Bayesian Q-learning With Imperfect Expert Demonstrations

Fengdi Che[*]
University of Alberta

Xiru Zhu [†]
McGill University

Doina Precup
McGill University, MILA, DeepMind

David Meger
McGill University

Gregory Dudek
McGill University, Samsung

Abstract

Guided exploration with expert demonstrations improves data efficiency for reinforcement learning, but current algorithms often overuse expert information. We propose a novel algorithm to speed up Q-learning with the help of a limited amount of imperfect expert demonstrations. The algorithm is based on a Bayesian framework to model suboptimal expert actions and derives Q-values' update rules by maximizing the posterior probability. It weighs expert information by the uncertainty of learnt Q-values and avoids excessive reliance on expert data, gradually reducing the usage of uninformative expert data. Experimentally, we evaluate our approach on a sparse-reward chain environment and six more complicated Atari games with delayed rewards. With the proposed methods, we can achieve better results than Deep Q-learning from Demonstrations (Hester et al., 2017) in most environments.

1 Introduction

Reinforcement learning (RL) trains an agent to maximize expected cumulative rewards through online interactions with an environment (Sutton & Barto, 2018). To speed up learning, online RL can be combined with offline expert demonstrations (Hester et al., 2018), which guides the agent toward high-rewarding behaviours and thus improves data efficiency. Most existing expert demonstration methods (Brys et al., 2015; Abbeel & Ng, 2004) either clone behaviours from the expert or bonus expert actions. However, agents are provided with imperfect expert data and gain distracting guidance. Some works start handling the imperfect expert data but still lack a solid theoretical foundation (Nair et al., 2018; Zhang et al., 2022).

Our paper adopts Bayesian frameworks (Ghavamzadeh et al., 2015; Dearden et al., 1998), which provide an analytical method to incorporate extra sub-optimal expert information for learning. In the probabilistic model for Bayesian inference, the sub-optimal expert decision is assumed to rely on the Boltzmann distribution dependent on the optimal expected returns from state-action pairs, called the optimal Q-values. Intuitively, our paper considers that the expert would prefer actions with higher optimal Q-values. Based on this relationship, the agent can infer values of the optimal Q-values from the given expert data during online learning and correct estimated Q-values to better correspond to the expert behaviours. This inference is equivalent to maximizing the posterior distribution of Q-values conditioned on expert data, but computing the maximum of a posterior probability is a difficult task (Hoffman et al., 2013; Diaconis & Ylvisaker, 1979; West et al., 1985). Therefore, our paper proposes to utilize the generalized extended Kalman filter (GEKF) (Fahrmeir, 1992) to derive a posterior maximum, requiring fewer restrictions on Q-values functions than other Bayesian RL frameworks (Dearden et al., 1998; Engel et al., 2003; Osband et al., 2019).

Under the assumptions from the GEKF framework, we derive an iterative forward way to compute Q-values, which consists of a Q-value-update step as in the Q-learning algorithm and an expert correction step. In the correction step, our update rule weighs expert information according to

[*] equal contribution, fengdi@ualberta.ca

[†] equal contribution

an agent's uncertainty of its self-learning result, measured by the estimated posterior variance of learned Q-values. Thus, the larger Q-values' variance is, the more our expert correction encourages expert behaviours by increasing expert actions' Q-values and decreasing non-experts' Q-values. This mechanism reduces the influence of uninformative expert data and avoids excessive guided exploration.

Furthermore, we propose our computationally efficient deep algorithm, Bayesian Q-learning from Demonstrations (BQfD), built on top of the Deep Q-learning Network (DQN) (Mnih et al., 2013). The algorithm embeds reliable expert knowledge into Q-values and leads to a more efficient exploration, as shown on a sparse-reward chain environment DeepSea and six randomly chosen Atari games. In most environments, our algorithm learns faster than Deep Q-learning from Demonstrations (DQfD) (Hester et al., 2018) and Prioritized Double Duelling (PDD) DQN (Wang et al., 2016).

