Exploration Methods Based on Value Functions

Navid Ardeshir (na2844)

Statistics Department

October 2020

Abstract

This is an overview on some exploration methods based on the value function in online-Reinforcement Learning. Methods, such as Ensemble sampling, Randomized Value Functions Value Iteration (RLSVI), and Randomized Prior Functions (RPF) are included. In this report, we provide a simple comparison between these methods and attempt to give a theoretical bound for the ensemble version of RLSVI by exhibiting a connection to the classical Bandit Problem.

1 Introduction:

Recent problems more or less are involved with huge state space representations along with a sparse reward distribution which demands generalization and exploration. To that end, an efficient algorithm requires to gather useful data (containing informative and rewarding states) and learn from that experience subsequently. Essentially, having a well performed RL agent is immensely tied with the quality of exploration. This issue has attracted a bulk of work towards this subject.

Classical exploration methods such as Dithering (i.e. ϵ-greedy, Boltzmann, and etc.) are widely used in practice but they suffer from lack of memory over actions that are known to be inferior. To reiterate, exploratory actions are those which some information will be gained upon taking them. Trivially, an action that we are certain that leads to an undesirable trajectory is not worth exploring. Moreover, it is quite easy to construct an environment so that ϵ-greedy fails to explore in poly-time. Alternatives such as ϵ-annealing were introduced in order to subside the issue by decreasing ϵ value over time as we get more certain regarding value function estimates. However, in order to resolve this critical issue entirely we need a more rigorous way to measure uncertainty.

Indeed, the environment is not fully known a-priori and the goal is to estimate the corresponding value functions. From a statistical point of view, there are two natural ways to proceed:

- **Frequentist**: Based on the existing history and observed values we may construct an empirical model of the environment for which we can apply classical methods (e.g. Value Iteration + Dithering) to obtain an estimate for value functions. Moreover, we are able to introduce uncertainty by constructing confidence sets surrounding the observed MDP. Methods such as Optimism in The Face of Uncertainty [5] follows this approach.

- **Bayesian**: Instead of having point estimates (of environment) we can incorporate uncertainty by having a prior distribution over the environment and update it as we observe data. Eventually, it's well known that the posterior distribution under minimal conditions (the support of prior distribution should contain the true environment) will concentrate around the truth. This approach is inspired by Thompson Sampling [5] in the online-learning literature.

Although, both of these approaches are valid in practice but the latter offers a more efficient way of exploration which leads to better regret bounds [4]. Indeed, there is always a trade-off between the efficacy and computational efficiency of the algorithm. In Bayesian approaches often we are facing with the issue of intractability of computing the posterior. However, methods discussed here resolve this issue by approximating the posterior distribution in variety of ways.

Mathematical Formulation: Suppose we are interacting with an environment which can be modeled as a Markov Decision Process (MDP) $\mathcal{M} = (H, \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \rho)$ where H is time horizon, $\mathcal{S}$ is a finite state space, $\mathcal{A}$ is action space, $\mathcal{P}$ is transition kernel, $\mathcal{R}$ is reward distribution, and $\rho(.)$ represents the initial distribution on the state space in the beginning of an episode. For convenience we assume state space decomposes $\mathcal{S} = \cup_{t \leq H}(t, \mathcal{S}_t)$ where t is the time-step and for every state $s \in \mathcal{S}$ we have $s = (t(s), \mathcal{S}_{t(s)})$ and ρ is supported on $(1, \mathcal{S}_1)$.[1] We define a policy $\pi : \mathcal{S} \mapsto \Delta(\mathcal{A})$ as a mapping from states to a distribution over actions. Note that we may assume the policy to be stationary since we are including time-step in our state variable. Let us further denote the observed data in episode ℓ with $O_\ell^h = (s_t^\ell, a_t^\ell, r_t^\ell)_{t \leq h}$ and history $\mathcal{H}_L = (O_\ell^H)_{\ell \leq L}$. Assuming we are taking actions via policy π we can then define the state-action value function as the following:

$$Q_{\mathcal{M}}^{\pi}(s, a) = \mathbb{E}[\sum_{t=t(s)}^{H} r_t \mid \mathcal{M}, \pi, s, a]$$

In addition, $V_{\mathcal{M}}^{\pi}(s) = \mathbb{E}[Q_{\mathcal{M}}^{\pi}(s, \pi(s))|\mathcal{M}, \pi, s]$ is the value function associated to every state.

