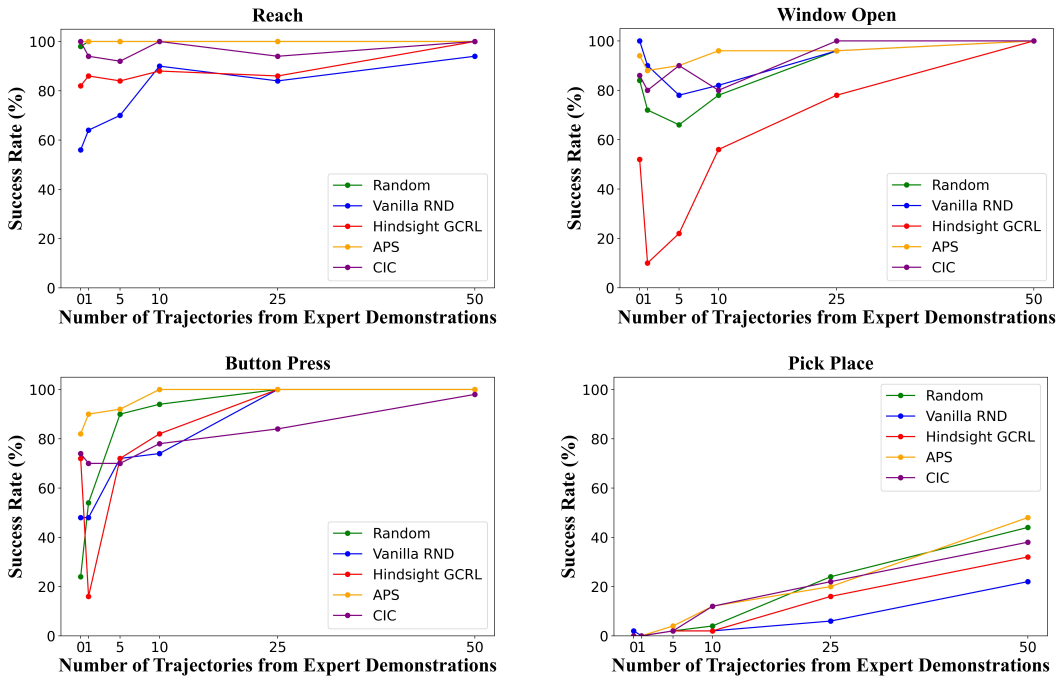

Figure 5: Success rate of the model trained with data incorporated by different numbers of trajectories from expert demonstration data.

One of the primary goals of our study is to evaluate how the amount of expert demonstration data affects the performance of downstream offline models. To this end, we combine the data collected from our 5 methods with varying numbers of expert demonstration trajectories. We conduct experiments on four representative environments: "reach", "window open", "button press" and "pick place". Figure 5 shows the success rates of offline IQL models trained with 1, 5, 10, 25, and 50 expert demonstrations, as well as the same unsupervised-only baseline model described in the previous section.

In general, APS outperforms other methods when combined with expert demonstration data. On the "button press" and "window open" tasks, APS combined with just 10 expert demonstration trajectories can reach a nearly 100% success rate, while most other methods require 25 or even 50 trajectories. On "pick place," one of the most difficult tasks, combining the unsupervised data with expert demonstrations can allow the trained models to achieve non-zero success rates. Combined with 50 expert trajectories, APS can produce the highest success rate on this complex task.

We can further observe that, when combined with sufficiently many expert demonstration trajectories, virtually all models can achieve an equivalent or higher success rate than unsupervised-only models, and in most cases this increase correlates with the number of expert demonstrations. However, one major unexpected result is that incorporating expert demonstration data can significantly harm the performance of models in the very few-shot case. For example, on the task "window open," the model trained on Hindsight GCRL achieves a 52% success rate using only the unsupervised data, but plummets to 10% when combined with only a single expert demonstration trajectory. To consider a possible cause for this counter-intuitive relationship, we recall how our unsupervised data contains 10^6 total steps of relatively low extrinsic reward uniform exploratory data, whereas each expert demonstration is 500 steps of high extrinsic reward and explicitly directed trajectories.