2 Related Work

Reinforcement learning from demonstrations (RLfD) has drawn attention in recent years. This method only requires a small number of offline expert demonstrations and can noticeably improve performance. Deep Q-learning from demonstrations (DQfD) (Hester et al., 2018) includes an additional margin classification loss to ensure that Q-values of expert actions are higher than other actions. Reinforcement learning from demonstrations through shaping (Brys et al., 2015) assigns high potential to a state-action pair (s, a) when the action a is used by the expert in the neighbourhood of the state s. Wu et al. (Wu et al., 2020) further leverages reward potential computed by generative models. However, this requires an added complexity of training generative models. Also, expert data is used implicitly to learn rewards inversely at the beginning (Abbeel & Ng, 2004; Brown et al., 2019).

However, these approaches do not consider the case where the expert is imperfect and can often have difficulty exceeding expert performance. In contrast, Nair et al. (Nair et al., 2018) handle suboptimal demonstrations by only cloning expert actions when their current estimated Q-values are higher than others. Zhang et al. (Zhang et al., 2022) utilizes expert information when the cumulative Q-values on expert state-action pairs are larger than cumulative value functions. However, the learned Q-value is often unreliable and frequently changes, making it a poor judgment of expert data quality. At the same time, our measurement based on the posterior variance of estimated Q-values is more reasonable.

Jing et al. (Jing et al., 2020) models the suboptimal expert policy as a local optimum to maximize the expected returns and then constrains the agent to learn within a region around the expert policy, measured by KL divergence between occupancy measures. However, the local optimum condition is hard to satisfy, and the limitation of learning around the expert policy cannot deal with mistaken expert actions. In contrast, our assumption on Boltzmann distributed expert policy is much weaker.

3 Background

A Markov decision process (Sutton & Barto, 2018) is a tuple $M = \langle \mathcal{S}, \mathcal{A}, P, \gamma, R, \rho \rangle$ where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, P is the time-homogeneous transition probability matrix with $P(s'|s, a)$ as elements, $\gamma \in (0, 1)$ is the discount factor, R is the reward function, with $R(s, a)$ denoting a random vector describing rewards received after state-action pair (s, a) and ρ is the distribution of the initial state. At each time step h, the agent samples an action A_h from a policy μ, and then transits to the next state S_{h+1} and gains a reward $R(S_h, A_h)$. Our paper focuses on finite horizon cases with horizon H, where an agent stops at time step H. Also, our paper works on finite action and state spaces.

The expected discounted cumulative return starting from each state-action pair (s, a) at time step h and following the policy $\mu$is called the Q-value, denoted by $q_h^{\mu}(s, a)$, and is defined as follows:

$$q_h^{\mu}(s, a) = \mathbb{E}_{\tau}[\sum_{t=h}^{H} \gamma^t R(S_t, A_t)],$$

where τ denotes trajectories with $S_h = s$, $A_h = a$, $S_t \sim P(\cdot|S_{t-1}, A_{t-1})$, $\quad A_t \sim \mu(\cdot|S_h)$ and rewards $R(S_t, A_t)$. The unique optimal Q-value function at time h is defined as:

$$q_h^*(s,a) = q_h^{\mu^*}(s,a) = \sup_{\mu} q_h^{\mu}(s,a).$$

The optimal Q-values also satisfy the optimal Bellman equation for all states and actions:

$$q_h^*(s,a) = \mathbb{E}[R(s,a)] + \gamma \mathbb{E}_{S' \sim P(\cdot|s,a)}[\max_{a'} q_{h+1}^*(S',a')] =: \mathcal{T}_{h+1} q_{h+1}^*(s,a), h = 0, \cdots, H-1$$
$$q_H^*(s,a) = 0, \ \forall (s,a). \tag{1}$$

The optimal Bellman operator $\mathcal{T}$ is defined by the equation. Moreover, the optimal Q-values at each time h for all state-action pairs can be combined into one vector, $q_h^* \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{A}|}$, each element representing the optimal Q-value for a state-action pair.

Q-LEARNING

Optimal Q-values can be computed by Q-learning (Watkins & Dayan, 1992; Jin et al., 2018), which estimates Q-values directly and is based on the following update rule:

$$Q_h(s,a) = (1-\alpha) Q_h(s,a) + \alpha[R(s,a) + \gamma \max_{a'} Q_{h+1}(S',a'),$$

where Q is an estimation of Q-values, $S' \sim P(\cdot|s,a)$ is a sampled next state, and $\alpha \in (0,1)$ is the learning rate. A choice of adaptive learning rate is to assign each state-action pair a learning rate and decay it with respect to the number of visitations to the state-action pair. We use $n^l(s,a)$ to represent the number of visitation times of state-action pair (s,a) until and including episode l.

