EFFICIENT EXPLORATION THROUGH BAYESIAN DEEP Q-NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose Bayesian Deep Q-Networks (BDQN), a principled and a practical Deep Reinforcement Learning (DRL) algorithm for Markov decision processes (MDP). It combines Thompson sampling with deep-Q networks (DQN). Thompson sampling ensures more efficient exploration-exploitation tradeoff in high dimensions. It is typically carried out through posterior sampling over the model parameters, which makes it computationally expensive. To overcome this limitation, we directly incorporate uncertainty over the value (Q) function. Further, we only introduce randomness in the last layer (i.e. the output layer) of the DQN and use independent Gaussian priors on the weights. This allows us to efficiently carry out Thompson sampling through Gaussian sampling and Bayesian Linear Regression (BLR), which has fast closed-form updates. The rest of the layers of the Q network are trained through back propagation, as in a standard DQN. We apply our method to a wide range of Atari games in Arcade Learning Environments and compare BDQN to a powerful baseline: the double deep Q-network (DDQN). Since BDQN carries out more efficient exploration, it is able to reach higher rewards substantially faster: in less than 5M$\pm$1M samples for almost half of the games to reach DDQN scores while a typical run of DDQN is 50-200M. We also establish theoretical guarantees for the special case when the feature representation is fixed and not learnt. We show that the Bayesian regret is bounded by $\widetilde{\mathcal{O}}(d\sqrt{N})$, after N time steps for a d-dimensional feature map, and this bound is shown to be tight up-to logarithmic factors. To the best of our knowledge, this is the first Bayesian theoretical guarantee for Markov Decision Processes (MDP) beyond the tabula rasa setting.

1 INTRODUCTION

A central challenge in reinforcement learning (RL) is to design efficient exploration-exploitation tradeoff that also scales to high-dimensional state and action spaces. Recently deep RL has shown good promise in being able scale to high-dimensional (continuous) spaces. These successes are mainly demonstrated in simulated domains where exploration is considered to be inexpensive and simple exploration strategies are deployed, e.g. ε-greedy which uniformly explores over all the non-greedy strategies with $1-\varepsilon$ probability. Such exploration strategies inherently inefficient for complex high-dimensional environments. On the other hand, more sophisticated strategies have mostly been limited to low dimensional MDPs. For example, optimism-under-uncertainty (OFU) is only practical when the domain is small enough to be represented with lookup tables for the Q-values (Jaksch et al., 2010; Brafman & Tennenholtz, 2003).

An alternative to optimism-under-uncertainty is Thompson Sampling (TS), a general sampling and randomizathttps://www.overleaf.com/1332425641pdxdghynmdhyion approach (in both frequentist and Bayesian settings) (Thompson, 1933). Under the Bayesian framework, Thompson sampling maintains a prior distribution over the environmental models (i.e. model of reward and dynamics), and updating the posterior based on observations collected. An action is chosen from the posterior distribution of the belief such that it maximizes the expected return. Thompson sampling has been observed to work significantly better than optimistic approaches in many low dimensional settings such as contextual bandits Chapelle & Li (2011), small MDPs Osband et al. (2013) and also has strong theoretical bounds. (Russo & Van Roy, 2014; Agrawal & Goyal, 2012; Osband et al., 2013; Abbasi-Yadkori & Szepesvári, 2015).

In the MDP setting, (model-based) Thompson Sampling involves sampling the parameters of reward and dynamics model, performing MDP planning using the sampled model and then computing the policy Strens (2000); Osband et al. (2013). However, the computational costs of posterior sampling and planning scales as cubic in the problem dimension, which makes it intractable for large MDPs. Therefore, some form of function approximation is required to scale Thompson Sampling to high dimensions, and this can be either of the model, the Q-value function, or the policy. To address this, Osband et al. (2014) introduced randomized least-squares value iteration (RLSVI) which combines linear value function approximation with Bayesian regression to directly sample the value-function weights from a distribution. The authors prove a regret bound for this approach in tabular MDPs. This has been extended to continuous spaces by Osband et al. (2016), where deep networks are used to approximate the Q function. Through a bootstrapped-ensemble approach, several deep-Q network (DQN) models are trained in parallel to approximate the posterior distribution. Other works use the posterior distribution over the parameters of each node in the network and employ variational approximation (Lipton et al., 2016b) or use noisy networks (Fortunato et al., 2017). These approaches significantly increase the computation cost compared to the standard DQN. For instance, the. bootstrapped-ensemble incurs a computation overhead that is linear in the number of bootstrap models. Moreover, despite principled design of the methods in the above works, they only produced modest gains over DQN in empirical studies.

Contribution 1 – Design of BDQN: We introduce Bayesian Deep Q-Network (BDQN), a Thompson-sampling algorithm for deep RL. It is a simple approach that extends randomized least-squares value iteration (Osband et al., 2014) to deep networks. We introduce stocasticity only in the last layer of the Q-network using independent Gaussian priors on the weights. This allows us to do efficient (approximate[1]) Thompson sampling using Bayesian linear regression (BLR), which has fast closed-form updates, and sampling from the resulting posterior distribution. The rest of the Q-network is trained through usual back propagation.

Contribution 2 – Strong empirical results for BDQN: We test BDQN on a wide range of Atari games (Bellemare et al., 2013; Machado et al., 2017), and compare our results to a powerful baseline: Double DQN (DDQN) (Van Hasselt et al., 2016) 3 a bias-reduced extension of DQN. BDQN and DDQN use the same network architecture, and follow the same target objective, and differ only in the way they select actions: DDQN uses ε-greedy exploration while BDQN performs (approximate) Thompson sampling.

We find that BDQN is able to reach much higher cumulative rewards, and also at a higher speed, compared to DDQN on all the tested games. We also found that BDQN can be trained with much higher learning rates compared to DDQN. This is intuitive since BDQN has better exploration strategy. We found that the cumulative reward (score) for BDQN at the end of training improves by a median of 300% with a maximum of $80k\%$ in these games. Also, BDQN has $300\% \pm 40\%$ (i.e. mean and standard deviation) improvement over these games w.r.t. area under the performance measure. This can be considered as a surrogate for sample complexity and regret. Indeed, no single measure of performance provides a complete picture of an algorithm, and we present detailed experiments in Section 4

In terms of computational cost, BDQN is only slightly more expensive compared to DQN and DDQN. This is the cost of inverting a 512×512 matrix every 100,000 time steps, which is negligible. On the other hand, more sophisticated Bayesian RL techniques are significantly more expensive and have not lead to large gains over DQN and DDQN.

Contribution 3 – Tight Bayesian regret bounds for continuous MDPs: We establish theoretical guarantees for the special case when the feature representation is fixed (i.e. all layers except the last), and not learnt. In other words, we consider episodic MDPs with continuous space of states and actions such that the Q-function is a linear function of a given d-dimensional feature map. We show that the Bayesian regret has an upper bound of $\widetilde{\mathcal{O}}(d\sqrt{N})$ after N time steps, which is shown to be tight up-to logarithmic factors. Since linear bandits are a special case of episodic continuous MDPs (with horizon length 1), it implies that our Bayesian regret bound is tight in the dimension d and in the number of episodes (up to logarithmic factors). This bound matches the Bayesian

[1]Since BDQN approximate the posterior, sampling from the induced confidence interval is an approximation to Thompson sample. In the remaining, we state Thompson Sampling and its approximation as Thomson samples unless we specify.

regret bound of linear bandits (Russo & Van Roy, 2014) as well as the frequentist regret bound for linear bandits (Abbasi-Yadkori et al., 2011) and for adaptive control of linear quadratic (LQ) framework (Abbasi-Yadkori & Szepesvári, 2011). To the best of our knowledge, this is the first Bayesian theoretical guarantee for continuous MDPs beyond the tabular setting.

Thus, our proposed approach has several desirable features – faster learning and better sample complexity due to targeted exploration, negligible computational overhead due to simplicity, significant improvement in experiments, and tight theoretical bounds.

2 THOMPSON SAMPLING VS ε-GREEDY AND BOLTZMANN EXPLORATION

In a value approximation RL algorithm, there are different ways to manage the exploration-exploitation trade-off. DQN uses a naive ε-greedy for exploration, where with ε probability it chooses a random action and with $1-\varepsilon$ probability it chooses the greedy action based on the estimated Q function. Note that there are only point estimates of the Q function in DQN. In contrast, our proposed Bayesian approach BDQN maintains uncertainties over the estimated Q function, and employs it to carry out Thompson Sampling based exploration. Here, we demonstrate the fundamental benefits of Thompson Sampling over ε-greedy and Boltzmann exploration strategies using simple examples. In Table. 1, we list the three strategies and their properties.

ε-greedy is the simplest exploration strategy since it is uniformly random over all the non-greedy actions. Boltzmann exploration is an intermediate strategy since it uses the estimated Q function to sample the actions. However, it does not maintain uncertainties over the Q function estimation. In contrast, Thompson sampling also incorporates uncertainties over Q estimation and utilizes most information for exploration strategy.

Consider the example in Figure 1(a) with current estimates and uncertainties of the Q function over different actions. ε-greedy is wasteful since it assigns uniform probability to explore over 5 and 6, which are obviously sub-optimal when the uncertainty estimates are available. In this setting, a possible remedy is Boltzmann exploration since it assigns lower probability to actions 5 and 6 but randomizes with almost the same probabilities over the remaining actions.

Table 1: Characteristics of Thompson Sampling, ε-greedy, and Boltzmann exploration what information they use for exploration

Strategy	Greedy-Action	Estimated Q-values	Estimated uncertainties
ε-greedy	✓	✗	✗
Boltzmann exploration	✓	✓	✗
Thompson Sampling	✓	✓	✓

However, Boltzmann exploration is sub-optimal in settings where there is high variances. For example if the current Q estimate is according to Figure 1(b), then Boltzmann exploration assigns almost equal probability to actions 5 and 6, even though action 6 has much higher uncertainty and needs to be explored more. Thus, both ε-greedy and Boltzmann exploration strategies are sub-optimal since they do not maintain an uncertainty estimate over the Q function estimation. In contrast, Thompson sampling uses both estimated Q function and its uncertainty estimates to carry out the most efficient exploration.

3 BAYESIAN DEEP Q-NETWORKS

Consider an MDP M as a tuple $\langle \mathcal{X}, \mathcal{A}, P, P_0, R, \gamma \rangle$, with state space $\mathcal{X}$, action space $\mathcal{A}$, the transition kernel P, accompanied with reward function of R, and discount factor $0 \leq \gamma < 1$. In value based model free RL, the core of most prominent approaches to tackle the learning of the Q function follows minimizing the Bellman residual (Lagoudakis & Parr, 2003; Antos et al., 2008). of the Q-function

$$\mathcal{L}(Q) = \mathbb{E}_{\pi}\left[\left(Q(x,a) - r - \gamma Q(x',a')\right)^2\right] \tag{1}$$

Where the tuple (x, a, r, x') consists of consecutive samples under behavioral policy π. Mnih et al. (2015) carries the same idea, and propose DQN where the Q-function is parameterized by a deep

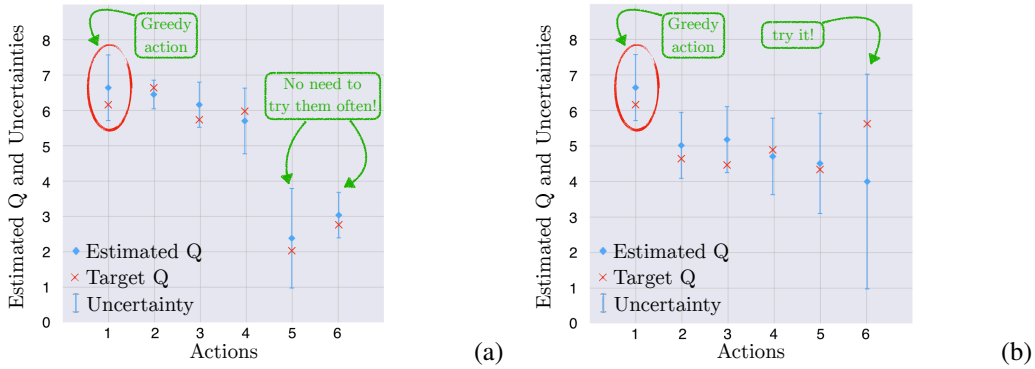

Figure 1: Thompson Sampling vs ε-greedy and Boltzmann exploration. (a) ε-greedy is wasteful since it assigns uniform probability to explore over 5 and 6, which are obviously sub-optimal when the uncertainty estimates are available. Boltzmann exploration randomizes over actions even if the optimal action is identifies. (b) Boltzmann exploration does not incorporate uncertainties over the estimated action-values and chooses actions 5 and 6 with similar probabilities while action 6 is significantly more uncertain. Thomson Sampling is a simple remedy to all these issues.

