COMBOU: Leveraging Unlabeled Data in Conservative Offline Model-Based RL

Stanford CS224R Final Project

Haozhuo (Tommy) Li
Department of Statistics
Stanford University
tommy01@stanford.edu

Linan (Frank) Zhao
Department of Mathematics
Stanford University
frankz24@stanford.edu

Abstract

Due to the tremendous cost in rolling out polices and collecting data in environments, offline Reinforcement Learning (RL) algorithms have emerged to use static data sets for training. Specifically, as offline data set can be limited, a family of offline RL algorithms aims to use a dynamics model to generate synthetic data in the environment and incorporate the synthetic data into training. This family is called model-based offline RL. A recent work proposes a new model-based offline RL algorithm, COMBO, that induces conservatism in evaluating real and synthetic data. COMBO has shown promising results in standard offline RL benchmarks and tasks that requires generalization. However, model-based offline RL algorithms, including COMBO, mostly assumes access to a large labeled data set with reward. But in real settings, especially when collecting data in the real world, it is common to have only limited expert data labeled with rewards that targets a task. Besides, there are abundant noisy, unlabeled, and task-irrelevant data in the same environment that is mostly left unused. However, the unlabeled data can provide valuable information on the environment dynamics. Therefore, in this paper, we propose several methods to incorporate the unlabeled data set into the training pipeline for model-based offline RL. And we specifically integrate these methods into COMBO and test their performance on two standard offline RL benchmark tasks (Hopper and Walker2D) from D4RL. The evaluation results show that our methods are able to reach a better performance with 10k expert labeled data and 1 million random unlabeled data than the standard COMBO algorithm with 2 million of expert and medium labeled data. This suggests that our methods successfully learn a better dynamics and conservatism during training by adding in unlabeled data. Further, our proposed approach achieves more robust results than the recent unlabelled data sharing (UDS) method. This framework has potential to perform well in many other offline RL settings like multi-task learning and meta learning, which can be explored in future studies.

1 Introduction

Reinforcement learning (RL) mainly consists of an agent acting in an environment which contains a state space and an action space. The agent acts according to a policy $\pi(a|s)$ that predicts the next action given the current state. The qualify of a particular action is judged by the reward r. The goal of reinforcement learning is to learn a policy π that maximizes the expected reward over time.

In reinforcement learning, the agent is intuitively learning from experience. It can try a series of actions, see the results (rewards), and adjust its behavior based on the results. The bottle-neck in this process is often in rolling out policies in the environment and collecting data, especially in real-world environments. Therefore, a family of RL algorithms is developed that learns from static data sets with reward annotation. This family of approaches is called offline RL.

More formally, standard RL is modelled as a Markov decision process, $M = (S, A, P, \gamma, R)$, where S and A denote the state space and action space, $P(s'|s, a)$ models the state dynamics, $\gamma \in [0, 1]$ is the discount factor, and R corresponds to the reward function that measure the quality of an action at a state. Offline RL tackles this problem of learning a policy $\pi(a|s)$ using a static labeled data set L, usually generated by a behaviour policy $\pi_\beta(a|s)$ of varying qualities (Yu et al. (2022a)). Most RL algorithms assumes the existence of the reward function R and attempts to learn the state dynamics P from the labeled data set L.

However, in many cases, the assumption of having a large labeled data with reward annotation is too strong. A much more realistic case is when we have a small amount of high-quality labeled task-specific data with a large amount of unlabeled task agnostic data that does not contain reward information. In many cases, labeling reward is costly especially if the reward needs to be annotated by humans. But collecting random trajectories from an environment is often inexpensive and fast. For example, if we want to train a robot to open and close a microwave, we might obtain some data of the robot attempting the task under the supervision of a human manipulator. But we are more likely to have much more background data that are either structurally related or provides information about the environment. For instance, we might have more data of the robot opening and closing other objects like a door, or a drawer. These data sets wouldn't be labeled with the right reward annotation specific to opening and closing microwaves but they can still be leveraged for training the agent. This setup raises the important question of how we can utilize unlabeled data for training offline RL agents.