BAYESIAN MODEL-FREE FRAMEWORK

Bayesian reinforcement learning assumes that there is an initial guess or a prior probability over the model parameters, denoted by $P(\mathcal{M})$. Then in the model-based case, the agent gradually learns a posterior probability for the model parameters conditioned on observed trajectories (Duff, 2003). Next, an agent can learn a policy based on the most likely MDP or by sampling MDPs from the posterior. In the model-free case, we treat Q-values as random variables and implicitly embed parametric uncertainties from different MDPs. A prior distribution of the optimal Q-values at time h is defined as:

$$P(q_h^*(s,a) \le c) = P(\{\mathcal{M} : q_{\mathcal{M},h}^*(s,a) \le c\}).$$

Then algorithms compute the posterior probability of Q-values (Dearden et al., 1998; Osband et al., 2018).

4 MODEL

In this section, we present the probabilistic model used for Bayesian inference and then derive the update rule of Q-values by maximizing posterior probability.

4.1 PROBABILISTIC MODEL OF SUBOPTIMAL EXPERT ACTIONS

We start by modelling the relationship between expert actions and the optimal Q-values. The expert chooses the optimal actions most frequently but may contain mistakes and select low probability actions. The Boltzmann distribution can describe this behaviour with the optimal Q-values as parameters. Under this distribution, expert actions are sampled proportional to the exponential of optimal Q-values up to a constant multiplier. As known, this assumed expert policy maximizes the expected returns regularized by the entropy (Haarnoja et al., 2018), describing almost optimal but sometimes mistaken expert behaviours. The above assumption is formally presented as follows and shown on the left of figure 1, where the expert action at time h and state s is denoted by $A_{exp,h}$(s).

Assumption 1. *Assume that expert demonstrations are drawn from a policy π_{expert} dependent on the optimal Q-values. Also, this policy follows the Boltzmann distribution:*

$$\pi_{expert}(a|s) = \frac{e^{\eta q^*(s,a)}}{\sum_{b \in \mathcal{A}} e^{\eta q^*(s,b)}}, \tag{2}$$

where η is any positive constant.

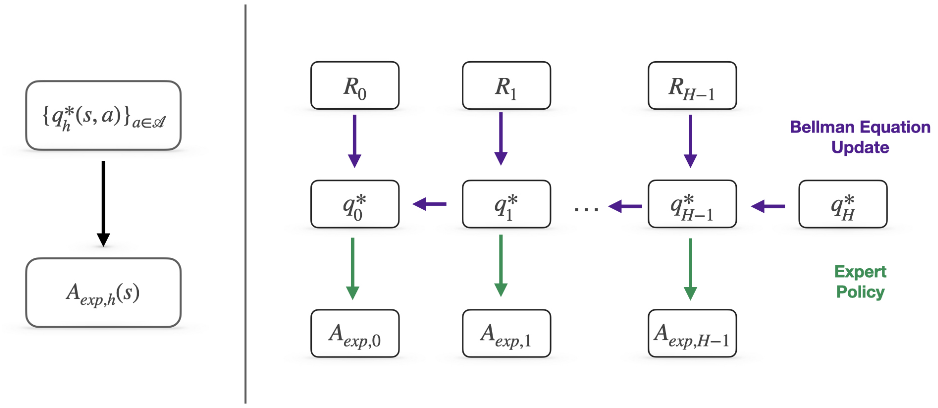

Figure 1: (Left) The figure demonstrates the relationship between optimal Q-values and expert actions. Expert actions are sampled proportional to the exponential of optimal Q-values up to a constant multiplier. (Right) The figure also describes the Bellman update rule for the Q-values along a trajectory.

Moreover, we capture the relationship between the optimal Q-values at different time steps in our probabilistic model through the Bellman equation 1. Considering the Bellman equation cannot be computed without the knowledge of environment dynamics, our model relies on samples of the reward and the next state, which are widely accepted in the field (Watkins & Dayan, 1992). We re-write the Bellman equation in a stochastic form as

$$q^*(s,a) = R(s,a) + max_{a'\in\mathcal{A}}\gamma q^*(S',a') + \nu, \tag{3}$$

where $R(s,a)$ and the next state S' are sampled according to the rules of the underlying MDP. The random noise ν is defined as

$$\nu = \mathbb{E}[R(s,a) + max_{a'\in\mathcal{A}}\gamma q^*(S',a')] - R(s,a) - max_{a'\in\mathcal{A}}\gamma q^*(S',a'),$$

which captures the randomness from the dynamics. Then, the whole model of relationships among rewards, optimal Q-values, and expert actions along a trajectory is presented on the right of figure 1.