Algorithm 1 BDQN

Initialize parameter sets θ, θ^{target}, W, W^{target}, Cov, counter $= 0$
for episode = 1 to inf **do**
  for $t =$ to the end of episode **do**
    if count mod $T^{sample} = 0$ **then** sample $W \sim \mathcal{N}\left(W^{target}, Cov\right)$ **end if**
    Execute $a_t = \arg\max_{a'} \left[W^\top \phi_\theta(x_t)\right]_{a'}$, observe reward r_t, successor State x_{t+1}
    Store transition (x_t, a_t, r_t, x_{t+1}) in the replay buffer
    Sample a random minibatch of transitions $(x_\tau\ , a_\tau, r_\tau, x_{\tau+1})$ from replay buffer
    $y_\tau \leftarrow \begin{cases} r_\tau & \text{for terminal } x_{\tau+1} \\ r_\tau + \left[W^{target\top} \phi_{\theta^{target}}(x_{\tau+1})\right]_{\arg\max_{a'}[W^\top \phi_\theta(x_{\tau+1})]_{a'}} & \text{for non-terminal } x_{\tau+1} \end{cases}$
    $\theta \leftarrow \theta - \eta \cdot \nabla_\theta (y_\tau - \left[W^\top \phi_\theta(x_\tau)\right]_{a_\tau})^2$
    if count mod $T^{target} = 0$ **then** set $\theta^{target} \leftarrow \theta$ **end if**
    if count mod $T^{Bayes\ target} = 0$ **then** Update W^{target} and Cov **end if**
    count = count + 1

network. To improve the quality of Q estimate, they use back propagation on loss $\mathcal{L}(Q)$ using the TD update (Sutton & Barto, 1998). In order to reduce the bias of the estimator, they utilize a target network Q^{target} and target value $y = r + \gamma Q^{target}(x', \hat{a})$ where $\hat{a} = \arg\max_{a'} Q(x', a')$ with a new loss $\mathcal{L}(Q, Q^{target})$.

$$\mathcal{L}(Q, Q^{target}) = \mathbb{E}_\pi \left[(Q(x,a) - y)^2\right] \tag{2}$$

This regression problem is pursued to minimize the estimated loss $\widehat{\mathcal{L}}(Q, Q^{target})$, which minimize the error between the Q and the target y. A DQN agent, once in a while updates the Q^{target} network by setting it to Q network, follow the regression with the new target value and provides a biased estimator of the target. To mitigate the bias in this estimator, DDQN proposes to instead $\hat{a} = \arg\max_{a'} Q^{target}(x', a')$. We deploy this approach for the remaining of this paper. In this work, instead of regression, we minimize $\widehat{\mathcal{L}}(Q, Q^{target})$ via Bayesian Linear regression, which is also inspired by our theoretical work. While deploying BLR, we also train the feature representation in order to approximately develop a linear features for Q

We utilize the DQN architecture, remove its last layer, and directly build a BLR (Rasmussen & Williams, 2006) on the output of the deep network parametrized by θ, $\phi^\theta(x) \in \mathbb{R}^d$, known as feature representation layer. We use BLR to efficiently approximate the distribution over the Q-values where the uncertainty over the values is captured. A common assumption in DNN is that the feature representation is suitable for linear classification or regression (same assumption in DDQN), therefore,

building a linear model on the features is a suitable choice. The Q-functions are approximated as a linear transformation of the deep neural network features, i.e. for a given pair of state-action, $Q(x, a) = \phi^\theta(x)^\top w_a$, where $w_a \in \mathcal{R}^d, \forall a \in \mathcal{A}, x \in \mathcal{X}$.

Consequently, as mentioned in the previous section, the target value is generated using a target model. The target model follows the same structure as the Q model, and contains $\phi^{\theta^{target}}(\cdot) \in \mathbb{R}^d$ denotes the feature representation of target network, and $w^{target}{}_{\hat{a}},\ \forall \hat{a} \in \mathcal{A}$ denotes the target weight. For a given tuple of experience (x, a, r, x')

$$Q(x, a) = \phi^\theta(x)^\top w_a \rightarrow y := r + \gamma \phi^{\theta^{target}}(x') w^{target}{}_{\hat{a}},\ \textit{where} \quad \hat{a} = \arg\max_a \phi^{\theta^{target}\top} w_a^{target}.$$

Therefore, by deploying BLR on the features, we approximate the posterior distribution of model parameter for each w_a, as well as the posterior distribution of the Q-function using the corresponding target values. As in BLR methods, for each weight vector we maintain a Gaussian prior $\mathcal{N}(0, \sigma^2 I)$ while the target value $y \sim w_a^\top \phi^\theta(x) + \epsilon$ and $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$ is an i.i.d. noise.

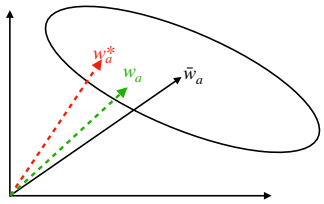

Figure 2: BDQN deploys Thompson Sampling to, sample $w_a\ \forall a \in A$ around the empirical mean $\overline{w}_a$ with w_a^* the underlying parameter of interest.

Given a experience replay buffer $\mathcal{D} = \{x_\tau, a_\tau, y_\tau\}_{\tau=1}^D$, we construct $|\mathcal{A}|$ (number of actions) disjoint datasets for each action, $\mathcal{D}_a$, a set of experiences under action a. We are interested in approximating the posterior distribution of each w_a, correspondingly the $Q(x, a)$. For each action a, we construct a matrix $\Phi_a^\theta \in \mathbb{R}^{d \times |\mathcal{D}_a|}$, a concatenation of feature column vectors $\{\phi^\theta(x_i)\}_{i=1}^{|\mathcal{D}_a|}$, and $\mathbf{y}_a \in \mathbb{R}^{|\mathcal{D}_a|}$, a concatenation of target values in set $\mathcal{D}_a$. Therefore the posterior distribution of w_a Fig. 2 is approximated as follows:

$$w_a \sim \mathcal{N}\left(\overline{w}_a := \frac{1}{\sigma_\epsilon^2} \Xi_a \Phi_a^\theta \mathbf{y}_a, \Xi_a\right), \quad \Xi_a = \left(\frac{1}{\sigma_\epsilon^2} \Phi_a^\theta \Phi_a^{\theta\top} + \frac{1}{\sigma^2} I\right)^{-1} \tag{3}$$

A sample of $Q(x, a)$ is $w_a^\top \phi^\theta(x)$ where w_a is a draw as in Eq. 3 and Fig. 2. Every T^{sample} times step, we draw $w_a,\ \forall a \in \mathcal{A}$ follow a_{TS} and train the feature network under loss $(y_\tau - w_{a_\tau}^\top \phi^\theta(x_\tau))^2$ with x_τ, a_τ, y_τ samples from replay buffer i.e.

$$a_{\text{TS}} = \arg\max_a w_a^\top \phi^\theta(x), \qquad \theta \leftarrow \theta - \eta \cdot \nabla_\theta (y_\tau - \left[W^\top \phi^\theta(x_\tau)\right]_{a_\tau})^2 \tag{4}$$

We update the target network every T^{target} steps and set θ^{target} to θ. With the period of $T^{Bayes\ target}$, we update the approximated posterior distribution using a larger minibatch of data, drawn from replay buffer, and set the $w_a^{target},\ \forall a \in \mathcal{A}$ to the mean of the posterior A.6. Algorithm 1 gives the full description of BDQN.

4 Experiments

We apply BDQN on a variety of Atari games using the Arcade Learning Environment (Bellemare et al., 2013) through OpenAI Gym[2] (Brockman et al., 2016). As a baseline, we run the DDQN algorithm and evaluate BDQN on the measures of sample complexity and score. All the implementations are coded in MXNet framework (Chen et al., 2015). The details on architecture, Apx. A.1, learning rate Apx. A.4, computation A.5. In Apx. A.2 and A.3 we describe how we spend less than two days on a single game for the hyper parameter choices which is another evidence of significance of BDQN.

Baselines: We implemented DDQN and BDQN exactly the same way as described in Van Hasselt et al. (2016). We also aimed to implement a couple of other deep RL methods that employ strategic exploration. Unfortunately we encountered several implementation challenges where neither code nor the implementation details was publicly available. Despite the motivation of this work on sample

[2]Each input frame is a pixel-max of the two consecutive frames. We detailed the environment setting in the implementation code

Table 2: Comparison of scores and sample complexities (scores in the first two columns are average of 100 consecutive episodes). Sample complexity, *SC*: the number of samples the BDQN requires to beat the human score (Mnih et al., 2015)(" − " means BDQN could not beat human score). SC^+: the number of samples the BDQN requires to beat the score of DDQN$^+$. The scores of DDQN$^+$ as well as bootstrap DQN (Osband et al., 2016), CTS, Pixel, Reactor (Ostrovski et al., 2017) are borrowed from the original papers. For NoisyNet (Fortunato et al., 2017), the scores of NoisyDQN are reported.