Currently, there exists plenty of offline RL algorithms. But they all require the offline data sets to be labeled. In this case, the vast amount of unlabeled data is left unused. However, although unlabeled data sets do not contain reward, they provide valuable information about the world dynamics. Therefore, in this paper, we aim to incorporate unlabeled data in offline model-based RL, specifically in Conservative Offline Model Based Policy Optimization (COMBO) by Yu et al. (2022b), which will be discussed below. To simulate the scenario of having a small labeled data set and a large unlabeled data set, we use a combination of expert data and random data from the Walker2D and Hopper environments from D4RL (Fu et al. (2021)). We will start with a literature review of prior works, experiment with some existing approaches, propose a novel method, and analyze their effectiveness.

2 Related Work

2.1 Leveraging Unlabeled Data

As mentioned, offline RL considers the challenge of learning a good policy from a static data set without the need to roll out policies in the environment. It has achieved promising results in numerous domains like NLP (Jaques et al. (2019)), healthcare (Wang et al. (2018)), and robotics (Rafailov et al. (2021)). The main issue with a naive offline RL algorithm is usually the distributional shift between the learned policy and the behaviour policy (Yu et al. (2022a)). For instance, many offline RL algorithms will overestimate value functions on out-of-distribution (OOD) transitions from the data set, biasing the agent to chase unreliable OOD actions. To tackle this problem, many prior works have tried to constrain the learned policy in some way. Some of these approaches include policy regularization (Ran et al. (2023)), conservative value functions like CQL (Kumar et al. (2020)), and model-based training with conservative penalties like COMBO (Yu et al. (2022b)). However, all these prior works assume access to a large labeled data set. Here, we want to study how unlabeled data can be leveraged into existing offline RL framework.

In using unlabeled data for offline reinforcement learning, prior works have attempted to tackle this problem via directly imitating expert trajectories like in adversarial imitation learning frameworks (GAIL) (Ho and Ermon (2016)), indirectly learning reward functions from expert data using inverse RL (Ng and Russell (2000) Konyushkova et al. (2020)), or learning a reward classifier that discriminates successes and failures (Eysenbach et al. (2021)). Moreover, Xu and Denil (2019) argued that reward learning can lead to agent exploiting errors in the reward model to achieve high reward behaviors that do not correspond to intended task. These reward delusions are common in reward-prediction methods and can lead to unintended and even dangerous behaviours. On the other hand, GAIL and discriminatory methods tend to suffer the opposite problem, where the discriminator learns to trivially distinguish agent and expert behavior, in effect resulting in a reward prediction that produces low reward regardless of the input state. Because of these reasons, Xu and Denil (2019) proposed to use positive-unlabeled learning, which concerns the problem of learning classifiers from positive data and unlabeled data to train an accurate reward model. These reward prediction methods have succeeded to varying degrees.

However, Yu et al. (2022a) showed that these complicated reward prediction methods may be unnecessary. They showed that simply setting the reward of all unlabeled data to zero (UDS) is effective both in practice and in theory. Yu et al. (2022a) proved that UDS guarantees policy improvement and proposed a variant called conservative data sharing (CDS) by reducing reward bias through re-weighting data. Formally, UDS trains any offline RL method on an effective data set given by $D^{eff} = L \cup \{(s_j, a_j, s'_j, 0) \in U\}$. In fact, UDS outperforms sophisticated reward prediction methods in many single task and multi-task settings. Our paper is a natural extension of the UDS method but integrates this line of thought naturally into model-based conservative learning approaches. Basically, we propose to train a reward prediction model based on expert data and then fill in the reward predictions for unlabeled data. During training, we build in pessimism on these reward predictions by pushing down the value functions on unlabeled data. We will show that this simple approach is very effective in practice and may surpass the UDS method. Intuitively, these two methods are very similar in principle. UDS can be seen as forcing the worst-possible pessimism with regard to the unlabeled data set as all rewards are marked as zero. However, our approach still builds in pessimism on unlabeled data points but makes the process more dynamic and adaptive.