4.2 GEKF Framework

Next, our desired estimated Q-values should not only follow the stochastic Bellman equation, but they should also most likely give the Boltzmann distribution that the expert is following. This objective is equivalent to maximizing the posterior probabilities of optimal Q-values equaling our estimations conditioned on expert information while constraining to the stochastic Bellman equation.

In order to compute the posterior probability, we need help from a framework under which the posterior probability density function or the posterior mode is of analytical form. Thus, our paper leverages the generalized extended Kalman filter (GEKF) (Fahrmeir, 1992), which can analyze the time series of Q-values with extra expert information and provides an estimation of the posterior maximum. This framework requires extra expert information following an exponential family distribution, which is already assumed.

Furthermore, the framework requires the random noise ν in the stochastic Bellman equation 3 at each Q-value update step to be Gaussian, which is also assumed in Osband et al. (2019) (Osband et al., 2019). Meanwhile, this assumption does not hurt in the latter stage of training since the influence of the random noise gradually approaches zero as the number of samples increases, and the learning rate decays. Notice that the random noise should have a zero expectation and bounded variance when rewards and time horizons are bounded. Thus, our paper considers ν as a Gaussian random variable with zero expectation and a fixed variance λ, as shown in the following assumption.

Assumption 2. *For all state-action pair (s,a),the noise ν is modeled independently by a Gaussian random variable $\nu \sim N(0,\lambda)$.*

Also, the GEKF framework approximates the posterior distribution by Gaussian and linearizes the maximization operator in Q-value updates as in Osband et al. (2019) (Osband et al., 2019).

4.3 Derivation of Q-values Update Rules

We derive our Q-value estimations Q under the above assumptions given by the GEKF framework (Fahrmeir, 1992). Firstly, this estimation should satisfy the stochastic Bellman equation in equation 3. Secondly, it should maximize the posterior probability of the optimal Q-values conditioned on expert demonstrations. This task is stated as follows:

$$\begin{aligned} \max_{Q_{0:H-1}} \; & P(q^*_{0:H-1} = Q_{0:H-1} | A_{exp,0:H-1}, R_{0:H-1}) \\ s.t. \quad & q^*_H(s,a) = 0, \; \forall s \in \mathcal{S} \text{ and } \forall a \in \mathcal{A} \\ & q^*_{h-1}(s,a) = R_{h-1}(s,a) + \gamma \max_{a'} q^*_h(S', a') + \nu, \; h = H, \cdots, 1, \; \forall s \in \mathcal{S} \text{ and } \forall a \in \mathcal{A} \end{aligned} \tag{4}$$

Recall that $q^*_h \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$ represents the Q-value vector at step h with each element as the value of one state-action pair, and similar for R_h; moreover, $A_{exp,h} \in \mathbb{R}^{|\mathcal{S}|}$ represents the expert actions chosen at all states at step h. Then, we can eliminate terms independent of Q-values in the posterior and obtain:

$$P(q^*_{0:H} | A_{exp,0:H-1}, R_{0:H-1}) \propto \prod_{h=0}^{H-1} P(A_{exp,h} | q^*_h) \prod_{h=0}^{H-1} P(q^*_h | q^*_{h+1}, R_h) P(q^*_H),$$

The three distributions are modelled through our previous assumptions: the expert action follows a Boltzmann distribution, Q-values at step h follow a Gaussian distribution with the expectation given by the Bellman equation, and Q-values at step H are all set to be zero.

Next, the task in equation 4 forms a constrained optimization problem and can further be transformed into a two-point boundary value problem (Fahrmeir, 1992; Sage & Melsa, 1971). This boundary value problem is then solved by the invariant embedding method, which provides approximate solutions for non-linear partial differential equations. The derivation is shown in the appendix following the steps in Sage (1997) (Sage & Melsa, 1971).