Game	BDQN	DDQN	DDQN^{+3}	Bootstrap[4]	NoisyNet[5]	CTS	Pixel	Reactor	Human	SC	SC^+	Step
Amidar	**5.52k**	0.99k	0.7k	1.27k	1.5k	1.03k	0.62k	1.18k	1.7k	22.9M	4.4M	100M
Alien	3k	2.9k	2.9k	2.44k	2.9k	1.9k	1.7k	**3.5k**	6.9k	-	36.27M	100M
Assault	**8.84k**	2.23k	5.02k	8.05k	3.1k	2.88k	1.25k	3.5k	1.5k	1.6M	24.3M	100M
Asteroids	**14.1k**	0.56k	0.93k	1.03k	2.1k	3.95k	0.9k	1.75k	13.1k	58.2M	9.7M	100M
Asterix	**58.4k**	11k	15.15k	19.7k	11.0	9.55k	1.4k	6.2k	8.5k	3.6M	5.7M	100M
BeamRider	8.7k	4.2k	7.6k	**23.4k**	14.7k	7.0k	3k	3.8k	5.8k	4.0M	8.1M	70M
BattleZone	**65.2k**	23.2k	24.7k	36.7k	11.9k	7.97k	10k	45k	38k	25.1M	14.9M	50M
Atlantis	3.24M	39.7k	64.76k	99.4k	7.9k	1.8M	40k	**9.5M**	29k	3.3M	5.1M	40M
DemonAttack	11.1k	3.8k	9.7k	82.6k	26.7k	**39.3k**	1.3k	7k	3.4k	2.0M	19.9M	40M
Centipede	**7.3k**	6.4k	4.1k	4.55k	3.35k	5.4k	1.8k	3.5k	12k	-	4.2M	40M
BankHeist	0.72k	0.34k	0.72k	**1.21k**	0.64k	1.3k	0.42k	1.1k	0.72k	2.1M	10.1M	40M
CrazyClimber	124k	84k	102k	**138k**	121k	112.9k	75k	119k	35.4k	0.12M	2.1M	40M
ChopCmd[6]	**72.5k**	0.5k	4.6k	4.1k	5.3k	5.1k	2.5k	4.8k	9.9k	4.4M	2.2M	40M
Enduro	1.12k	0.38k	0.32k	1.59k	0.91k	0.69k	0.19k	**2.49k**	0.31k	0.82M	0.8M	30M
Pong	**21**	18.82	**21**	20.9	**21**	20.8	17	20	9.3	1.2M	2.4M	5M

complexity, since we do not have access to the performance plots of these methods, the least is to report their final scores. To try to illustrate the performance of our approach we instead, extracted the best reported scores from a number of state-of-the-art deep RL methods and include them in Table 2, which is the only way to bring a comparison. These scores are reported score of the trained model on 200M steps, mainly in the evaluation phase where exploration is almost zero. Note that this is not a perfect comparison, as there are additional details that are not included in the papers, i.e. it is hard to just compare the reported results (an issue that has been discussed extensively recently, e.g. (Henderson et al., 2017)).[7]. Moreover, when the regret analysis of an algorithm is considered, no evaluation phase required, and the reported results of BDQN are those while exploration. It is worth noting that, the scores during evaluation are much higher than those during the course of learning and exploration. Apx. A.8

Results: The results are provided in Fig. 3. We observe that BDQN significantly improve the sample complexity of DDQN and reaches the highest score of DDQN in much fewer number of samples than DDQN needs. We expected BDQN, due to it better exploration-exploitation strategy, to improve the regret and enhance the sample complexity, but we also observed a significantly improvement in scores. It worth noting that since BDQN is design to minimize the regret, and the study in Fig. 3 also for sample complexity analysis, either of the reported BDQN DDQN scores are while exploration. For example, DDQN gives score of 18.82 during the learning phase, but setting ε to zero, it mostly gives the score of 21. We report the number of samples (sample complexity (SC)) it take from BDQN to reach human scores, *SC*, and DDQN$^+$ scores, SC^+ in Table. 2 and Table. 3.

For the game *Atlantis*, DDQN$^+$ gives score of $64.67k$ during the evaluation phase, while BDQN reaches score of $3.24M$ after $20M$ samples. As it is been shown in Fig. 3, BDQN saturates for *Atlantis* after 20M samples. We realized that BDQN reaches the internal *OpenAIGym* limit of $max_episode$, where relaxing it improves score after $15M$ steps to $62M$, Apx. A.7. We observe that BDQN immediately learns significantly better policies due to its targeted exploration in a much shorter period of time. Since BDQN on game *Atlantis* promise a big jump around time step $20M$, we ran it five more times in order to make sure it was not just a coincidence Apx. A.7 Fig. 6. For the game Pong, we ran the experiment for a longer period but just plotted the beginning of it in order to observe the difference. Due to cost of deep RL methods, for some games, we run the experiment until a plateau is reached.

[7]To further reproducibility, we released our codes and trained models. Since DRL experiments are expensive, we also have released the recorded arrays of returns. We also implemented bootstrapped DQNOsband et al. (2016) and released the code but we were not able to reproduce their results beyond the performance of random policy

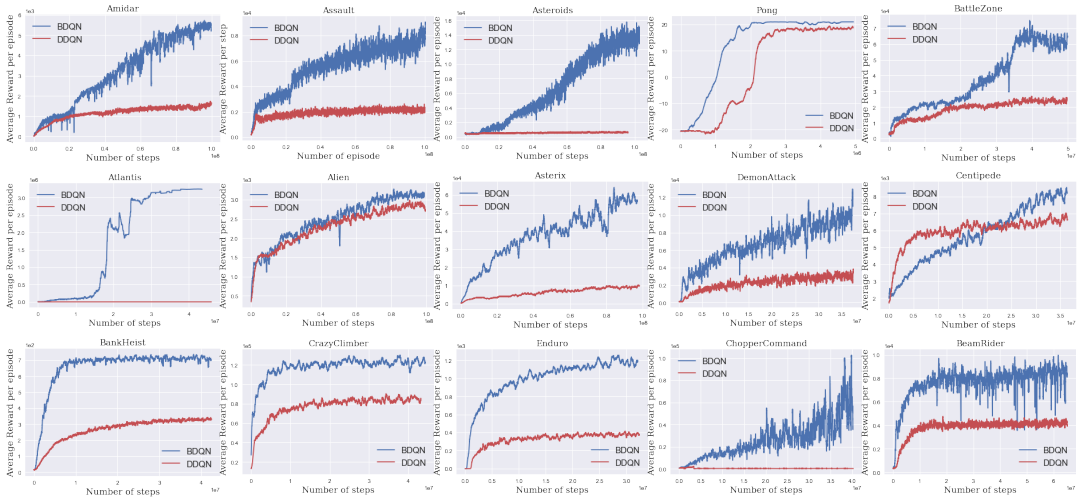

Figure 3: The efficient and targeted exploration of BDQN

5 Bayesian Regret Bound

In this section we provide the analysis of Bayesian regret bound when the feature representation is given and fixed. Consider a finite horizon stochastic MDP M as a tuple $\langle \mathcal{X}, \mathcal{A}, P, P_0, R, H \rangle$, with state space $\mathcal{X}$, action space $\mathcal{A}$, the transition kernel P, accompanied with reward function of R, and H as the length of episodes. The agent objective is to optimize the overall expected cumulative reward. For the feature representation $\phi(\cdot,\cdot) := \mathcal{X} \times \mathcal{A} \to \mathcal{R}^d$, we say Q-function is a linear transformation of this feature representation if there exists a problem dependent vector $\omega^* \in \mathbb{R}^d$ such that $Q^{\omega^*}_{\pi^*}(x,a) := \phi(x,a)^\top \omega^*$, $\forall x, a \in \mathcal{X} \times \mathcal{A}$. Here $\pi^* : \mathcal{X} \to \mathcal{A}$ is the optimal policy of MDP and also satisfies $\pi^*(x) := \arg\max_{a \in \mathcal{A}} Q^{\omega^*}_{\pi^*}(x,a)$. Respectively, let $V^{\omega^*}_{\pi^*}$ denote the value of the π^* under the model with ω^*. The following is the generative model of the environment;

$$\overline{\mathbf{R}} := \Psi \mathbf{R} = \mathbf{\Phi}(x^{\{1:H\}}, a^{\{1:H\}})\omega^* + \Psi \nu(x^{\{1:H\}}, a^{\{1:H\}}) \tag{5}$$

Where Ψ is upper triangular matrix all ones 6. The vector $\mathbf{R} \in \mathbb{R}^H$ is the random vector of rewards in an episode while $\overline{\mathbf{R}} \in \mathbb{R}^H$, the corresponding per step return. For a sequence of states and actions, $x^{\{1:H\}}, a^{\{1:H\}}$, the matrix $\mathbf{\Phi}(x^{\{1:H\}}, a^{\{1:H\}}) \in \mathbb{R}^{H \times d}$ is row-wise concatenation of the corresponding features. Our algorithm maintains a prior over the vector ω^* while updating the posterior by observing data. At the beginning of an episode t, a sample of ω_t is drawn from the posterior and the following decisions in the same episode are optimal with respected to ω_t, i.e. π_t and $a := \arg\max_{a \in \mathcal{A}} \phi^\top(x,a)\omega_t, \forall x \in \mathcal{X}$. Similar to the linear bandit problems, consider the following generic assumptions,

- The noise vector ν is sub-Gaussian vector with parameter σ. Look at Asm. 1
- The random vectors ϕ^h and variable ν_h $h \in [1, \dots, H]$ satisfies the Markov property.
- $\|\omega^*\|_2 \leq L_\omega$ and $trace\left(\mathbf{\Phi}(\cdot)\mathbf{\Phi}(\cdot)^\top\right) \leq L$, a.s.

Theorem 1 (Bayesian Regret)**.** *For an episodic MDP with episode length H, feature mapping $\phi(x,a) \in \mathbb{R}^d$, the PSRL on ω, after T episodes, guarantees;*

$$\textbf{\textit{BayesReg}}_T := \sum_t^T \mathbb{E}_{\omega*}\left[V^{\omega^*}_{\pi^*} - V^{\omega^*}_{\pi_t}\right] \leq \mathcal{O}\left(d\sqrt{HT\log(T/\delta)}\right)$$

with probability at least $1-\delta$. Proof: in Theorem. B

Proof Sketch: In order to prove this regret bound, we show that over the course of exploration and exploitation, the estimated parameter vector $\widehat{\omega}_t$ concentrates to to the true ω^* with a fast rate of $\widetilde{\mathcal{O}}(1/t)$ Lemma. 3. Since the sampled parameter ω_t is also close to $\widehat{\omega}_t$, overall per step regret shrinks as rate of $\widetilde{\mathcal{O}}(1/t)$. Moreover, each episode has length H, the regret after T episodes should not grow

faster than $\widetilde{\mathcal{O}}(TH)$, i.e. $\widetilde{\mathcal{O}}(N)$ where is the total number of interactions with the environment. This theorem directly extendable to $0 \leq \gamma \leq 1$. This bounded shows the complexity of this problem grows not faster than linear in the dimension, and $\sqrt{HT}$. This bound is similar to those in linear bandits (Abbasi-Yadkori et al., 2011; Russo & Van Roy, 2014) and linear quadratic control (Abbasi-Yadkori & Szepesvári, 2011), i.e. $\widetilde{\mathcal{O}}(d\sqrt{T})$. On lower bound; since for $H = 1$, this problem reduces to linear bandit and for linear bandit the lower bound is $\Omega(d\sqrt{T})$ therefore, our bound is order optimal in d and T.

6 Related Work

The complexity of the exploration-exploitation trade-off has been deeply investigated in RL literature for both continuous and discrete MDPs (Kearns & Singh, 2002; Brafman & Tennenholtz, 2003; Asmuth et al., 2009; Kakade et al., 2003; Ortner & Ryabko, 2012). Jaksch et al. (2010) investigates the regret analysis of MDPs with finite state and action where Optimism in Face of Uncertainty (OFU) principle is deployed to guarantee a regret upper bound, while Ortner & Ryabko (2012) relaxes it to continues state space and propose sublinear regret bound. Azizzadenesheli et al. (2016a) deploys OFU and propose a regret upper bound for Partially Observable MDPs (POMDPs) using spectral methods (Anandkumar et al., 2014). Furthermore, Bartók et al. (2014) tackles a general case of partial monitoring games and provides minimax regret guarantee. For linear quadratic models, (Abbasi-Yadkori & Szepesvári, 2011), OFU is deployed to provide optimal regret bound.

In multi-arm bandit, there are compelling empirical pieces of evidence that Thompson Sampling sometimes provides better results than optimism-under-uncertainty approaches (Chapelle & Li, 2011), while also the state of the art performance bounds are preserved (Russo & Van Roy, 2014; Agrawal & Goyal, 2012). A natural adaptation of this algorithm to RL, posterior sampling RL (PSRL) Strens (2000) also shown to have good frequentist and Bayesian performance guarantees (Osband et al., 2013; Abbasi-Yadkori & Szepesvári, 2015).