When we train the IQL model on these two different sources of data, the gradient updates between unsupervised and expert data are inherently different. Relating to the concepts in Rashidinejad et al. (2023), we can view few-shot expert data as out-of-distribution compared to the learning process conditioned on the significantly more numerous unsupervised data. In general Rashidinejad et al. (2023) find that mixing these data sources is fine as long as the distribution from both sources is well-represented and/or the algorithm explicitly allows for differing data source inputs during training. So for the case of insufficient expert demonstrations, the training algorithms receives an out-of-distribution signal that is not well represented across many goals, and the learning process is thus unstable and can lead to poor policy performance on some goals. In contrast when we have enough expert demonstrations, the model can incorporate the complete distribution of diverse expert trajectories and properly combine the expert and unsupervised data to create generalizable policies.

4.3.3 Ablation Study: Multi-task Learning

In this ablation study, we train a single multi-goal, multi-task model with data collected from each unsupervised method on all 10 task environments. We encode the environment by concatenating its integer index, in the range 0 to 9, to the observation as the input to the model.[1]

The success rates of the multi-task models on each environment are illustrated in Figure 6, and the average success rates across all tasks are reported in Table 2. We can observe that the performance of all 5 methods is worse than that of models trained independently by environments. Most multi-task models even fail on environments where an individual model can achieve a high success rate (*e.g.* "window open", 'window close", "door open", and "drawer open"). Two exceptions are models trained on data collected by APS and CIC, whose average success rates outperform other methods by a clear margin, see Table 2. APS still achieves a high success rate on the environment of "window open" and "window close", while CIC outperforms all other methods on "window close" and "drawer open". Data collected by Vanilla RND and Hindsight GCRL is less efficient to train a multi-task model as their success rate on most tasks are extremely low.

[1]In a separate experiment, we trained the same model using a one-hot encoding of the environment. We found that directly using an integer index, despite the task index having no real metric relationship, resulted in significantly better performance, so we report those results in this paper.

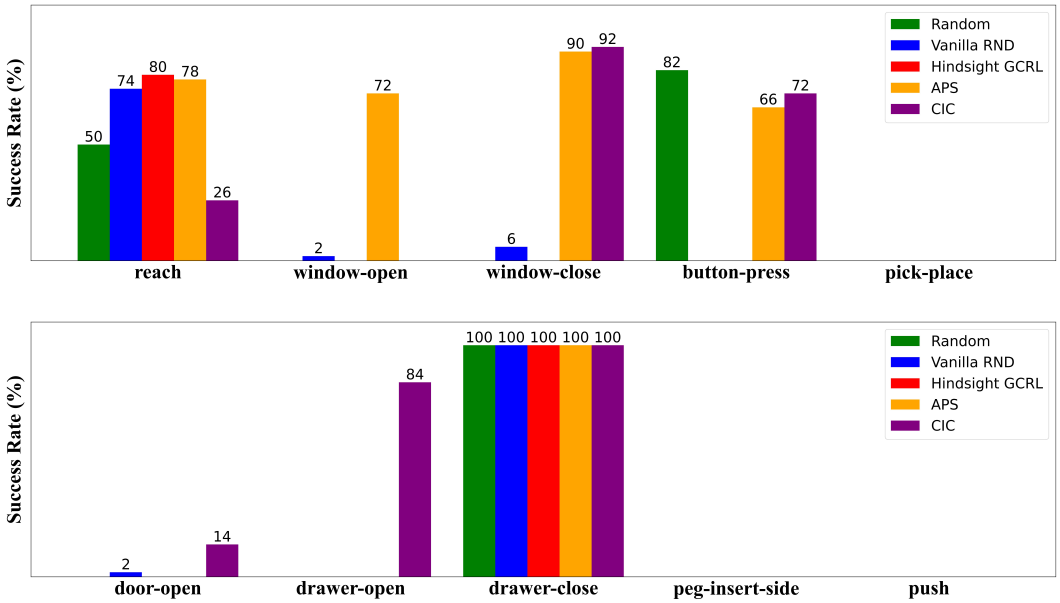

Figure 6: Success rates of multi-goal, multi-task models trained on unsupervised data from all 10 tasks. One model is trained and evaluated for each data collection method.