Finally, in a backward view, our estimations can be computed step-by-step as presented in algorithm 1. In the prediction step, the algorithm updates Q-values based on sampled rewards and sampled next states by the Bellman equation as the Q-learning algorithm (Watkins & Dayan, 1992). This update of estimated Q-values can be written as

$$Q_h = R_h + T_h Q_{h+1},$$

using a transformation matrix $T \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}| \times |\mathcal{S}||\mathcal{A}|}$. Its element at each state-action pair and its next-state-maximal-action pair equals to one, that is $T[(s,a), (S', a')] = 1$ where $S' \sim P(\cdot|s,a)$ and $a' = argmax_b Q_{h+1}(S', b)$. And all other elements are zero.

Then, the algorithm computes the estimated posterior variance W. Our derivation gives that this variance not only depends on the transformation matrix T, but also the Hessian of log expert policy $U \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}| \times |\mathcal{S}||\mathcal{A}|}$. So the update first computes $N_h = T_h^T W_{h+1} T_h + \lambda I$, and then calculates the estimated variance as $W_h = (N_h^{-1} + U_h)^{-1}$.

After computing the variance, expert information corrects Q-values as

$$Q_h(s,a) = Q_h(s,a) + \eta W_h[(s,a),(s,a)][\mathbb{1}[A_{exp,h}(s) = a] - \frac{exp(\eta Q_h(s,a))}{\sum_{b \in \mathcal{A}} exp(\eta Q_h(s,b))}].$$

Recall that η is the parameter of the expert policy. So a bonus weighted by the variance is given to expert actions and encourages the agent to explore them when it is not confident of the current learning result. When the variance of a Q-value estimate is low, the agent stops guided exploration by the expert and follows its own learnt policy. Thus, the agent can deal with inadequate expert data by gradually ignoring it. Meanwhile, this expert modification does not break the stochastic Bellman equation in equation 3 since it is listed as the constraint in the posterior maximization task in equation 4.

Algorithm 1 GEKF update rule for one episode

```
Initialize: Q-values at the last step $Q_H = 0$, Weight matrices of Q-value functions $W_H = \mathbf{0}^{|\mathcal{S}||\mathcal{A}|\times|\mathcal{S}||\mathcal{A}|}$, and Identity matrix $I$.
for time step $h = H-1, ..., 0$ do
    Prediction step:
    Update Q-value of each state-action pair based on sampled reward and next state though Bellman equation
    $Q_h(s,a) = R_h(s,a) + T_h Q_{h+1}, \forall s, a$
    Correction step:
    Compute the estimated posterior variance
    $W_h = T_h^T W_{h+1} T_h + \lambda I$
    $W_h = (W_h^{-1} + U_h)^{-1}$
    Embed in the expert correction
    $Q_h(s,a) + = +\eta W_h[(s,a),(s,a)][\mathbb{1}[A_{exp,h}(s) = a] - \frac{exp(\eta Q_h(s,a))}{\sum_{b\in\mathcal{A}} exp(\eta Q_h(s,b)}]$
end for
```

5 Algorithm

Our paper proposes an algorithm Bayesian Q-learning from Demonstrations (BQfD) to realize the GEKF update rules of Q-values. This algorithm samples a trajectory at the beginning of each turn and then updates Q-values by the sampled-based Bellman rule and the expert correction backward. Thus, our algorithm only updates the Q-value of the state-action pair (S_h, A_h) at each time step h and introduces a learning rate α for Bellman equation updates to reduce variances. This learning rate $\alpha_{n(s,a)}(s,a)$ is considered to be state-action pair dependent and decays as the number of that pair's visitations $n(s,a)$ increases.

However, the posterior variance calculation in the correction step is computationally inefficient due to the Hessian and inverse operations. To avoid these computations, we would like to approximate the posterior variance at a state-action pair (s,a). Given a first-order decaying learning rate $\alpha_{n(s,a)}(s,a) = \frac{1}{\beta + n(s,a)}$, short written as α and ignoring the variance change by expert information, the variance is roughly a function of the visitation times, denoted by $w_{n(s,a)} = O(\frac{1}{(n(s,a)}\lambda)$.