Further, although we focus on single-task environments in this project for simplicity, data sharing across different tasks has been found to be effective as well. It improves performance drastically in multi-task settings (Kalashnikov et al. (2021) Yu et al. (2021)) and in meta-RL settings (Dorfman et al. (2021)). We hypothesize that our proposed method will also work well in these settings but leave the experiments to future works. Nevertheless, these papers show the importance of leveraging unlabeled data for offline reinforcement learning.

2.2 Conservative Offline Model Based Policy Optimization (COMBO) (Yu et al. (2022b))

As discussed above, a main paradigm shift from online to offline RL is to incorporate conservatism or regularization to prevent over-estimation on value functions. Model-free offline RL directly adds conservatism into model training processes, which only learns from offline data set and can be too conservative and fail to generalize. In contrast, model-based offline RL first learns a dynamics model based on offline data set which can generate synthetic data. The training process then leverage both the offline data and synthetic data, conservatively estimating the value functions for each state-action data point. Therefore, model-based algorithms have a higher potential for generalization and solving new tasks. However, all previous model-based offline RL algorithms rely on some uncertainly estimation in conservative evaluation such as in Yu et al. (2020b) and Kidambi et al. (2021). But in practice, these uncertainty estimation methods can be unreliable.

So the COMBO paper proposes a new model-based offline RL algorithm COMBO, which doesn't need access to uncertainty estimation oracles. Instead, it employs actor-critic method using both offline and synthetic data while penalizing Q-values of state-action tuples from synthetic data. In addition, the paper theoretically proves that the Q-function learned by COMBO is a lower bound on the true Q-function, and it is less conservative than Q-functions learned by model-free counterparts such as CQL (Kumar et al. (2020)) as seen in Figure 1. Finally, the paper experiments COMBO on three kinds of tasks: (1) tasks that require generalization, (2) tasks with high-dimensional image observations, and (3) standard offline RL benchmarks. The empirical results show that COMBO out-performs prior methods in most tasks and can generalize to high-dimensional vision-based tasks.

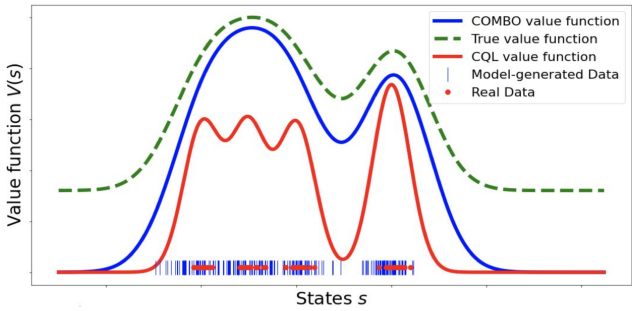

Figure 1: Comparing the estimated value function from COMBO (model-based) with CQL (model-free). COMBO learns a conservative value function but is less conservative than CQL.

As a promising offline RL algorithm, COMBO will act as the baseline in our study. We will try to improve its performance by exploring different ways of leveraging unlabeled data into the pipeline of COMBO. We will discuss this process in detail in the next section.

3 Approach

3.1 Data sets

D4RL (Fu et al. (2021)) is an open-source benchmark for offline RL. It provides standardized environments and data sets for training and evaluating offline RL algorithms. In this paper, we use two tasks **Walker2D** and **Hopper** in D4RL to experiment our proposed methods. These two environments are based on oepneAI's Gym-Mujoco tasks that are wildly used by prior works in offline RL like by Fujimoto et al. (2019) and Wu et al. (2019).