A Reinforcement Learning algorithm $alg^L : \mathcal{H}_{L-1} \mapsto \Pi$ can be characterised as a mapping from the observed history to a policy. Indeed, we are trying to find the optimal policy through epistemic learning, however, the notion of "optimal" may change based on the setting we are working with. The first notion of "optimal" is to consider $\mathcal{M}$ to be fixed and fully known in the hindsight. Thus, we may define optimal policy $\pi^* : s \mapsto \arg\max_{a \in \mathcal{A}} Q_{\mathcal{M}}(s, a)$ where $Q_{\mathcal{M}}$ is the true value function associated to the MDP. In this setting the regret can be defined in a Frequentist manner as follows:

$$\text{Regret}(alg, \mathcal{M}, L) \coloneqq \mathbb{E}[\sum_{\ell=1}^{L} V_{\mathcal{M}}^{\pi^*}(s) - V_{\mathcal{M}}^{\pi_\ell}(s) \mid \mathcal{M}, s \sim \rho(.)]$$

Where $\pi_\ell = alg(\mathcal{H}_{\ell-1})$ is the policy generated by the algorithm.[2] Trivially, the regret incurred by an algorithm in this setting is task specific and does not provide a way for us to compare algorithms. Frequentists tend to resolve this by measuring the regret in the worst case scenario $\sup_{\mathcal{M}} \text{Regret}(alg, \mathcal{M}, L)$ among a family of MDPs. On the contrary, Bayesians take an average of regrets over some distribution on a family of MDPs. More precisely, let us assume μ is a measure over a family of MDPs. Then:

$$\text{Regret}(alg, L) \coloneqq \mathbb{E}_\mu[\text{Regret}(alg, \mathcal{M}, L)] = \int \text{Regret}(alg, \mathcal{M}, L)\ \mu(d\mathcal{M})$$

One can imagine Bayesian setting as a more flexible alternative since we can recover the Frequentist setting by setting μ to be a Dirac measure on the underlying truth. The connection is more transparent through the following lemma.[3]

Lemma 1 (Duality) *Consider a valid algorithm $alg = (alg^\ell)_{\ell \leq L}$ and an arbitrary measure μ over a family of MDPs. Then the following duality equation holds where suprema and infima are over a*

[1] This is merely required for the regret bound analysis and the methods are applicable beyond this setting.

[2] It is worth mentioning that we are implicitly assuming actions in O_ℓ are taken based on π_ℓ.

[3] All the infima and suprema are over a valid set of families of algorithms and MDPs.

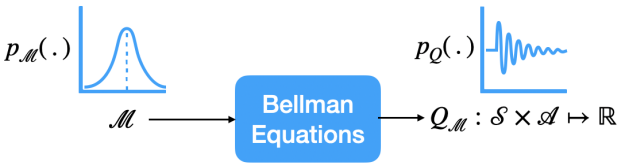
family of valid algorithms and MDPs:

$$\sup_{\mu} \inf_{alg} Regret(alg, L) = \inf_{alg} \sup_{\mathcal{M}} Regret(alg, \mathcal{M}, L) \tag{1}$$

One might now ask what is the optimal algorithm through the Bayesian lens which achieves the infimum in the left hand side of equation (1). At episode ℓ given the history $\mathcal{H}_{\ell-1}$ the optimal policy can be characterized as follows:

$$\pi^*_\ell : s \mapsto \arg\max_{a \in \mathcal{A}} \mathbb{E}[Q_{\mathcal{M}}(s,a)|\mathcal{H}_{\ell-1}]$$