Furthermore, our paper utilizes deep neural networks and proposes a deep BQfD based on Prioritized Dueling Double (PDD) Deep Q-learning Network (DQN) (Mnih et al., 2013; Wang et al., 2016; Hessel et al., 2018). Notice that our current update rule requires expert information for every state, while a limited set of expert demonstrations is usually provided. Meanwhile, the indicator value in the expert correction may have conflict values if an expert provides multiple trajectories but chooses different actions at a state. Instead, our algorithm suggests storing expert data in the replay buffer and only performing expert correction when sampling expert data from the buffer.

Also, our algorithm weighs expert data to reduce off-policy biases. The importance sampling ratio is hard to calculate, so our algorithm uses a computationally-efficient weight

$$\left[\frac{e^{\eta Q(s,a)}}{\sum_b e^{\eta Q(s,b)}}\right]^\zeta$$

where the power ζ is a rescaling factor to avoid miniature weights. This ratio assigns higher values to more frequent state-action pairs under the current ϵ-greedy policy with respect to Q-estimates. The detailed algorithm is presented in algorithm 3 in the appendix.

6 Experiment

To evaluate the discernment of expert data, we test our algorithm using optimal and misleading expert data in a sparse-reward chain environment called DeepSea (Osband et al., 2019). Then, we further test the data efficiency of our algorithm with the help of a limited amount of suboptimal human data on six complicated Atari games (Brockman et al., 2016). We compare against other two discrete-action algorithms: Prioritized Dueling Double (PDD) Deep Q-learning Network (DQN) (Mnih et al., 2013;

Wang et al., 2016; Hessel et al., 2018) with EZ-greedy (Dabney et al., 2020) exploration and Deep Q-learning from demonstrations (DQfD). The DeepSea environment requires less than one hour of training on an Nvidia 2070 GPU. Each Atari game requires 21 days of training on Compute Canada server with either one Nvidia V100 or one Nvidia P100.

6.1 Chain Environment

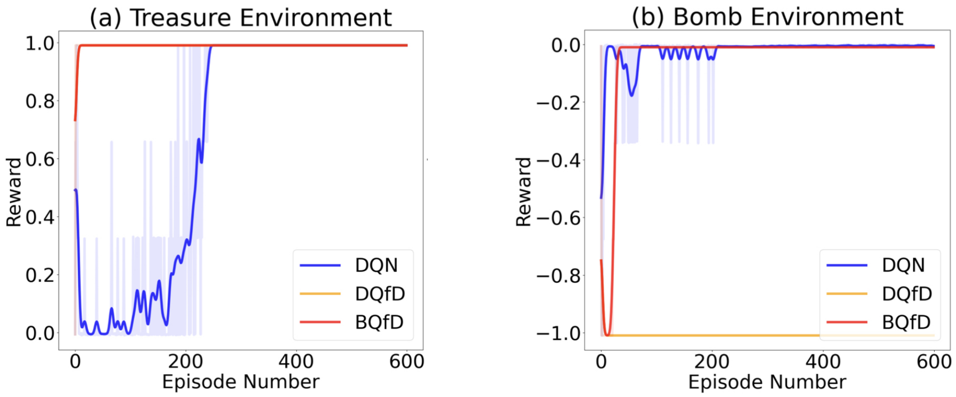

Figure 2: This figure shows agents' performances for the treasure case on the left and the bomb case on the right. It presents the training rewards versus the number of training episodes in an environment of size 50. Our algorithm can rapidly discover the optimal reward given the optimal expert data and acquire the optimal behaviour given wrong expert information.

DeepSea is a deterministic chain environment of length N with two actions: left and right. A figure to describe the environment is shown in the appendix. The agent is initialized on the left in each episode and terminates after N steps. The agent receives rewards of 0 for left actions and $\frac{-0.01}{N}$ for the right actions except for a fixed but unknown reward at the most right state. The unknown but fixed reward consists of either a treasure with a bonus of $+1$ or a bomb with a cost of -1. Regardless of the environment, we always provide agents with an expert trajectory towards the most right. Thus, in the treasure case, we test our algorithms' efficiency; in the bomb case, we evaluate our algorithms' robustness in the presence of highly misleading expert information. To provide a fair comparison, we include an expert trajectory to the replay buffer of DQN and provide no pretraining for any algorithms.