Even though the theoretical RL addresses the exploration and exploitation trade-offs, these problems are still prominent in empirical reinforcement learning research (Mnih et al., 2015; Abel et al., 2016; Azizzadenesheli et al., 2016b). On the empirical side, the recent success in the video games has sparked a flurry of research interest. Following the success of Deep RL on Atari games (Mnih et al., 2015) and the board game Go (Silver et al., 2017), many researchers have begun exploring practical applications of deep reinforcement learning (DRL). Some investigated applications include, robotics (Levine et al., 2016), self-driving cars (Shalev-Shwartz et al., 2016), and safety (Lipton et al., 2016a). Inevitably for PSRL, the act of posterior sampling for policy or value is computationally intractable with large systems, so PSRL can not be easily leveraged to high dimensional problems (Ghavamzadeh et al., 2015; Engel et al., 2003). To remedy these failings Osband et al. (2017) consider the use of randomized value functions. For finite state-action space MDP, Osband et al. (2014) propose posterior sampling directly on the space of Q-functions and provide a strong Bayesian regret bound guarantee. To approximate the posterior, they use BLR on one-hot encoding of state-action and improve the computation complexity of PSRL. The approach deployed in BDQN is strongly related and similar to this work, and is a generalization to continues state-action space MDPs.

To combat these shortcomings, Osband et al. (2016) suggests a bootstrapped-ensemble approach that trains several models in parallel to approximate the posterior distribution. Other works suggest using a variational approximation to the Q-networks (Lipton et al., 2016b) or a concurrent work on noisy network (Fortunato et al., 2017). However, most of these approaches significantly increase the computational cost of DQN and neither approach produced much beyond modest gains on Atari games. Interestingly, Bayesian approach as a technique for learning a neural network has been deployed for object recognition and image caption generation where its significant advantage has been verified Snoek et al. (2015).

Concurrently, Levine et al. (2017) proposes least squares temporal difference which learns a linear model on the feature representation in order to estimate the Q-function while ε-greedy exploration is employed and improvement on 5 tested Atari games is provided. Out of these 5 games, one is common with our set of 15 games which BDQN outperform it by factor of 360% (w.r.t. the score reported in their paper). As motivated by the theoretical understanding, our empirical study shows that performing Bayesian regression instead, and sampling from the result, can yield a substantial

benefit, indicating that it is not just the higher data efficiency at the last layer, but that leveraging an explicit uncertainty representation over the value function is of substantial benefit.

Drop-out, as another randomized exploration method is proposed by Gal & Ghahramani (2016) but Osband et al. (2016) investigates the sufficiency of the estimated uncertainty and hardness in driving a suitable exploitation out of it. As stated before, in spite of the novelties proposed by the methods, mentioned in this section, neither of them, including TSbased approaches, produced much beyond modest gains on Atari games while BDQN provides significant improvements in terms of both sample complexity and final performance.

7 CONCLUSION

In this work we proposed BDQN, a practical Thompson sampling based RL algorithm which provides targeted exploration in a computationally efficient manner. It involved making simple modifications to the DDQN architecture by replacing the last layer with a Bayesian linear regression. Under the Gaussian prior, we obtained fast closed-form updates for the posterior distribution. We demonstrated significantly faster training and much better performance in many games compared to the reported results of a wide number of state-of-the-art baselines. Due to the computational limitations we did not try the algorithm on all games and it remains an interesting issue to further explore its performance, and combine it with other advances in deep RL. We also established theoretical guarantees for an episodic MDP with continuous state and action spaces in the special case where the feature representation is fixed, BDQN is given the feature representation. We derived an order-optimal Bayesian regret bound of $\widetilde{\mathcal{O}}(d\sqrt{N})$ after N time steps.

In future, we plan to extend the analysis to the more general Frequentist setting. The current theoretical guarantees are mainly developed for the class of linear functions. For general class of functions, optimism in the face of uncertainty is deployed to guarantee a tight PAC bound (Jiang et al., 2016) but the proposed algorithm requires solving a NP-hard problem at each iteration. We aim to further provide a tight theoretical bound through Thomson sampling while also preserving the computational feasibility.

REFERENCES

Yasin Abbasi-Yadkori and Csaba Szepesvári. Regret bounds for the adaptive control of linear quadratic systems. In *COLT 2011 - The 24th Annual Conference on Learning Theory, June 9-11, 2011, Budapest, Hungary*, 2011.

Yasin Abbasi-Yadkori and Csaba Szepesvári. Bayesian optimal control of smoothly parameterized systems. In *UAI*, pp. 1–11, 2015.

Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems 24 - NIPS*, pp. 2312–2320, 2011.

David Abel, Alekh Agarwal, Fernando Diaz, Akshay Krishnamurthy, and Robert E Schapire. Exploratory gradient boosting for reinforcement learning in complex domains. *arXiv*, 2016.

Shipra Agrawal and Navin Goyal. Analysis of thompson sampling for the multi-armed bandit problem. In *COLT*, 2012.

Animashree Anandkumar, Rong Ge, Daniel Hsu, Sham M Kakade, and Matus Telgarsky. Tensor decompositions for learning latent variable models. *The Journal of Machine Learning Research*, 15(1):2773–2832, 2014.

András Antos, Csaba Szepesvári, and Rémi Munos. Learning near-optimal policies with bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning*, 2008.

John Asmuth, Lihong Li, Michael L Littman, Ali Nouri, and David Wingate. A bayesian sampling approach to exploration in reinforcement learning. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, 2009.

Kamyar Azizzadenesheli, Alessandro Lazaric, and Animashree Anandkumar. Reinforcement learning of pomdps using spectral methods. In *Proceedings of the 29th Annual Conference on Learning Theory (COLT)*, 2016a.

Kamyar Azizzadenesheli, Alessandro Lazaric, and Animashree Anandkumar. Reinforcement learning in rich-observation mdps using spectral methods. *arXiv preprint arXiv:1611.03907*, 2016b.

Gábor Bartók, Dean P Foster, Dávid Pál, Alexander Rakhlin, and Csaba Szepesvári. Partial monitoringclassification, regret bounds, and algorithms. *Mathematics of Operations Research*, 2014.

Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *J. Artif. Intell. Res.(JAIR)*, 2013.

Ronen I Brafman and Moshe Tennenholtz. R-max-a general polynomial time algorithm for near-optimal reinforcement learning. *The Journal of Machine Learning Research*, 3:213–231, 2003.

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

Olivier Chapelle and Lihong Li. An empirical evaluation of thompson sampling. In *Advances in neural information processing systems*, pp. 2249–2257, 2011.

Tianqi Chen, Mu Li, Yutian Li, Min Lin, Naiyan Wang, Minjie Wang, Tianjun Xiao, Bing Xu, Chiyuan Zhang, and Zheng Zhang. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv*, 2015.

Wei Chu, Lihong Li, Lev Reyzin, and Robert Schapire. Contextual bandits with linear payoff functions. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 208–214, 2011.

Varsha Dani, Thomas P Hayes, and Sham M Kakade. Stochastic linear optimization under bandit feedback. 2008.

Victor H de la Pena, Michael J Klass, and Tze Leung Lai. Self-normalized processes: exponential inequalities, moment bounds and iterated logarithm laws. *Annals of probability*, pp. 1902–1933, 2004.

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)*, 2003.

Meire Fortunato, Mohammad Gheshlaghi Azar, Bilal Piot, Jacob Menick, Ian Osband, Alex Graves, Vlad Mnih, Remi Munos, Demis Hassabis, Olivier Pietquin, et al. Noisy networks for exploration. *arXiv*, 2017.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, 2016.

Mohammad Ghavamzadeh, Shie Mannor, Joelle Pineau, Aviv Tamar, et al. Bayesian reinforcement learning: A survey. *Foundations and Trends® in Machine Learning*, 2015.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. *arXiv*, 2017.

Daniel Hsu, Sham Kakade, Tong Zhang, et al. A tail inequality for quadratic forms of subgaussian random vectors. *Electronic Communications in Probability*, 17, 2012.

Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 2010.

Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, John Langford, and Robert E Schapire. Contextual decision processes with low bellman rank are pac-learnable. *arXiv*, 2016.

Sham Kakade, Michael J Kearns, and John Langford. Exploration in metric state spaces. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pp. 306–312, 2003.

Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine Learning*, 49(2-3):209–232, 2002.

Michail G Lagoudakis and Ronald Parr. Least-squares policy iteration. *Journal of machine learning research*, 4(Dec):1107–1149, 2003.

Nir Levine, Tom Zahavy, Daniel J Mankowitz, Aviv Tamar, and Shie Mannor. Shallow updates for deep reinforcement learning. *arXiv*, 2017.

Sergey Levine et al. End-to-end training of deep visuomotor policies. *JMLR*, 2016.

Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pp. 661–670. ACM, 2010.

Zachary C Lipton, Kamyar Azizzadenesheli, Abhishek Kumar, Lihong Li, Jianfeng Gao, and Li Deng. Combating reinforcement learning's sisyphean curse with intrinsic fear. *arXiv preprint arXiv:1611.01211*, 2016a.

Zachary C Lipton, Jianfeng Gao, Lihong Li, Xiujun Li, Faisal Ahmed, and Li Deng. Efficient exploration for dialogue policy learning with bbq networks & replay buffer spiking. *arXiv preprint arXiv:1608.05081*, 2016b.

Marlos C Machado, Marc G Bellemare, Erik Talvitie, Joel Veness, Matthew Hausknecht, and Michael Bowling. Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. *arXiv preprint arXiv:1709.06009*, 2017.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 2015.

Ronald Ortner and Daniil Ryabko. Online regret bounds for undiscounted continuous reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 1763–1771, 2012.

Ian Osband, Dan Russo, and Benjamin Van Roy. (more) efficient reinforcement learning via posterior sampling. In *Advances in Neural Information Processing Systems*, 2013.

Ian Osband, Benjamin Van Roy, and Zheng Wen. Generalization and exploration via randomized value functions. *arXiv*, 2014.

Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped dqn. In *Advances in Neural Information Processing Systems*, 2016.

Ian Osband, Daniel Russo, Zheng Wen, and Benjamin Van Roy. Deep exploration via randomized value functions. *arXiv*, 2017.

Georg Ostrovski, Marc G Bellemare, Aaron van den Oord, and Rémi Munos. Count-based exploration with neural density models. *arXiv*, 2017.

Carl Edward Rasmussen and Christopher KI Williams. *Gaussian processes for machine learning*, volume 1. MIT press Cambridge, 2006.

Paat Rusmevichientong and John N Tsitsiklis. Linearly parameterized bandits. *Mathematics of Operations Research*, 35(2):395–411, 2010.

Daniel Russo and Benjamin Van Roy. Learning to optimize via posterior sampling. *Mathematics of Operations Research*, 39(4):1221–1243, 2014.

Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv*, 2016.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 2017.

Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Mostofa Patwary, Mr Prabhat, and Ryan Adams. Scalable bayesian optimization using deep neural networks. In *ICML*, 2015.

Malcolm Strens. A bayesian framework for reinforcement learning. In *ICML*, 2000.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press Cambridge, 1998.

William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 1933.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *AAAI*, 2016.

A APPENDIX

In the main text, for simplicity, we use the terms "BLR" for i.i.d. samples and BLR for non i.i.d. samples exchangeable, even though, technically, the do not have equal meaning. In RL, the data is not i.i.d, and we extend the BLR to non i.i.d. setting by deploying additional Martingale type argument and handles the data with temporal dependency.

Table 3: $1st$ column: score ratio of BDQN to DDQN run for same number of time steps. $2nd$ column: score ratio of BDQN to DDQN$^+$. $3rd$ column: score ratio of BDQN to human scores reported at Mnih et al. (2015). $4th$ column: Area under the performance plot ration (AuPPr) of BDQN to DDQN. AuPPr is the integral of area under the performance plot ration. For Pong, since the scores start form -21, we shift it up by 21. $5th$ column: Sample complexity, *SC*: the number of samples the BDQN requires to beat the human score (Mnih et al., 2015)("$-$" means BDQN could not beat human score). $6th$ column: SC^+: the number of samples the BDQN requires to beat the score of DDQN$^+$. We run both BDQN and DDQN for the same number of times steps, stated in the last column.