Specifically, the agent in Walker2D is a two dimensional twp-legged figure with four main body parts. And the main goal is to coordinate all body parts to move in the forward direction by applying torques on different hinges. The agent in Hopper is a two-dimensional one-legged figure with four main body parts. Its goal is to apply torques to hinges that makes it hop in the forward direction (Durrant-Whyte et al. (2012)). Note that the Walker2D environment is somewhat an extension of the Hopper environment and is a bit more difficult. We present two images of the two environments below for visualization.

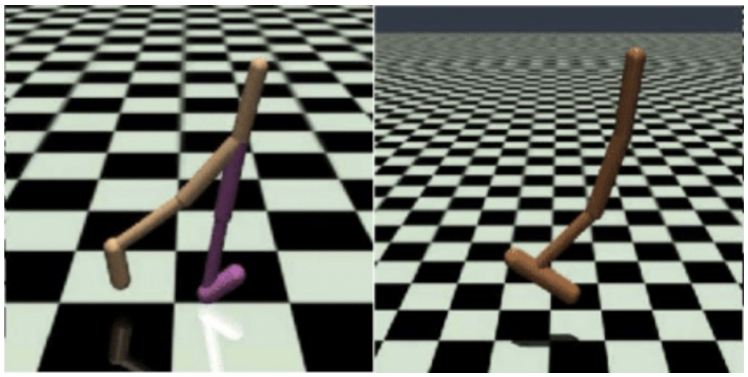

Figure 2: The Walker2D (left) And Hopper (right) Environments.

Note that the data in D4RL are all labeled with rewards. But our problem setting consists of labeled expert data and irrelevant unlabeled data. To resemble this setting, for each task, we use 10k labeled expert data $(s, a, s', r) \sim \mathbf{L}$ from a policy trained with SAC to completion and 1 million random unlabeled data from a random policy with rewards removed $(s, a, s') \sim \mathbf{U}$.

In addition, the reward calculation in our data sets is always normalized, where 0 represents the worst possible reward from randomized data and 100 represents the best possible reward from expert data. Note that it is possible to exceed 100 if the learned policy is even better than the provided policy. Having detailed our data sets, we can now discuss our main approaches.

3.2 Incorporating unlabeled data set into COMBO

Let's first clarify the notations in this problem setting. We are given a labeled data set **L** and an unlabeled data set **U**. And the dynamics model is decoupled into reward prediction and next-state prediction: **R(r | s, a)** and **P(s' | s, a)** respectively. As we are using COMBO as a building block for our methods, we will state the COMBO algorithm here for future reference:

Algorithm 1 Conservative Offline Model Based Policy Optimization (COMBO)

1: Train dynamics model **R** and **P** using **L**
2: **while** training **do**
3: Roll out dynamics model for model data **M**
4: Conservatively evaluate critics:
5: $Q^\pi := argmin_Q \mathbb{E}_{(s,a,s',r)\sim L\cup M}[(Q(s,a) - (r + \gamma\mathbb{E}[Q(s',a')]))^2] +$
$\alpha\mathbb{E}_{(s,a)\sim M}[Q(s,a)] - \alpha\mathbb{E}_{(s,a)\sim L}[Q(s,a)]$
6: Improve policy π based on updated critics
7: **end while**

The intuition behind the COMBO algorithm is that it bakes in pessimism with respect to the model-generated data M. It pushes down on the predicted Q-values on M while pushing up on the predicted Q-values on real labeled data. This allows COMBO to conservatively model the value function using both the offline data set and the simulated data set from the model and prevents overestimation on out-of-distribution states. On a higher level, COMBO consists of 3 parts:

1. Dynamics (state and reward) Training
2. Critics Training
3. Conservative Policy Evaluation

This project explores different ways to incorporate unlabeled data into these three parts. Based on the usage of unlabeled data, we proposes the following methods:

1. **Baseline 1 (COMBO with no data sharing):**
   - Run COMBO on labeled data set L only.
2. **Baseline 2 (naive reward prediction):**
   - Use L and U to train dynamics model to predict next state $P(s'|s,a)$.
   - Use L alone to train a reward model R and use R to fill in the rewards for data in U.
   - Run COMBO on L and U.
3. **Variant 1 (only use unlabeled data for training state dynamics):**
   - Use L and U to train dynamics model to predict next state $P(s'|s,a)$.
   - Use L alone to train a reward model R.
   - Run COMBO on L alone.
4. **Variant 2 (UDS):**
   - Set the reward of all unlabeled data in U to 0.
   - Then combine L and U to run COMBO on them.
5. **Variant 3 (reward prediction with built-in pessimism):**
   - Use L and U to train dynamics model to predict next state $P(s'|s,a)$.
   - Use L alone to train a reward model R and use R to fill in the rewards for data in U.
   - Run COMBO on L and U with built-in pessimism on U ($\mathbb{E}_{(s,a)\sim L\cup U}$):

3.3 Experimental Details

We balance the labeled and unlabeled data sets during dynamics training by up-sampling. Otherwise, unlabeled data set completely dominates and the dynamics model would learn noisy transitions. For all models, we use the Adam optimizer for optimization. When training the critics with unlabeled data, a batch contains 1/4 model data, 1/4 unlabeled data, and 1/2 labeled data. If not using unlabeled data in training critics, we use a even split between model and labeled data. And all model backbones are MLPs that follow the COMBO paper.

4 Result

First, we present the average evaluation reward during training for both environments. Since reinforcement learning is often highly stochastic, we present these training curves using smoothed data points from tensorboard.

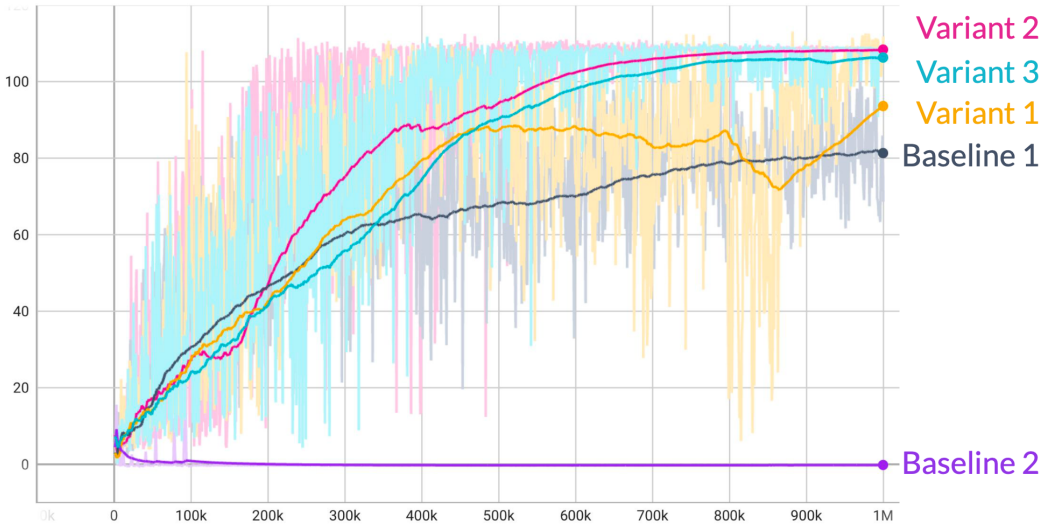

Figure 3: Average evaluation reward during training for Walker2D environment for all five methods.

From this figure, we can see that both UDS (Variant 2) and reward prediction with built-in pessimism (Variant 3) are very effective in leveraging unlabeled data set. They present a large increase from the baseline method that relies solely on labeled data. Further, naive reward prediction and labeling (Baseline 2) does not work at all in this environment. This aligns with the conclusion from the UDS paper by Yu et al. (2022a). Lastly, using unlabeled data $\mathbf{U}$ only for training the state dynamics (Variant 1) is useful in itself as it surpasses the baseline 1 method. However, it is more stochastic and its performance is still worse than other two variants.