The results for the treasure case are shown on the left of figure 2. BQfD and DQfD can gain the optimal reward after one training episode much faster than DQN since the expert data directly leads to the reward, and these two algorithms extract expert information efficiently. But DQfD assumes the expert data to be optimal and causes catastrophic performance when expert demonstrations are misguiding, as seen in the bomb case, shown on the right of figure 2. On the contrary, our learning-from-demonstrations approach successfully learns the optimal policy, though the convergence speed is hindered due to confusing information. This result proves that BQfD efficiently extracts expert information and gradually stops using expert information.

6.2 Arcade Learning Environment

Experiment Settings

We further evaluated our approach on Arcade Learning Environments (Brockman et al., 2016; Mnih et al., 2013): Breakout, ChopperCommand, Freeway, MsPacman, VideoPinball, and Seaquest. The architecture for all three algorithms is the same as PDD DQN (Mnih et al., 2013; Wang et al., 2016; Hester et al., 2018; Hessel et al., 2018) with prioritized replay buffer (Schaul et al., 2015). Baselines' hyperparameters follow the choices in DQfD and the prioritized replay buffer. Meanwhile, we tune hyperparameters for BQfD on the environment Seaquest, and the detailed results are shown

Game	Length	DQN	DQfD	BQfD
Breakout	91M	221 ± 75	291 ± 119	**354** ± **132**
ChopperCommand	63M	4493 ± 1295	7067 ± 2403	**7377** ± **2801**
Freeway	23M	**30.4** ± **0.4**	28.2 ± 0.8	29.8 ± 0.8
MsPacman	76M	3628 ± 725	**4927** ± **993**	3550 ± 613
Seaquest	96M	20595± 8248	22388 ± 6294	**28546** ± **11613**
VideoPinball	116M	8619 ± 12582	17235 ± 15906	**30668** ± **18772**

Table 1: We present the average evaluation rewards over the previous 2M steps ± and the average standard deviations over the last 2M steps among three random seeds. BQfD shows apparent advantages in Breakout, ChopperCommand, Seaquest and VideoPinball and performs comparably to the best in Freeway and MsPacman.

in the appendix. All approaches are provided with one human-generated expert trajectory and are pre-trained with this expert data. Given the randomness during training, all results shown are averaged over three random seeds.

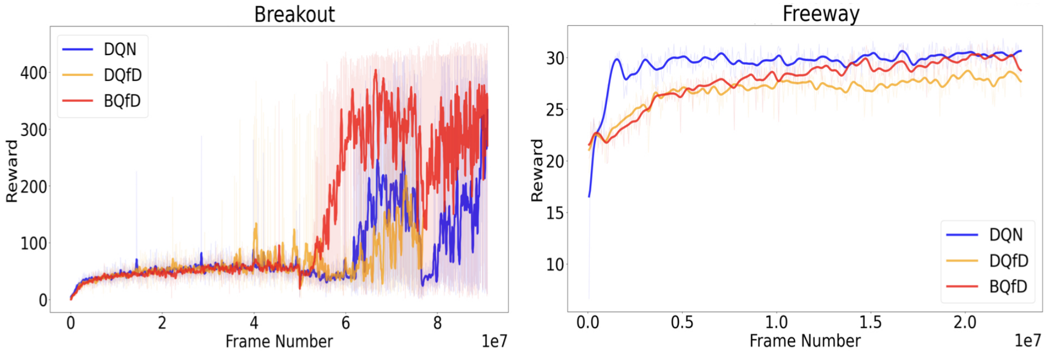

Figure 3: These figures present Breakout, ChopperCommand, SeaQuest, and Freeway's results averaged over three random seeds of BQfD, DQfD, and PDD DQN. We observe that BQfD's performance far exceeds other approaches and shows an obvious advantage in the late stage of learning.

Results

The overall results are shown in table 1. Here, we present final-learned evaluation rewards: the average means and standard deviations of evaluation rewards over the last 2M steps. Our algorithm outperforms the baselines in Breakout, ChopperCommand, Seaquest, and VideoPinball. In MsPacman, it still performs better than DQN but slightly worse than DQfD. In general, both three algorithms perform much worse than humans on MsPacman. DQN dominates the game in Freeway, but our algorithm recovers from weak expert data compared to DQfD.

The learning curve of Breakout and Freeway are presented in figure 3. Initially, the three algorithms' performances do not show significant differences. But the average evaluation performance slowly tilts in favour of BQfD; it learns fastest, and the order of DQN and DQfD varies. Surprisingly, BQfD's advantage arises in the latter stages of training instead of improving learning speed in the early stages. One reason might be that expert data leads to adequate explorations. Also, our approach encourages the agent to deviate from expert states and learn from trials and errors.