Game	$\frac{\text{BDQN}}{\text{DDQN}}$	$\frac{\text{BDQN}}{\text{DDQN}^+}$	$\frac{\text{BDQN}}{\text{HUMAN}}$	AuPPr	SC	C^+	Steps
Amidar	558%	788%	325%	280%	22.9M	4.4M	100M
Alien	103%	103%	43%	110%	-	36.27M	100M
Assault	396%	176%	589%	290%	1.6M	24.3M	100M
Asteroids	2517%	1516%	108%	680%	58.2M	9.7M	100M
Asterix	531%	385%	687%	590%	3.6M	5.7M	100M
BeamRider	207%	114%	150%	210%	4.0M	8.1M	70M
BattleZone	281%	253%	172%	180%	25.1M	14.9M	50M
Atlantis	80604%	49413%	11172%	380%	3.3M	5.1M	40M
DemonAttack	292%	114%	326%	310%	2.0M	19.9M	40M
Centipede	114%	178%	61%	105%	-	4.2M	40M
BankHeist	211%	100%	100%	250%	2.1M	10.1M	40M
CrazyClimber	148%	122%	350%	150%	0.12M	2.1M	40M
ChopperCommand	14500%	1576%	732%	270%	4.4M	2.2M	40M
Enduro	295%	350%	361%	300%	0.82M	0.8M	30M
Pong	112%	100%	226%	130%	1.2M	2.4M	5M

A.1 NETWORK ARCHITECTURE:

The input to the network part of BDQN is $4 \times 84 \times 84$ tensor with a rescaled and averaged over channels of the last four observations. The first convolution layer has 32 filters of size 8 with a stride of 4. The second convolution layer has 64 filters of size 4 with stride 2. The last convolution layer has 64 filters of size 3 followed by a fully connected layer with size 512. We add a BLR layer on top of this.

A.2 CHOICE OF HYPER-PARAMETERS:

For BDQN, we set the values of W^{target} to the mean of the posterior distribution over the weights of BLR with covariances Cov and draw W from this posterior. For the fixed W and W^{target}, we randomly initialize the parameters of network part of BDQN, θ, and train it using RMSProp, with learning rate of 0.0025, and a momentum of 0.95, inspired by (Mnih et al., 2015) where the discount factor is $\gamma = 0.99$, the number of steps between target updates $T^{target} = 10k$ steps, and weights W are re-sampled from their posterior distribution every T^{sample} steps. We update the network part of BDQN every 4 steps by uniformly at random sampling a mini-batch of size 32 samples from the replay buffer. We update the posterior distribution of the weight set W every $T^{Bayes\ target}$ using mini-batch of size B (if the size of replay buffer is less than B at the current step, we choose the minimum of these two), with entries sampled uniformly form replay buffer. The experience replay contains the $1M$ most recent transitions. Further hyper-parameters are equivalent to ones in DQN setting.

Furthermore, for the BLR part of BDQN, we have noise variance σ_ϵ, variance of prior over weights σ, sample size B, posterior update period $T^{Bayes\ target}$, and the posterior sampling period T^{sample}. To optimize for this set of hyper-parameters we set up a very simple, fast, cheap, and non-exhaustive hyper-parameter tuning procedure using a pre-trained DQN model for the game of *Assault*. The simplicity and cheapness of our hyper parameter tuning proves the robustness and superiority of BDQN where the exhaustive hyper-parameter search is likely to provide even better performance. The details of hyper parameters tuning is provided in bellow.

A.3 Hyper-parameters tuning:

For the BLR, we have noise variance σ_ϵ, variance of prior over weights σ, sample size B, posterior update period $T^{Bayes\ target}$, and the posterior sampling period T^{sample}. To optimize for this set of hyper-parameters we set up a very simple, fast, and cheap hyper-parameter tuning procedure which proves the robustness of BDQN. To find the first three, we set up a simple hyper-parameter search. We used a pretrained DQN model for the game of *Assault*, and removed the last fully connected layer in order to have access to its already trained feature representation. Then we tried combination of $B = \{T^{target}, 10 \cdot T^{target}\}$, $\sigma = \{1, 0.1, 0.001\}$, and $\sigma_\epsilon = \{1, 10\}$ and test for 1000 episode of the game. We set these parameters to their best $B = 10 \cdot T^{target}, \sigma = 0.001, \sigma = 1$.

The above hyper-parameter tuning is cheap and fast since it requires only a few times the B number of forwarding passes. For the remaining parameter, we ran BDQN (with weights randomly initialized) on the same game, *Assault*, for $5M$ time steps, with a set of $T^{Bayes\ target} = \{T^{target}, 10 \cdot T^{target}\}$ and $T^{sample} = \{\frac{T^{target}}{10}, \frac{T^{target}}{100}\}$ where BDQN performed better with choice of $T^{Bayes\ target} = 10 \cdot T^{target}$. For both choices of T^{sample}, it performed almost equal where we choose the higher one. We started off with the learning rate of 0.0025 and did not tune for that. Thanks to the efficient Thompson sampling exploration and closed form BLR, BDQN can learn a better policy in an even shorter period of time. In contrast, it is well known for DQN based methods that changing the learning rate causes a major degradation in the performance, 4. The proposed hyper-parameter search is very simple where the exhaustive hyper-parameter search is likely to provide even better performance.

A.4 Learning rate:

It is well known that DQN and DDQN are sensitive to the learning rate and change of learning rate can degrade the performance to even worse than random policy. We tried the same learning rate as BDQN, 0.0025, for DDQN and observed that its performance drops. Fig. 4 shows that the DDQN with higher learning rates learns as good as BDQN at the very beginning but it can not maintain the rate of improvement and degrade even worse than the original DDQN.

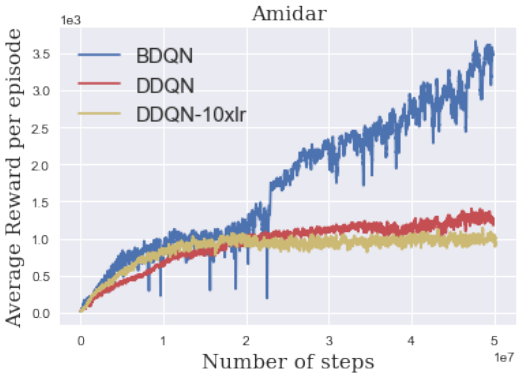

Figure 4: Effect of learning rate on DDQN

A.5 COMPUTATIONAL AND SAMPLE COST COMPARISON:

For a given period of game time, the number of the backward pass in both BDQN and DQN are the same where for BDQN it is cheaper since it has one layer (the last layer) less than DQN. In the sense of fairness in sample usage, for example in duration of $10 \cdot T^{Bayes\ target} = 100k$, all the layers of both BDQN and DQN, except the last layer, sees the same number of samples, but the last layer of BDQN sees 16 times fewer samples compared to the last layer of DQN. The last layer of DQN for a duration of $100k$, observes $25k = 100k/4$ (4 is back prob period) mini batches of size 32, which is $16 \cdot 100k$, where the last layer of BDQN just observes samples size of $B = 100k$. As it is mentioned in Alg. 1, to update the posterior distribution, BDQN draws B samples from the replay buffer and needs to compute the feature vector of them. Therefore, during the duration of $100k$ decision making steps, for the learning procedure, DDQN does $32 * 25k$ of forward passes and $32 * 25k$ of backward passes, while BDQN does same number of backward passes (cheaper since there is no backward pass for the final layer) and $36 * 25k$ of forward passes. One can easily relax it by parallelizing this step along the main body of BDQN or deploying on-line posterior update methods.

A.6 THOMPSON SAMPLING FREQUENCY:

The choice of Thompson sampling update frequency can be crucial from domain to domain. Theoretically, we show that for episodic learning, the choice of sampling at the beginning of each episode, or a bouned number of episodes is desired. If one chooses T^{sample} too short, then computed gradient for backpropagation of the feature representation is not going to be useful since the gradient get noisier and the loss function is changing too frequently. On the other hand, the network tries to find a feature representation which is suitable for a wide range of different weights of the last layer, results in improper waste of model capacity. If the Thompson sampling update frequency is too low, then it is far from being Thompson sampling and losses randomized exploration property. We are interested in a choice of T^{sample} which is in the order of upper bound on the average length of each episode of the Atari games. The current choice of T^{sample} is suitable for a variety of Atari games since the length of each episode is in range of $\mathcal{O}(T^{sample})$ and is infrequent enough to make the feature representation robust to big changes.

For the RL problems with shorter horizon we suggest to introduce two more parameters, $\tilde{T}^{sample}$ and $\tilde{W}$ where $\tilde{T}^{sample}$, the period that of $\tilde{W}$ is sampled our of posterior, is much smaller than T^{sample} and $\tilde{W}$ is being used just for making Thompson sampling actions while W is used for backpropagation of feature representation. For game Assault, we tried using $\tilde{T}^{sample}$ and $\tilde{W}$ but did not observe much a difference, and set them to T^{sample} and W. But for RL setting with a shorter horizon, we suggest using them.

A.7 FURTHER INVESTIGATION IN ATLANTIS:

After removing the maximum episode length limit for the game Atlantis, BDQN gets the score of 62M. This episode is long enough to fill half of the replay buffer and make the model perfect for the later part of the game but losing the crafted skill for the beginning of the game. We observe in Fig. 5 that after losing the game in a long episode, the agent forgets a bit of its skill and loses few games but wraps up immediately and gets to score of $30M$. To overcome this issue, one can expand the replay buffer size, stochastically store samples in the reply buffer where the later samples get stored with lowers chance, or train new models for the later parts of the episode. There are many possible cures for this interesting observation and while we are comparing against DDQN, we do not want to advance BDQN structure-wise.

A.8 FURTHER DISCUSSION ON REPRODUCIBILITY

In Table. 2, we provide the scores of bootstrap DQN (Osband et al., 2016) and NoisyNet[8]Fortunato et al. (2017) along with BDQN. These score are directly copied from their original papers and we did not make any change to them. We also desired to report the scores of count-based method (Ostrovski et al., 2017), but unfortunately there is no table of score in that paper in order to provide them here.

[8]This work does not have scores of Noisy-net with DDQN objective function but it has Noisy-net with DQN objective which are the scores reported in Table. 2

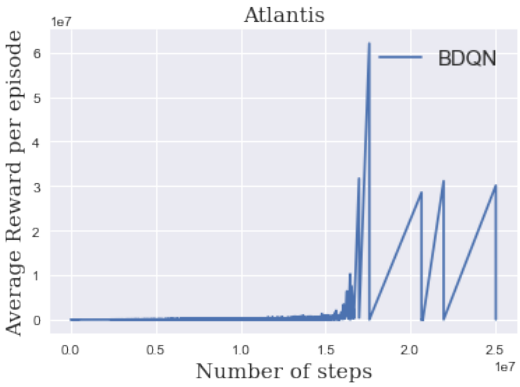

Figure 5: BDQN on Atlantis after removing the limit on max of episode length hits the score of $62M$ in $16M$ samples.