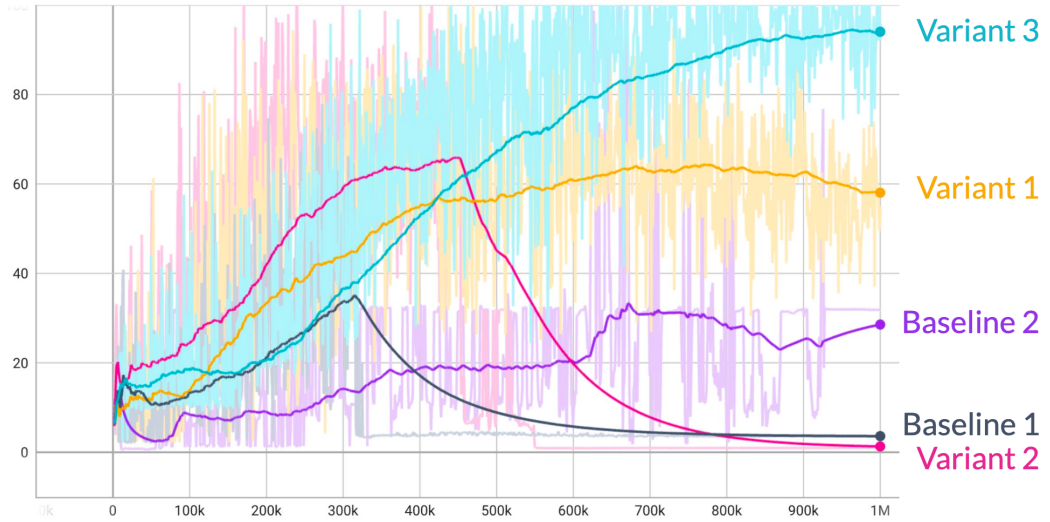

Figure 4: Average evaluation reward during training for Hopper environment for all five methods.

This figure of the training curves from the Hopper environment is more stochastic. In particular, we see that the two curves for baseline 1 and variant 2 both suddenly decreases in the middle of training. We hypothesize that this is due to model over-fitting and thus we ignore the data points after the

peak for evaluation. Realistically, we can easily incorporate some form of early stopping to avoid this over-fitting behaviour. Nonetheless, we can see that variant 3, which is reward prediction with built-in pessimism works extremely well and is almost triple the performance of baseline 1. Using UDS also performs quite well but still falls behind variant 3 significantly. Again, we can confirm that using unlabeled data set for training state dynamics alone is useful while naive reward prediction tends to over-estimate and performs poorly.

Next, we summarize the evaluation results in the table below. For the evaluation metric, we use the average normalized evaluation episode reward in the last 100 training epochs, where the policy has likely converged. And as mentioned above, we ignore the data points after the sudden decrease in performance in the Hopper environment.

Table 1: *Evaluation results*

	Walker2D	Hopper
Baseline 1	82.372 ± 10.187	39.918 ± 16.656
Baseline 2	-0.167 ± 0.017	29.653 ± 9.960
Variant 1	100.794 ± 9.877	55.355 ± 9.292
Variant 2	$\mathbf{108.534} \pm 1.611$	66.466 ± 16.027
Variant 3	106.064 ± 5.444	$\mathbf{95.167} \pm 8.3813$

Our main result is that reward prediction with built-in pessimism (Variant 3) and UDS (Variant 2) both work very well. They demonstrate the effectiveness of leveraging unlabeled data in an offline RL setting. Using unlabeled data in the correct way can lead to roughly 30% performance increase in Walker2D and roughly 137% performance increase in Hopper. In addition, our proposed reward-prediction with pessimism (Variant 3) is more robust to over-fitting and works 43% better than UDS in the Hopper environment.