7 Future Work

Learning with demonstrations is an exciting direction. Though our approach is limited to discrete action problems, it can be extended to continuous actions and policy gradient algorithms left as future work. Furthermore, a more accurate approximation of posterior variance should be considered for future work. Moreover, when experts are not limited to being optimal but instead generate informative trajectories, the quality of expert demonstrations becomes more challenging to judge. The task of selecting the correct expert becomes an important question.

References

Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 1, 2004.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Daniel Brown, Wonjoon Goo, Prabhat Nagarajan, and Scott Niekum. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In *International conference on machine learning*, pp. 783–792. PMLR, 2019.

Tim Brys, Anna Harutyunyan, Halit Bener Suay, Sonia Chernova, Matthew E Taylor, and Ann Nowé. Reinforcement learning from demonstration through shaping. In *Twenty-fourth international joint conference on artificial intelligence*, 2015.

Will Dabney, Georg Ostrovski, and André Barreto. Temporally-extended {\epsilon}-greedy exploration. *arXiv preprint arXiv:2006.01782*, 2020.

Richard Dearden, Nir Friedman, and Stuart Russell. Bayesian q-learning. *Aaai/iaai*, 1998:761–768, 1998.

Persi Diaconis and Donald Ylvisaker. Conjugate priors for exponential families. *The Annals of statistics*, pp. 269–281, 1979.

Michael O Duff. Optimal learning: Computational procedures for bayes-adaptive markov decision processes. 2003.

Yaakov Engel, Shie Mannor, and Ron Meir. Bayes meets bellman: The gaussian process approach to temporal difference learning. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pp. 154–161, 2003.

Ludwig Fahrmeir. Posterior mode estimation by extended kalman filtering for multivariate dynamic generalized linear models. In *Journal of the American Statistical Association*, pp. 87.418: 501–509, 1992.

Mohammad Ghavamzadeh, Shie Mannor, Joelle Pineau, Aviv Tamar, et al. Bayesian reinforcement learning: A survey. *Foundations and Trends® in Machine Learning*, 8(5-6):359–483, 2015.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Dan Horgan, John Quan, Andrew Sendonaris, Ian Osband, et al. Deep q-learning from demonstrations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 2013.

Chi Jin, Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael I Jordan. Is q-learning provably efficient? *Advances in neural information processing systems*, 31, 2018.

Mingxuan Jing, Xiaojian Ma, Wenbing Huang, Fuchun Sun, Chao Yang, Bin Fang, and Huaping Liu. Reinforcement learning from imperfect demonstrations under soft expert guidance. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 5109–5116, 2020.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Ashvin Nair, Bob McGrew, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Overcoming exploration in reinforcement learning with demonstrations. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6292–6299. IEEE, 2018.

Ian Osband, John Aslanides, and Albin Cassirer. Randomized prior functions for deep reinforcement learning. *Advances in Neural Information Processing Systems*, 31, 2018.

Ian Osband, Benjamin Van Roy, Daniel J Russo, Zheng Wen, et al. Deep exploration via randomized value functions. *J. Mach. Learn. Res.*, 20(124):1–62, 2019.

Andrew P Sage and James L Melsa. Estimation theory with applications to communications and control. Technical report, SOUTHERN METHODIST UNIV DALLAS TEX INFORMATION AND CONTROL SCIENCES CENTER, 1971.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

Richard S Sutton and Andrew G Barto. Reinforcement learning: an introduction. 2018.

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, and Nando Freitas. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, pp. 1995–2003. PMLR, 2016.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3):279–292, 1992.

Mike West, P Jeff Harrison, and Helio S Migon. Dynamic generalized linear models and bayesian forecasting. *Journal of the American Statistical Association*, 80(389):73–83, 1985.

Yuchen Wu, Melissa Mozifian, and Florian Shkurti. Shaping rewards for reinforcement learning with imperfect demonstrations using generative models. *arXiv preprint arXiv:2011.01298*, 2020.

Xiaoqin Zhang, Huimin Ma, Xiong Luo, and Jian Yuan. Lidar: learning from imperfect demonstrations with advantage rectification. *Frontiers of Computer Science*, 16(1):1–10, 2022.