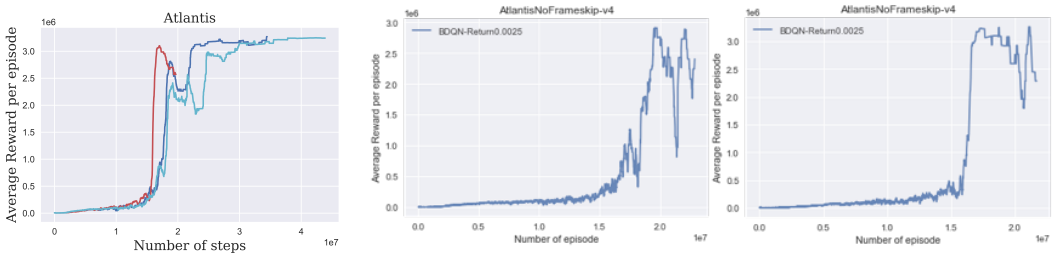

Figure 6: A couple of more runs of BDQN where the jump around $15M$ constantly happens

In order to make it easier for the readers to compare against the results in Ostrovski et al. (2017), we visually approximated their plotted curves for CTS, $Pixel$, $Pixel$, and $Reactor$, and added them to the Table. 2. We added these numbers just for the convenience of the readers. Surly we do not argue any scientific meaning for them and leave it to the readers to interpret them.

Table. 2 shows a significant improvement of BDQN over these baselines by looking at Table. 2. Despite the simplicity and negligible computation overhead of BDQN over DDQN, we can not scientifically claim that BDQN outperforms these baselines by just looking at the scores in Table.2 because we are not aware of their detailed implementation as well as environment detail. For example, in this work, we directly implemented DDQN by following the implementation details mentioned in the original DDQN paper and the scores of our DDQN implementation during the evaluation time almost matches the scores of DDQN reported in the original paper. But the reported scores of implemented DDQN in Osband et al. (2016) are much different from the reported score in the original DDQN paper.

A discussion on safety In BDQN, as mentioned in Eq. 3, the prior and likelihood are conjugate of each others. Therefore, we have a closed form posterior distribution of the discounted return, $\sum_{t=0}^{N} \gamma^t r_t | x_0 = x, a_0 = a, \mathcal{D}_a$, is approximated as

$$\mathcal{N}\left(\frac{1}{\sigma_\epsilon^2}\phi^\theta(x)^\top \Xi_a \Phi_a^\theta \mathbf{y}_a, \phi^\theta(x)^\top \Xi_a \phi^\theta(x)\right)$$

One can use this distribution and come up with a criterion for safe RL by just looking at the low and high probability events under different actions.

B BAYESIAN REGRET, PROOF OF THEOREM 1

Let's consider the following linear model over the Q functions and returns for episodic MDPs with episode length of H. (We consider the discount factor to be any number $0 \leq \gamma \leq 1$)

$$\Psi \mathbf{R} = \mathbf{\Phi}(x^{\{1:H\}}, a^{\{1:H\}})\omega + \Psi\nu(x^{\{1:H\}}, a^{\{1:H\}}) \ , \ \textit{where} \ , \ \Psi := \begin{bmatrix} 1 & \gamma & \gamma^2 & \dots & \gamma^{H-1} \\ 0 & 1 & \gamma & \dots & \gamma^{H-2} \\ 0 & 0 & 1 & \dots & \vdots \\ \vdots & \dots & \dots & \dots & \vdots \\ 0 & \dots & \dots & \dots & 1 \end{bmatrix} \tag{6}$$

Where the Q function is a linear function of $\phi(x, a) \in \mathcal{R}^d$, i.e. $Q(x, a) := \phi(x, a)^\top \omega$ (for the theoretical analysis, we use ω instead of ω to avoid any possible confusion). In order to study the most general setting, we also represent actions through the same feature representation. The vector $\mathbf{R} \in \mathbb{R}^H$ is the random vector of rewards in a episode. The matrix $\mathbf{\Phi}(x^{\{1:H\}}, a^{\{1:H\}}) \in \mathbb{R}^{H \times d}$ is a row-wised concatenation of sequence of state-action features, $\phi(x, a)$ under a given policy, also any instance of the feature representing matrix, $trace\left(\mathbf{\Phi}(x^{\{1:H\}}, a^{\{1:H\}})\mathbf{\Phi}(x^{\{1:H\}}, a^{\{1:H\}})^\top\right) \leq L$, *for any*$x^{\{1:H\}}, a^{\{1:H\}}$. The vector $\nu(x^{\{1:H\}}, a^{\{1:H\}}) \in \mathbb{R}^H$is a state-action dependent noise vector where, for any h', the noises after time step h', $\nu^{h=i \geq h'}$, are independent from the noises before time step h', $\nu^{h=i<h'}$ condition on (x^h, a^h), the state action pair at time step h. Moreover, since the underlying model of environment is assumed to be an MDP, the noise at time step h, ν^h models the induces noise by stochasticity of transition distribution to state x^h, i.e. $f(x^h|x^{h-1}, a^{h-1})$, and the stochasticity of reward at time step h, i.e. $r(x^h, a^h)$. In the Bayesian approach, we consider a Gaussian prior over the parameter vector $\omega \sim \mathcal{N}(0, \lambda I)$, then exploit the samples to compute the posterior. After computing the posterior at time t we sample a vector ω_t, then follow the policy induced by ω_t, i.e. π_t. We then collect the resulting trajectory to update the posterior and the policy for next episode.

In the following, we show that, given the true underlying model has parameter ω^*, after T episodes of following TS, the estimated $\hat{b}_t$ concentrates around its true value ω^*. The main body of the following analyses on concentration of measure is a matrix extension of contextual linear bandit analyses Abbasi-Yadkori et al. (2011); Chu et al. (2011); Li et al. (2010); Rusmevichientong & Tsitsiklis (2010); Dani et al. (2008) and self normalized processes(de la Pena et al., 2004). Our regret analysis extends prior analysis in frequentist and Bayesian regret literature Osband et al. (2013); Jaksch et al. (2010).

In order to provide more intuition on the model, we can exploit the property of the matrix Ψ where

$$\Psi^{-1} = \begin{bmatrix} 1 & -\gamma & 0 & \dots & 0 \\ 0 & 1 & \text{-}\gamma & \dots & 0 \\ 0 & 0 & 1 & \dots & \vdots \\ \vdots & \dots & \dots & \dots & \vdots \\ 0 & \dots & \dots & \dots & 1 \end{bmatrix}$$

and rewrite Eq. 6 as follows:

$$\mathbf{R}_t = \Psi^{-1}\mathbf{\Phi}_t\omega + \nu_t \tag{7}$$

We denote $\Psi' := {\Psi^{-1}}^\top$. For the sake of analysis, we use the following formulation;

$$\overline{\mathbf{R}}_t := \Psi\mathbf{R}_t = \mathbf{\Phi}_t\omega + \overline{\nu}_t \ , \quad \textit{where} \quad \overline{\nu}_t := \Psi\nu_t \tag{8}$$

Assumption 1 (Sub-Gaussian random vector (Hsu et al., 2012))**.** *The noise model at time t, conditioned on the filtration at time* $t-1$*,* $\mathcal{F}_{t-1}$ *is sub-Gaussian random vector, i.e. there exists a parameter* $\sigma \geq 0$ *such that* $\forall \alpha \in \mathcal{R}^d$

$$\mathbb{E}\left[\exp\left(\alpha^\top \overline{\nu}_t\right) \middle| \mathcal{F}_{t-1}\right] \leq \exp\left(\|\alpha\|_2 \sigma^2/2\right)$$

Asm. 1 also implies that $\mathbb{E}\left[\overline{\nu}_t \middle| \mathcal{F}_{t-1}\right] = 0$ which means that after drawing the matrix $\mathbf{\Phi}_t$, the generative model for $\overline{\nu}_t$ is means zero. A imilar assumption on the noise model is assumed in prior analyses of linear bandit (Abbasi-Yadkori et al., 2011).

Lemma 1. *Let $\alpha \in \mathcal{R}^d$ be be an arbitrary vector and for any $t \geq 0$ define*

$$M_t^\alpha := \exp\left(\sum_i^t \left[\frac{\alpha^\top \mathbf{\Phi}_i^\top \overline{\nu}_i}{\sigma} - \frac{1}{2}\|\mathbf{\Phi_i}\alpha\|_2^2\right]\right)$$

Then, for a stopping time under the filtration $\{\mathcal{F}_t\}_{t=0}^\infty$, $M_\tau^\lambda \leq 1$.

Proof. Lemma. 1 We first show that $\{M_t^\alpha\}_{t=0}^\infty$ is a supermartingale sequence. Let

$$D_i^\alpha = \exp\left(\frac{\alpha^\top \mathbf{\Phi}_i^\top \overline{\nu}_i}{\sigma} - \frac{1}{2}\|\mathbf{\Phi_i}\alpha\|_2^2\right)$$

Therefore, we can rewrite $\mathbb{E}\left[M_t^\alpha\right]$ as follows:

$$\mathbb{E}\left[M_t^\alpha|\mathcal{F}_{t-1}\right] = \mathbb{E}\left[D_1^\alpha \ldots D_{t-1}^\alpha D_t^\alpha|\mathcal{F}_{t-1}\right] = D_1^\alpha \ldots D_{t-1}^\alpha \mathbb{E}\left[D_t^\alpha|\mathcal{F}_{t-1}\right] \leq M_{t-1}^\alpha$$

The last inequality follows since $\mathbb{E}\left[D_t^\alpha|\mathcal{F}_{t-1}\right] \leq 1$ due to Assumption 1, therefore since for the first time step $\mathbb{E}\left[M_1^\alpha\right] \leq 1$, then $\mathbb{E}\left[M_t^\alpha\right] \leq 1$. For a stopping time τ, Define a variable $\overline{M}_t^\alpha = M_{\min\{t,\tau\}}^\alpha$ and since $\mathbb{E}\left[M_\tau^\alpha\right] = \mathbb{E}\left[\liminf_{t\to\infty} \overline{M}_t^\alpha\right] \leq \liminf_{t\to\infty} \mathbb{E}\left[\overline{M}_t^\alpha\right] \leq 1$, therefore the Lemma. 1 holds. □

Let's defined the following quantities for self-normalized processes;

$$S_t := \sum_i^t \mathbf{\Phi}_i^\top \overline{\nu}_i, \quad \chi_t := \sum_{i=0}^t \mathbf{\Phi}_i^\top \mathbf{\Phi}_i, \quad \overline{\chi}_t = \chi_t + \widetilde{\chi}$$

where $\widetilde{\chi}$ is a ridge regularization matrix and usually is equal to λI.