	Random	Vanilla RND	Hindsight GCRL	APS	CIC
Avg Success Rate	23.2%	18.4%	18.0%	**40.6%**	38.8%

Table 2: Average success rates of multi-task, multi-environment models trained on unsupervised data from all 10 tasks. One model is trained and evaluated for each data collection method.

4.4 Discussion

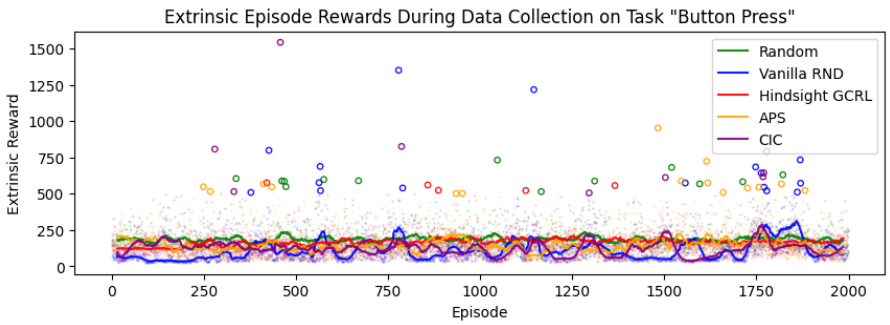

Figure 7: Extrinsic total episode rewards encountered during different data collection approaches. The 30-episode moving average is highlighted, and the episode rewards greater than 500 are the emphasized hollow dots.

In Figure 7, we plot the total episode extrinsic reward each unsupervised data collection algorithm encounters during the exploration process for "Button-Press". We notice that the most represented high episode rewards are for Vanilla RND, APS, and CIC. When taking a look at the downstream performance of these methods without expert data in Figure 4, we can see that APS and CIC perform well, but not Vanilla RND. We suspect that part of the Vanilla RND poor performance can be seen in the amount of extrinsic episode rewards being close to 0, as indicated by the blue line being at the bottom of the plot. While seeing very low reward can be useful to determine incorrect paths in the downstream offline learning process, having a significant proportion of trajectories with very low

reward here is likely indicative of poor exploration with unwanted clustering. This is confirmed by Figure 3 as well.

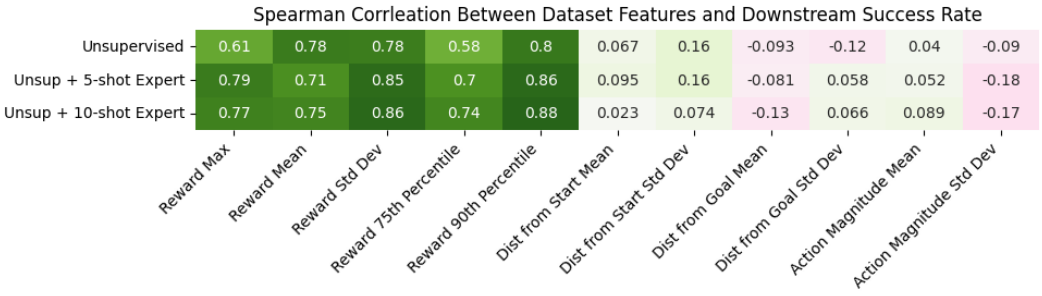

Figure 8: Spearman correlation ρ between aggregate dataset extrinsic reward, state, and action statistics, and the resulting IQL model success rate over all tasks and all data collection methods.

Moreover, we analyze the Spearman correlation between dataset statistics and offline RL success rate in Figure 8. We can see that extrinsic reward measures correlate very well with downstream performance for all data collection methods, environments, and expert data settings. In particular, the standard deviation of rewards and 90th percentile of rewards have the strongest correlations, suggesting that containing diverse rewards along with having a top 10th percentile of sufficiently high reward steps is important for downstream success. These two metrics broadly relate to data diversity and sufficiently successful trajectories, desirable dataset attributes as explained in Rashidinejad et al. (2023).