Further, we confirmed that naive reward prediction does not work at all as it leads to overestimated value functions on unlabeled data. And using unlabeled data to train the state dynamics alone is also useful for learning a better policy, but it never achieves comparable performance where unlabeled data is also used for critics training and policy evaluation.

Lastly, the original COMBO paper reports a performance of 103.3 using 2 million medium-expert data on Walker 2D (Yu et al. (2022b)), which is less than our result of 108.5. This shows that much less reliable data (10k expert) with a large random data (1M random) has the potential to perform better than using a huge amount of medium and expert data. Similarly, in the paper by Yu et al. (2022a), CQL plus UDS was able to achieve 81.5 using the same data set setup in Hopper. But our proposed approach achieves an even higher 95.2 performance, which may demonstrate the superiority of our method over pre-existing approaches applying UDS to model-free methods.

5 Discussion and Future Research

Although our initial result looks very promising, there are a number of limitations that naturally lead to many future research paths. First, we need to further investigate into over-fitting in the Hopper environment. We might try stronger regularization and different seeds to see if the over-fitting behaviour stops. Second, as we observed in the plots, training in reinforcement learning is very stochastic. Due to limited time and computing resources, we weren't able to repeat an experiment multiple times as each experiment takes about 10 hours training end-to-end. Thus, we would need to use different random seeds to repeat the experiment multiple times to confirm the robustness and validity of our result. Similarly, we only tested our methods on two standard environments. One future research is to test our approaches on more environments to study our methods' strengths and weaknesses.

Further, our problem setting naturally applies to multitask learning, where it is even more likely to have a very limited number of task-specific expert data and a very large number of task-agnostic data and shares some limited structural similarities. Thus, one very natural further research direction is to apply this method to multitask environments. For example, we can experiment on the Meta-World (Yu et al. (2020a)) and the Antmaze (Fu et al. (2021)) environments. In addition, there are many

combinations of labeled and unlabeled data we can explore. For example, we can check whether using medium-level data set in place of expert-level replays will decrease performance and whether using medium-level data set transition in place of random rollouts will increase performance. Lastly, we can also experiment with or incorporate Conservative Data Sharing (CDS) in Yu et al. (2021), which has shown promising results for using unlabeled data.

6 Conclusion

In this project, we proposed a novel method of incorporating unlabeled data into model-based offline reinforcement learning that naturally fits into the COMBO framework. Specifically, we train a reward model on the expert labeled data and fill in reward for unlabeled data points. Then, we push down on the value function predictions on the unlabeled data set, building in pessimism with respect to predicted rewards. We demonstrated that this method works very effectively across two standard offline RL benchmarks - Walker2D and Hopper. Our proposed method works at least as well as unlabeled data sharing (UDS) and has the potential to be more robust and reliable. Further, it surpasses some state-of-the-art approaches like using COMBO alone on a large labeled data and using UDS with conservative Q-learning. In addition, we confirmed that using naive reward prediction works poorly and demonstrated that using unlabeled data set correctly is crucial in model-based offline RL. As we discussed before, tackling the problem of leveraging unlabeled data sets has important implications for many domains of offline RL. Our project contributes to this line of work. In addition, our initial experiments give way to many other potential future research areas. In particular, we should confirm the usefulness and robustness of our method across more tasks to learn its strengths and weaknesses. And future research can explore and evaluate how our method can be applied to multi-task learning and meta-learning. We conclude that our approach shows promising results for incorporating unlabeled data into offline reinforcement learning.

7 Code from other sources

In this paper, in the process of implementing various approaches and the environment, a few existing repositories gave us inspirations and references.

1. D4RL Environment: We use this repository to set up our Walker2D and Hopper environments. D4RL is a standard benchmark for reinforcement learning and contains many other tasks. It is well-studied by a myriad of other researchers and so is suitable for our study.
2. COMBO Implementation: We use this repository for the baseline COMBO implementation. The repository largely follows the implementation mentioned in the original COMBO paper. For any other approaches involving the use of unlabeled data sets, we modified and innovate on this existing code base.