Lemma 2. *[Extension to Self-normalized bound in Abbasi-Yadkori et al. (2011)] For a stopping time τ and filtration $\{\mathcal{F}_t\}_{t=0}^\infty$, with probability at least $1-\delta$*

$$\|S_\tau\|_{\overline{\chi}_\tau^{-1}} \leq 2\sigma^2 \log(\frac{\det\left(\overline{\chi}_t\right)^{1/2} \det(\widetilde{\chi})}{\delta})$$

Proof. Lemma. 2 Given the definition of the parameters of self-normalized process, we can rewrite M_t^α as follows;

$$M_t^\alpha = \exp\left(\frac{\alpha^\top S_t}{\sigma} - \frac{1}{2}\|\alpha\|_{\chi_t}\right)$$

Consider Ω as a Gaussian random vector, denoted as $f(\Omega = \alpha)$, with covariance matrix of $\overline{\chi}^{-1}$. Define $M_t := \mathbb{E}\left[M_t^\Omega|\mathcal{F}_\infty\right]$, which also results in $\mathbb{E}\left[M_t\right] = \mathbb{E}\left[\mathbb{E}\left[M_t^\Omega|\Omega\right]\right] \leq 1$. Therefore

$$\begin{aligned}
M_t &= \int_{\mathbb{R}^d} \exp\left(\frac{\alpha^\top S_t}{\sigma} - \frac{1}{2}\|\alpha\|_{\chi_t}\right) f(\alpha) d\alpha \\
&= \int_{\mathbb{R}^d} \exp\left(\frac{1}{2}\|\alpha - \chi_t^{-1} S_t/\sigma\|_{\chi_t}^2 + \frac{1}{2}\|S_t/\sigma\|_{\chi_t^{-1}}^2\right) f(\alpha) d\alpha \\
&= \sqrt{\frac{\det\left(\widetilde{\chi}\right)}{(2\pi)^d}} \exp\left(\frac{1}{2}\|S_t/\sigma\|_{\chi_t^{-1}}\right) \int_{\mathbb{R}^d} \exp\left(\frac{1}{2}\|\alpha - \chi_t^{-1} S_t/\sigma\|_{\chi_t}^2 + \frac{1}{2}\|\alpha\|_{\widetilde{\chi}}^2\right) d\alpha
\end{aligned}$$

Since χ_t and $\widetilde{\chi}$ are positive semi definite and positive definite respectively, we have

$$\begin{aligned}
\|\alpha - \chi_t^{-1} S_t/\sigma\|_{\chi_t}^2 + \|\alpha\|_{\widetilde{\chi}}^2 &= \|\alpha - \left(\widetilde{\chi} + \chi_t^{-1}\right) S_t/\sigma\|_{\widetilde{\chi}+\chi_t}^2 + \|\chi_t^{-1} S_t/\sigma\|_{\chi_t}^2 - \|S_t/\sigma\|_{(\widetilde{\chi}+\chi_t)^{-1}}^2 \\
&= \|\alpha - \left(\widetilde{\chi} + \chi_t^{-1}\right) S_t/\sigma\|_{\widetilde{\chi}+\chi_t}^2 + \|\mathcal{S}_t/\sigma\|_{\chi_t^{-1}}^2 - \|S_t/\sigma\|_{(\widetilde{\chi}+\chi_t)^{-1}}^2
\end{aligned}$$

Therefore,

$$M_t = \sqrt{\frac{\det(\widetilde{\chi})}{(2\pi)^d}} \exp\left(\frac{1}{2}\|S_t/\sigma\|_{(\widetilde{\chi}+\chi_t)^{-1}}\right) \int_{\mathbb{R}^d} \exp\left(\frac{1}{2}\|\alpha - \left(\widetilde{\chi}+\chi_t^{-1}\right) S_t/\sigma\|^2_{\widetilde{\chi}+\chi_t}\right) d\alpha$$
$$= \left(\frac{\det(\widetilde{\chi})}{\det(\widetilde{\chi}+\chi_t)}\right)^{1/2} \exp\left(\frac{1}{2}\|S_t/\sigma\|^2_{(\widetilde{\chi}+\chi_t)^{-1}}\right)$$

Since $\mathbb{E}[M_\tau] \leq 1$ we have

$$\mathbb{P}\left(\|S_\tau\|^2_{(\widetilde{\chi}+\chi_\tau)^{-1}} \leq 2\sigma \log\left(\frac{\det(\widetilde{\chi}+\chi_\tau)^{1/2}}{\delta \det(\widetilde{\chi})^{1/2}}\right)\right) \leq \frac{\mathbb{E}\left[\exp\left(\frac{1}{2}\|S_\tau/\sigma\|^2_{(\widetilde{\chi}+\chi_\tau)^{-1}}\right)\right]}{\left(\frac{\det(\widetilde{\chi}+\chi_\tau)}{\delta \det(\widetilde{\chi})}\right)^{1/2}} \leq \delta$$

Where the Markov inequality is deployed for the final step. The stopping is considered to be the time step as the first time in the sequence when the concentration in the Lemma. 2 does not hold. □

Regret Decomposition Let's restate the definition of the expected regret and expected Bayesian regret.

$$\mathbf{Reg}_T := \sum_t^T V_{\pi^*}^{\omega^*} - V_{\pi_t}^{\omega^*}$$

where π_t is the deployed policy at episode t induced by ω^t, $\pi_t(x,a) = a = \arg\max_{a'} \omega_t \phi(x,a')$ and V_π^ω is the value of policy π on model derived by ω[9]. The RHS for model based counterpart is $V_{\pi^*}^{M^*} - V_{\pi_t}^{M^*}$. When there is a prior over the model choice ω^* the expected Bayesian Regret might be target of study.

$$\mathbf{BayesReg}_T := \mathbb{E}_{\omega^*}[\mathbf{Reg}_T] := \mathbb{E}_{\omega*}\left[\sum_t^T V_{\pi^*}^{\omega^*}(x_t^0) - V_{\pi_t}^{\omega^*}(x_t^0)\right]$$
$$= \sum_t^T \mathbb{E}_{\mathcal{H}_t}\left[\mathbb{E}_{\omega^*}\left[V_{\pi^*}^{\omega^*}(x_t^0) - V_{\pi_t}^{\omega^*}(x_t^0)\middle|\mathcal{H}_t\right]\right]$$
$$= \sum_t^T \mathbb{E}_{\mathcal{H}_t}\left[\mathbb{E}_{\omega^*}\left[V_{\pi^*}^{\omega^*}(x_t^0) - V_{\pi_t}^{\omega^t}(x_t^0)\middle|\mathcal{H}_t\right]\right] + \mathbb{E}_{\mathcal{H}_t}\left[\mathbb{E}_{\omega^*}\left[V_{\pi_t}^{\omega_t}(x_t^0) - V_{\pi_t}^{\omega^*}(x_t^0)\middle|\mathcal{H}_t\right]\right]$$

Here $\mathcal{H}_t$ is a multivariate random sequence which indicates history at the beginning of episode t, including x_t^0, and ω_t is the model parameter chosen through Thompson Sampling. Since the ω_t and ω^* are equally distributed conditioned on $\mathbb{H}_t$, the term $\left[\mathbb{E}_{\omega^*}\left[V_{\pi^*}^{\omega^*} - V_{\pi_t}^{\omega_t}\middle|\mathcal{H}_t\right]\right] = 0$ and as a consequence, the first term in the RHS of the last equality is zero. (This term of optimism analysis, when ω_t is the optimistic choice is zero.)

For a given state x_t^0, the value of the optimal policy π_t of model ω^t under the very same model ω^t is equal to

$$V_{\omega_t}(x_t^0, \pi_t) = \phi(x_t^0, \pi_t(x_t^0))^\top \omega_t = \mathbb{E}\left[r_{\omega_t}(x_t^0, \pi_t(x_t^0))\right] + \int_{x'} T_{\omega_t}(x'|x_t^0, \pi_t(x_t^0)) V_{\omega_t}(x', \pi_t)$$

and for π^* under model ω^* is equal to

$$V_{\omega^*}(x_t^0, \pi^*) = \phi(x_t^0, \pi^*(x_t^0))^\top \omega^* = \mathbb{E}\left[r_{\omega^*}(x_t^0, \pi^*(x_t^0))\right] + \int_{x'} T_{\omega^*}(x'|x_t^0, \pi^*(x_t^0)) V_{\omega^*}(x', \pi^*)$$

[9] In general, the analysis also hold for general class of policies (stochastic) as long as the mapping from ω to policy given and fixed for oracle and the RL algorithm

while $\phi(x_t^0, \pi_t(x_t^0))^\top \omega_t$ is value of π_t under model ω_t at state x_t^0 which is also the optimal value at state x_t^0 under the model ω_t, $\phi(x_t^0, \pi^*(x_t^0))^\top \omega_t$ is the value of following $\pi^*(x_t^0)$ for one step and then follow π_t for the remaining part of horizon.

Furthermore, the value of π_t under model ω^* is as follows:

$$V_{\omega^*}(x_t^0, \pi_t) = \mathbb{E}\left[r_{\omega^*}(x_t^0, \pi_t(x_t^0))\right] + \int_{x'} T_{\omega^*}(x'|x_t^0, \pi_t(x_t^0)) V_{\omega^*}(x', \pi_t)$$

Therefore, we can write the upper bound on the $\mathbf{BayesReg}_T$;

$$\mathbf{BayesReg}_T \leq \sum_t^T \mathbb{E}_{\mathcal{H}_t}\left[\mathbb{E}_{\omega^*}\left[\phi(x_t^0, \pi_t(x_t^0))^\top \omega_t - V_{\pi_t}^{\omega^*}(x_t^0) \middle| \mathcal{H}_t\right]\right]$$

Since

$$\phi(x_t^0, \pi_t(x_t^0))^\top \omega^* = \mathbb{E}\left[r_{\omega^*}(x_t^0, \pi_t(x_t^0))\right] + \int_{x'} T_{\omega^*}(x'|x_t^0, \pi_t(x_t^0)) V_{\omega^*}(x', \pi^*)$$

the RHS term can be decomposed as follows;

$$\begin{aligned}
&\phi(x_t^0, \pi_t(x_t^0))^\top \omega_t - V_{\pi_t}^{\omega^*}(x_t^0) = \\
&\quad \phi(x_t^0, \pi_t(x_t^0))^\top \omega_t - \phi(x_t^0, \pi_t(x_t^0))^\top \omega^* + \phi(x_t^0, \pi_t(x_t^0))^\top \omega^* - V_{\pi_t}^{\omega^*}(x_t^0) \\
&\quad = \phi(x_t^0, \pi_t(x_t^0))^\top (\omega_t - \omega^*) + \phi(x_t^0, \pi_t(x_t^0))^\top \omega^* - V_{\pi_t}^{\omega^*}(x_t^0) \\
&\quad = \phi(x_t^0, \pi_t(x_t^0))^\top (\omega_t - \omega^*) + \int_{X_t^1} T_{\omega^*}(X_t^1|x_t^0, \pi_t(x_t^0)) \left(\underbrace{V_{\omega^*}(X_t^1, \pi^*) - V_{\omega^*}(X_t^1, \pi_t)}_{\text{(a)}}\right)
\end{aligned}$$

where (a) can be decomposed the same way as $V_{\omega^*}(x_t^0, \pi^*) - V_{\omega^*}(x_t^0, \pi_t)$. The remaining is to show each of these parts are bounded.

Lemma 3 (Confidence ellipsoid for problem in Eq. 6)**.** *For a matrix problem in Eq. 6, given a sequence of $\{\mathbf{\Phi}_i, \mathbf{R}_i\}_{i=1}^t$, under the Asm. 1 and the following estimator for $\widehat{\omega}_t$*

$$\widehat{\omega}_t := \left(\sum_i^t \mathbf{\Phi}_i^\top \mathbf{\Phi}_i + \lambda I\right)^{-1} \sum_i^t \left(\mathbf{\Phi}_i^\top \overline{\mathbf{R}}_i\right)$$

with probability at least $1 - \delta$

$$\|\widehat{\omega}_t - \omega^*\|_{\overline{\chi}_t} \leq \sigma \sqrt{2 \log\left(\frac{1 + tL^2/\lambda}{\delta}\right)} + \lambda^{1/2} L_\omega$$

where $\|\omega^*\|_2 \leq L_\omega$ and $trace \mathbf{\Phi}(\cdot)\mathbf{\Phi}(\cdot)^\top \leq L$.