Notice that computing the Bayes optimal policy is essentially challenging since it requires you to know how a prior on $\mathcal{M}$ translates into a prior on the corresponding value functions $Q_{\mathcal{M}}$. Moreover, if we had access to an oracle for computing the prior, updating the posterior would seem to be intractable. Surprisingly, it is well known in the online-learning literature that acting greedily based upon posterior mean will lead to poor exploration [5]. Instead, inspired by Thompson Sampling we explore the environment more efficiently by trying to sample a value function from the posterior and act greedily upon that.



$$\tilde{\pi}_\ell : s \mapsto \arg\max_{a \in \mathcal{A}} \tilde{Q}(s,a) \ , \ \tilde{Q} \sim p_{Q_{\mathcal{M}}}(.|\mathcal{H}_{\ell-1})$$

Connection with Online Learning: In order to benefit from classical literature results we establish a more concrete connection in the Reinforcement Learning to the classical setting by translate our setting into a Contextual Bandit Problem in the following way. Let us define the set of actions $\mathcal{A}'$ to be the set of policies Π and state variable to be the underlying MDP $\mathcal{M}$. Thus, an agent at every episode is allowed to select a policy from Π based on the context $\tilde{Q}$ provided by an oracle which is a sample from the posterior distribution $p_{Q_{\mathcal{M}}}(.|\mathcal{H}_{\ell-1})$. Moreover, a Thompson Sampling agent opts greedy action $\tilde{\pi}_\ell$ upon receiving the context. Subsequently, the agent observes $Y_\ell : s \mapsto \tilde{Q}(s, \pi_{\ell-1}(s))$ and receives a reward $R_\ell := Y_\ell(s_1) + z_\ell$ where $z_\ell \sim \mathcal{N}(0, v)$ is white noise independent from the past and $s_1 \in \mathcal{S}$ is a fixed initial state. Note that given history this reward on average is equal to the average value function associated to the initial state. More precisely, $\mathbb{E}[R_\ell|\mathcal{H}_{\ell-1}] = \mathbb{E}[Q_{\mathcal{M}}(s_1, \pi_{\ell-1}(s_1))|\mathcal{H}_{\ell-1}] = \mathbb{E}[V^{\pi_{\ell-1}}_{\mathcal{M}}(s_1)|\mathcal{H}_{\ell-1}]$ and maximum average regret can be identified as:

$$R^* := \max_{\pi \in \Pi} \mathbb{E}[R_\ell|\mathcal{M}, \pi_{\ell-1} = \pi] = \max_{\pi \in \Pi} Q_{\mathcal{M}}(s_1, \pi(s_1)) = V^{\pi^*}_{\mathcal{M}}(s_1)$$

Hence, through this correspondence one can see that the regret defined in both cases match and inherit all the previous results in the classical literature for free!

Indeed, generating samples from intractable distributions are yet difficult. This has lead to alternative ideas such as ensemble sampling, approximate posterior sampling to name but a few which we discuss here.

2 Ensemble Sampling

The main idea in this section is to use bootstrap as a proxy to construct an approximate posterior distribution. Consider a fixed history/dataset $\mathcal{H}$ divided into K sub-datasets denoted by $\mathcal{H}^{(1)}, \ldots, \mathcal{H}^{(K)}$ using a bootstrap resampling method. Although, the constructed datasets have

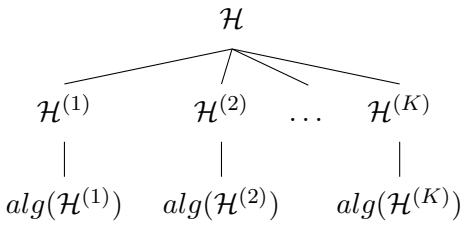
some amount of data shared among each other, however, given history $\mathcal{H}$ they are independent. By applying the algorithm to each of these datasets we may form a population of policies $(\hat{\pi}^{(i)} = alg(\mathcal{H}^{(i)}))_{i \leq K}$ which is an approximation to the true underlying distribution of $alg(\mathcal{H})$ given history $\mathcal{H}$.[4] Therefore, we may sample from the approximate posterior by simply choosing one of these policies uniformly at random.