8 Team Contributions

The contribution of this paper is split evenly between the group members. The original idea was from a discussion in the office hour with the head TA, Rafael Rafailov. Then the team members, Tommy and Frank, modified an existing implementation of COMBO together to support our methods. Frank then ran the experiment on Walker2D environment, while Tommy ran it on Hopper. Finally, we combined our experimental results and jointly wrote and edited the poster and paper.

References

Ron Dorfman, Idan Shenfeld, and Aviv Tamar. 2021. Offline meta reinforcement learning – identifiability challenges and effective data collection strategies. In *Advances in Neural Information Processing Systems*, volume 34, pages 4607–4618. Curran Associates, Inc.

Hugh Durrant-Whyte, Nicholas Roy, and Pieter Abbeel. 2012. *Infinite-Horizon Model Predictive Control for Periodic Tasks with Contacts*.

Benjamin Eysenbach, Sergey Levine, and Ruslan Salakhutdinov. 2021. Replacing rewards with examples: Example-based policy search via recursive classification.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. 2021. D4rl: Datasets for deep data-driven reinforcement learning.

Scott Fujimoto, David Meger, and Doina Precup. 2019. Off-policy deep reinforcement learning without exploration.

Jonathan Ho and Stefano Ermon. 2016. Generative adversarial imitation learning.

Natasha Jaques, Asma Ghandeharioun, Judy Hanwen Shen, Craig Ferguson, Agata Lapedriza, Noah Jones, Shixiang Gu, and Rosalind Picard. 2019. Way off-policy batch deep reinforcement learning of implicit human preferences in dialog.

Dmitry Kalashnikov, Jacob Varley, Yevgen Chebotar, Benjamin Swanson, Rico Jonschkowski, Chelsea Finn, Sergey Levine, and Karol Hausman. 2021. Mt-opt: Continuous multi-task robotic reinforcement learning at scale.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. 2021. Morel : Model-based offline reinforcement learning.

Ksenia Konyushkova, Konrad Zolna, Yusuf Aytar, Alexander Novikov, Scott Reed, Serkan Cabi, and Nando de Freitas. 2020. Semi-supervised reward learning for offline reinforcement learning.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. 2020. Conservative q-learning for offline reinforcement learning.

Andrew Y. Ng and Stuart J. Russell. 2000. Algorithms for inverse reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, page 663–670, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Rafael Rafailov, Tianhe Yu, Aravind Rajeswaran, and Chelsea Finn. 2021. Offline reinforcement learning from images with latent space models. In *Proceedings of the 3rd Conference on Learning for Dynamics and Control*, volume 144 of *Proceedings of Machine Learning Research*, pages 1154–1168. PMLR.

Yuhang Ran, Yi-Chen Li, Fuxiang Zhang, Zongzhang Zhang, and Yang Yu. 2023. Policy regularization with dataset constraint for offline reinforcement learning.

Lu Wang, Wei Zhang, Xiaofeng He, and Hongyuan Zha. 2018. Supervised reinforcement learning with recurrent neural network for dynamic treatment recommendation.

Yifan Wu, George Tucker, and Ofir Nachum. 2019. Behavior regularized offline reinforcement learning.

Danfei Xu and Misha Denil. 2019. Positive-unlabeled reward learning.

Tianhe Yu, Aviral Kumar, Yevgen Chebotar, Karol Hausman, Chelsea Finn, and Sergey Levine. 2022a. How to leverage unlabeled data in offline reinforcement learning.

Tianhe Yu, Aviral Kumar, Yevgen Chebotar, Karol Hausman, Sergey Levine, and Chelsea Finn. 2021. Conservative data sharing for multi-task offline reinforcement learning.

Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. 2022b. Combo: Conservative offline model-based policy optimization.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. 2020a. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pages 1094–1100. PMLR.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. 2020b. Mopo: Model-based offline policy optimization.