Of note is also the negative correlations for mean distance to the goal position and action magnitude. This informs that trajectories too far from the goal are likely not useful, and that exploration policies with actions that vary too greatly can lead to poor training data, so future applications would need to keep this in mind.

5 Conclusion

We compare five different unsupervised data collection methods by measuring the performance of downstream offline RL models. Our result shows that CIC achieves the best performance when no expert demonstration is available. We further find that augmenting unsupervised data collection with expert demonstrations can be beneficial in a multi-task and multi-goal setting, on the condition that the expert data maintains a relatively unbiased distribution over the target tasks. An ablation study on training a multi-task model with only the unsupervised data shows that such a model performs worse than training a single model per each task, regardless of the data collection method used. The multi-task models trained by CIC and APS data perform significantly better than other methods, indicating that the data collected by these two methods are more generalizable to multiple tasks and goals.

6 Contributions

George: I implemented and ran data collection scripts for Hindsight GCRL, APS, and CIC. I helped with running training and evaluation for a few of the offline RL evaluations, and significantly contributed to the ideation of the project in general. I also generated the extrinsic reward and dataset feature correlation plots and helped with creating the poster and report.

My contributions differed from the proposal in that the Endrawis et al. (2021) approach did not work that well, so I modified hindsight GCRL to use SAC and also implemented the APS and CIC methods in addition. Moreover, we focused on multi-task and multi-goal generalization rather than transfer learning as in the proposal.

Jared: I implemented and ran the data collection script for Vanilla RND. I also produced the PCA visualization for the unsupervised data, as well as helped run training for some of the offline IQL models and helped create the poster and report.

Unlike the proposal, which suggested the use of a pretrained model for data collection, our final project did not involve any transfer learning, so no such pretrained model was needed. Furthermore, we found that TD3 performed poorly as the backend model for Vanilla RND, so I implemented the PPO algorithm from Schulman et al. (2017) to replace it.

Jianhao: As planned in the proposal, I set up the Meta World simulator. I implemented offline IQL training, as well as the evaluation of the trained RL models. I ran most of the offline IQL training and evaluations, and produced the plots related to the success rate of the trained model. I also helped to make the poster and the report.

In addition to the planned proposal, we decided to further study the effect of multi-task RL. Therefore, I also implemented and conducted the multi-task IQL training and evaluation. I also conducted experiments of using one-hot encoding to concatenate the task condition, although we didn't report its result since it performs worse than index encoding.

REFERENCES

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation, 2018.

Shadi Endrawis, Gal Leibovich, Guy Jacob, Gal Novik, and Aviv Tamar. Efficient self-supervised data collection for offline robot learning, 2021.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, 2018.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

Nathan Lambert, Markus Wulfmeier, William Whitney, Arunkumar Byravan, Michael Bloesch, Vibhavari Dasagi, Tim Hertweck, and Martin Riedmiller. The challenges of exploration for offline reinforcement learning, 2022.

Michael Laskin, Denis Yarats, Hao Liu, Kimin Lee, Albert Zhan, Kevin Lu, Catherine Cang, Lerrel Pinto, and Pieter Abbeel. Urlb: Unsupervised reinforcement learning benchmark, 2021.

Michael Laskin, Hao Liu, Xue Bin Peng, Denis Yarats, Aravind Rajeswaran, and Pieter Abbeel. Cic: Contrastive intrinsic control for unsupervised skill discovery, 2022.

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning, 2019.

Hao Liu and Pieter Abbeel. Aps: Active pretraining with successor features, 2021.

Paria Rashidinejad, Banghua Zhu, Cong Ma, Jiantao Jiao, and Stuart Russell. Bridging offline reinforcement learning and imitation learning: A tale of pessimism, 2023.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

Takuma Seno and Michita Imai. d3rlpy: An offline deep reinforcement learning library, 2022.

Denis Yarats, David Brandfonbrener, Hao Liu, Michael Laskin, Pieter Abbeel, Alessandro Lazaric, and Lerrel Pinto. Don't change the algorithm, change the data: Exploratory data for offline reinforcement learning, 2022.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning (CoRL)*, 2019. URL https://arxiv.org/abs/1910.10897.