Proof of Lemma 3. Given the estimator $\widehat{\omega}_t$ we have the following:

$$\begin{aligned}
\widehat{\omega}_t &= \left(\sum_i^t \mathbf{\Phi}_i^\top \mathbf{\Phi}_i + \lambda I\right)^{-1} \sum_i^t \left(\mathbf{\Phi}_i^\top \left(\mathbf{\Phi}_i \omega^* + \overline{\nu}_i\right)\right) \\
&= \left(\sum_i^t \mathbf{\Phi}_i^\top \mathbf{\Phi}_i + \lambda I\right)^{-1} \sum_i^t \left(\mathbf{\Phi}_i^\top \overline{\nu}_i\right) \\
&+ \left(\sum_i^t \mathbf{\Phi}_i^\top \mathbf{\Phi}_i + \lambda I\right)^{-1} \left(\sum_i^t \mathbf{\Phi}_i^\top \mathbf{\Phi}_i + \lambda I\right) \omega^* \\
&- \lambda \left(\sum_i^t \mathbf{\Phi}_i^\top \mathbf{\Phi}_i + \lambda I\right)^{-1} \omega^*
\end{aligned}$$

therefore, for any vector $\zeta \in \mathbb{R}^d$

$$\zeta^\top \widehat{\omega}_t - \zeta^\top \omega^* = \zeta^\top \left(\sum_i^t \mathbf{\Phi}_i^\top \mathbf{\Phi}_i + \lambda I \right)^{-1} \sum_i^t \left(\mathbf{\Phi}_i^\top \overline{\nu}_i \right) - \zeta^\top \lambda \left(\sum_i^t \mathbf{\Phi}_i^\top \mathbf{\Phi}_i + \lambda I \right)^{-1} \omega^*$$

As a results, applying Cauchy-Schwarz inequality and inequalities $\|\omega^*\|_{\overline{\chi}_t^{-1}} \leq \frac{1}{\lambda(\overline{\chi}_t)} \|\omega^*\|_2 \leq \frac{1}{\lambda} \|\omega^*\|_2$ we get

$$|\zeta^\top \widehat{\omega}_t - \zeta^\top \omega^*| = \|\zeta\|_{\overline{\chi}_t^{-1}} \| \sum_i^t \left(\mathbf{\Phi}_i^\top \overline{\nu}_i \right) \|_{\overline{\chi}_t^{-1}} - \lambda \|\zeta\|_{\overline{\chi}_t^{-1}} \|\omega^*\|_{\overline{\chi}_t^{-1}}$$
$$\leq \|\zeta\|_{\overline{\chi}_t^{-1}} \| \left(\| \sum_i^t \left(\mathbf{\Phi}_i^\top \overline{\nu}_i \right) \|_{\overline{\chi}_t^{-1}} + \lambda^{1/2} \|\omega^*\|_2 \right)$$

where applying self normalized Lemma 2, with probability at least $1 - \delta$

$$|\zeta^\top \widehat{\omega}_t - \zeta^\top \omega^*| \leq \|\zeta\|_{\overline{\chi}_t^{-1}} \left(2\sigma \log \left(\frac{\det \left(\overline{\chi}_t \right)^{1/2}}{\delta \det \left(\widetilde{\chi} \right)^{1/2}} \right) + \lambda^{1/2} L_\omega \right)$$

hold for any ζ. By plugging in $\zeta = \overline{\chi}_t \left(\widehat{\omega}_t - \omega^* \right)$ we get the following;

$$\|\widehat{\omega}_t - \omega^*\|_{\overline{\chi}_t} \leq \theta_t(\delta) := \sigma \sqrt{2 \log \left(\frac{\det(\overline{\chi}_t)^{1/2} \det(\lambda I)^{1/2}}{\delta} \right)} + \lambda^{1/2} L_\omega$$

The $\det(\overline{\chi}_t)$ can be written as $\det(\overline{\chi}_t) = \prod_j^d \alpha_j$ therefore, $trace(\overline{\chi}_t) = \sum_j^d \alpha_j$. We also know that;

$$(\prod_j^d \alpha_j)^{1/d} \leq \frac{\sum_j^d \alpha_j}{d}$$

A matrix extension to Lemma 10 and 11 in Abbasi-Yadkori et al. (2011) resutls in $\det(\overline{\chi}_t) \leq \left(\frac{trace(\overline{\chi}_t)}{d} \right)^d$ while we have $trace(\overline{\chi}_t) = trace(\lambda I) + \sum_i^t \mathbf{\Phi}_i^\top \mathbf{\Phi}_i \leq d\lambda + tL$, therefore the main statement of Lemma 3 goes through. □

For the regret, condition on $\mathcal{H}_t$, then drawn $\mathbf{\Phi}_t$ following π_t

$$\phi(x_t^0, \pi_t(x_t^0))^\top \omega_t - V_{\pi_t}^{\omega^*}(x_t^0)$$
$$= \mathbb{1}^\top \mathbf{\Phi}_t \left(\omega_t - \omega^* \right) = \mathbb{1}^\top \mathbf{\Phi}_t \overline{\chi}_t^{-1/2} \overline{\chi}_t^{1/2} \left(\omega_t - \omega^* \right) \leq \sqrt{H} \|\mathbf{\Phi}_t\|_{\overline{\chi}_t^{-1}} \|\omega_t - \omega^*\|_{\overline{\chi}_t}$$

Therefore, by running the TS, we get a distribution over the whole history and for a given history we have;

$$\textbf{BayesReg}_T \leq \sqrt{H} \mathbb{E} \left[\sum_t^T \|\mathbf{\Phi}_t\|_{\overline{\chi}_t^{-1}} \|\omega_t - \omega^*\|_{\overline{\chi}_t} \right]$$
$$\leq \sqrt{H} \mathbb{E} \left[\sum_t^T \|\mathbf{\Phi}_t\|_{\overline{\chi}_t^{-1}} \|\omega_t - \widehat{\omega}\|_{\overline{\chi}_t} + \|\mathbf{\Phi}_t\|_{\overline{\chi}_t^{-1}} \|\omega^* - \widehat{\omega}\|_{\overline{\chi}_t} \right]$$
$$\leq 2\sqrt{H} \mathbb{E} \left[\sum_t^T \theta_t(\delta) \|\mathbf{\Phi}_t\|_{\overline{\chi}_t^{-1}} \right]$$

Since we know assumed the maximum cumulative return of each episode is as most 1, we have;

$$\textbf{BayesReg}_T \leq 2\sqrt{H} \mathbb{E} \left[\sum_t^T \min\{\theta_t(\delta) \|\mathbf{\Phi}_t\|_{\overline{\chi}_t^{-1}}, 1\} \right] \leq 2\sqrt{H} \mathbb{E} \left[\sum_t^T \theta_t(\delta) \min\{\|\mathbf{\Phi}_t\|_{\overline{\chi}_t^{-1}}, 1\} \right]$$

Moreover, at time T, we can use Jensen's inequality, exploit the fact that $\theta_t(\delta)$ is an increasing function of t and have

$$\begin{aligned}\textbf{BayesReg}_T &\leq 2\sqrt{H}\mathbb{E}\left[\sqrt{T\theta_T(\delta)^2\sum_t^T \min\{\|\mathbf{\Phi}_t\|_{\overline{\chi}_t^{-1}}, 1\}}\right] \\ &\leq 2\sqrt{H}\left(\sigma\sqrt{2\log\left(\frac{1+tL^2/\lambda}{\delta}\right)} + \lambda^{1/2}L_\omega\right)\mathbb{E}\left[\sqrt{T\sum_t^T \min\{\|\mathbf{\Phi}_t\|_{\overline{\chi}_t^{-1}}, 1\}}\right] \quad (9)\end{aligned}$$

Using the fact that for any scalar α such that $0 \leq \alpha \leq 1$, then $\alpha \leq 2\log(1+\alpha)$, we can rewrite the latter part of Eq. 9

$$\sqrt{\sum_t^T \min\{\|\mathbf{\Phi}_t\|_{\overline{\chi}_t^{-1}}, 1\}} \leq 2\sum_t^T \log\left(1 + \|\mathbf{\Phi}_t\|_{\overline{\chi}_t^{-1}}\right)$$

In order to bound the RHS of the this inequality, we use generalized matrix determinant lemma.

$$\begin{aligned}\det(\overline{\chi}_T) &= \det\left(\overline{\chi}_{T-1} + \mathbf{\Phi}_T^\top\mathbf{\Phi}_T\right) \\ &= \det\left(I_H + \mathbf{\Phi}_T\overline{\chi}_t^{-1}\mathbf{\Phi}_T^\top\right)\det\left(\overline{\chi}_{T-1}\right) \\ &= \prod_t^T \det\left(I_H + \mathbf{\Phi}_t\overline{\chi}_t^{-1}\mathbf{\Phi}_t^\top\right)\det(\widetilde{\chi})\end{aligned}$$

From linear algebra, and given a time step t, we know the characteristic polynomial of the matrix $\mathbf{\Phi}_t\overline{\chi}_t^{-1}\mathbf{\Phi}_t^\top$ is $CP_t(\overline{\lambda}) = \det(\overline{\lambda}I_H - \mathbf{\Phi}_t\overline{\chi}_t^{-1}\mathbf{\Phi}_t^\top)$. If the set $\{\lambda_1, \lambda_2, \ldots, \lambda_d\}$ represent the eigenvalues of $\mathbf{\Phi}_t\overline{\chi}_t^{-1}\mathbf{\Phi}_t^\top$ where $\lambda_i \geq \lambda_j$ if $i \geq j$, one can rewrite $CP_t(\overline{\lambda})$ as follows;

$$CP(\overline{\lambda}) = (\overline{\lambda} - \lambda_1)(\overline{\lambda} - \lambda_2)\cdots(\overline{\lambda} - \lambda_H)$$

by adding the matrix I_H to $\mathbf{\Phi}_t\overline{\chi}_t^{-1}\mathbf{\Phi}_t^\top$ where $\lambda_i \geq \lambda)j$, we get $CP_t'(\overline{\lambda}) = \det(\overline{\lambda}I_H - I_H - \mathbf{\Phi}_t\overline{\chi}_t^{-1}\mathbf{\Phi}_t^\top)$, therefore;

$$CP_t'(\overline{\lambda}) = (\overline{\lambda} - 1 - \lambda_1)(\overline{\lambda} - 1 - \lambda_2)\cdots(\overline{\lambda} - 1 - \lambda_H)$$

and the eigenvalues of $I_H + \mathbf{\Phi}_t\overline{\chi}_t^{-1}\mathbf{\Phi}_t^\top$ are shifted eigenvalues of $\mathbf{\Phi}_t\overline{\chi}_t^{-1}\mathbf{\Phi}_t^\top$ by $+1$, i.e. $\{\lambda_1 + 1, \lambda_2 + 1, \ldots, \lambda_d + 1\}$. Since $\xi^\top\mathbf{\Phi}_t\overline{\chi}_t^{-1}\mathbf{\Phi}_t^\top\xi = \left(\mathbf{\Phi}\overline{\chi}_t^{-1/2\top}\xi\right)^\top\left(\mathbf{\Phi}\overline{\chi}_t^{-1/2\top}\xi\right)$ all the eigenvalues $\{\lambda_1, \lambda_2, \ldots, \lambda_d\}$ are non-negative. As a results,

$$\det\left(I_H + \mathbf{\Phi}_t\overline{\chi}_t^{-1}\mathbf{\Phi}_t^\top\right) \geq \left(1 + \|\mathbf{\Phi}_t\|_{\overline{\chi}_t^{-1}}\right)$$

As a consequence;

$$\det(\overline{\chi}_T) \geq \det(\widetilde{\chi})\prod_t^T\left(1 + \|\mathbf{\Phi}_t\|_{\overline{\chi}_t^{-1}}\right)$$

Therefore,

$$2\sum_t^T \log\left(1 + \|\mathbf{\Phi}_t\|_{\overline{\chi}_t^{-1}}\right) \leq 2\left(\log\left(\det(\overline{\chi}_T)\right) - \log\left(\det(\widetilde{\chi})\right)\right) \leq 2d\log(\lambda + TL^2/d) \quad (10)$$

Substituting the RHS of Eq. 10 into Eq. 9, we get

$$\textbf{BayesReg}_T \leq 2\sqrt{H}\left(\sigma\sqrt{2d\log\left(\frac{1+TL^2/\lambda}{\delta}\right)} + \lambda^{1/2}L_\omega\right)\sqrt{2Td\log(\lambda + TL^2/d)}$$

which is the the statement in the Theorem.