Though, this is statistically plausible but it does not fit entirely to our setting since the data is constantly evolving and the evolution in fact depends on the agents policy. However, we may adapt to this concept so long as the algorithm at hand is comprised from an online learning procedure internally that does not require full dataset in advance. Fortunately, this is true for most of RL learning algorithms such as TD-learning. The following is a universal ensemble algorithm which is the core ingredient of methods we are trying to illustrate in this report:

Algorithm 1: Master Ensemble Algorithm

Input : **alg** : $\mathcal{H} \mapsto \Pi$, $K \in \mathbb{N}$
Initialize : initialize $\mathbf{alg}^{(i)}$ parameters $1 \leq i \leq K$

for $\ell = 1 \dots L$ **do**
  Sample : $I \sim \text{Unif}(\{1, \dots, K\})$
  Act : collect data with $\mathbf{alg}^{(I)}$ policy
  for $k = 1 \dots K$ **do**
    Buffer : store each sample w.p. p (independently)
    Parameters: update parameters by running `alg.update()`
  end
end



In [2], they follow this line of idea to produce an algorithm called **Bootstrapped-DQN** by performing a double-or-nothing bootstrap on the dataset with Deep-Q-Network agents and minimizing TD-loss. Loosely speaking, after data collection (for an episode) the parameters of the neural net will be updated through a few steps of batch stochastic gradient descent step.
One evident issue with this algorithm is that one needs to maintain an ensemble of deep neural networks which might not be scalable and efficient memory-wise. One way to circumvent this in practice is to share initial layers of the network which tries to provide a low dimensional representation of the input and only vary last few layers.

Note that [6] provides an upper bound on the Bayesian regret gap between the Ensemble approach and Thompson Sampling (true posterior sampling) when the ensemble size is of order $\Omega(|\mathcal{A}^{'}| \log(|\mathcal{A}^{'}|))$. However, the action space $\mathcal{A}^{'}$ is exponentially large in our correspondence which renders this bound useless but we conjecture that through a more customized analysis we should be able to reduce the order down. The main takeaway is that at least in the linear tabular setting (i.e. associate a parameter to every state action) we obtain sub-linear regret bounds for large enough ensemble size.

[4] One can think of this as a sensitivity analysis by perturbing the dataset and running the algorithm multiple times.

3 Approximate Posterior Sampling

In the previous section we introduced a non-parametric approach to posterior sampling where exact inference was intractable. As an alternative we may use a parametric sampling approach which approximates the posterior distribution. Consider $f_\theta : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ to be a family of functions (e.g. neural networks) indexed by θ which has a prior $\mathcal{N}(\bar{\theta}, \lambda I)$. One can imagine f_θ as a value function corresponding to some MDP $\mathcal{M}_\theta$ for every θ. We can even relax this assumption more by taking $\mathcal{M}_\theta \in \arg\min_{\mathcal{M}} ||f_\theta - Q_{\mathcal{M}}||$ as the corresponding MDP in the agnostic scenario but for the sake of simplicity we assume the more restrictive assumption. Heuristically speaking, the following should hold:

$$\mathbb{E}[r + \max_{a \in \mathcal{A}} f_\theta(s^{'}, a) - f_\theta(s, a) \mid \theta, s, a] = 0$$

where the expectation is with respect to randomness in the transition according to $\mathcal{M}_\theta$. Moreover, let us assume that the observed reward is noisy with additional white Gaussian noise. More precisely, the agent observes reward $\tilde{r}_t \coloneqq r_t + z_t$ where $z_t \overset{i.i.d}{\sim} \mathcal{N}(0, v^2)$ as opposed to observing r_t.[5] Hence, given transition $(s_t, a_t, \tilde{r}_t, s_{t+1})$, temporal difference on this transition is normally distributed. The core idea is to extend a simple computation from Bayesian Linear Regression for nonlinear regression. Let us define the modified TD-loss functions associated to Randomized Least Squares Value Iteration (RLSVI) and Randomized Prior Function (RPF) respectively as the following:

$$\mathcal{L}_{RLS}(\theta; \theta^-, \theta_0, \mathcal{H}) = \frac{1}{v^2} \sum_{(s,a,\tilde{r},s') \in \mathcal{H}} (\tilde{r} + \max_{a \in \mathcal{A}} f_{\theta^-}(s^{'}, a) - f_\theta(s, a))^2 + \frac{1}{\lambda}\|\theta - \theta_0\|_2^2$$

$$\mathcal{L}_{RPF}(\theta; \theta^-, \theta_0, \mathcal{H}) = \frac{1}{v^2} \sum_{(s,a,\tilde{r},s') \in \mathcal{H}} (\tilde{r} + \max_{a \in \mathcal{A}} (f_{\theta^-} + f_{\theta_0})(s^{'}, a) - (f_\theta + f_{\theta_0})(s, a))^2 + \frac{1}{\lambda}\|\theta\|_2^2$$

where $\theta_0 \sim \mathcal{N}(\bar{\theta}, \lambda I)$ is an independent copy of θ and is fixed through the whole training. One can simply extend ensemble sampling by utilizing these objectives instead of the TD-loss. Note that both of these objectives are equivalent by change of variable (translation) if f_θ is linear in θ which leads to the same optimization problem; Furthermore, in that case, minimizing either of these loss functions over an episode would result in a approximate posterior sample of $p_\theta(.|\mathcal{H})$ (see Lemma 3. [3]).

It is worth mentioning that Randomized Prior Function objective is fundamentally different from RLSVI in nonlinear case. This is due to an additive prior term which behaves as an intrinsic reward for exploration. Suppose the true reward is zero in most of the states and the bootstrap ensemble learned to predict zero for all states. However, the prior term abstain the agent from generalizing to predict zero since one might expect $\mathbb{E}[\max_{a \in \mathcal{A}} f_{\theta_0}(s^{'}, a)]$ to be positive for some agent in the ensemble [3].

Although, randomized prior function is practically more plausible, however, RLSVI algorithm is more suited for proving theoretical guarantees. In the following we assume we are running the modified ensemble algorithm with ensemble size $K = 1$. To our knowledge, valid theoretical bounds for larger K has not been proven yet even in the linear tabular case. [4] proves a Bayesian regret bound $\tilde{O}(H\sqrt{|\mathcal{S}||\mathcal{A}|HL})$ for RLSVI where H is time horizon and L is number of episodes. Moreover, [7] proves worst-case regret bounds $\tilde{O}(H^3|\mathcal{S}|^{\frac{3}{2}}\sqrt{|\mathcal{A}|L})$ which is not quite sharp. More

[5] One can think of this as injecting noise into the reward and then store it if we were provided with the true reward.

recently, [9] proves worst case regret bounds $\tilde{O}(d^2H^2\sqrt{T})$ over the set of MDPs with low rank dynamics where d is the dimension of the underlying feature space.

4 Simulation

In this section we provide a simple comparison study on these methods over the Deep Sea Environment [4]. This environment consists of a $N \times N$ grid where the agent starts off from the top-left state. The action space is binary which indicates a right or left action but the agent does not know action associations as well. In other words, the agent does not know which action results in going right or left at states that has not been visited yet. Moreover, moving right incurs a penalty (i.e. negative reward $\frac{-0.01}{N}$) on the agent which motivates the agent to not explore the environment. The bottom right state is a mystery state! It might contain a treasure (reward 1) or a bomb (reward -1). We implemented an Ensemble Randomized Prior Function with priors having the same neural network structure as the DQN except with different (independent) initialization. This task does not require to be modeled in a complicated way but following in [4], we modeled states as a a Boolean image (one at the current location of the agent and zero otherwise). Q-Networks consisted of only one hidden layer with size of 20 for $N = 20$ where updates of the network were handled using ADAM optimizer. Admittedly, we have found that simple SGD (without decay rate) does not converge to a valid solution and required careful step size tuning. Furthermore, buffers associated to each agent were identical, i.e. $p = 1$.

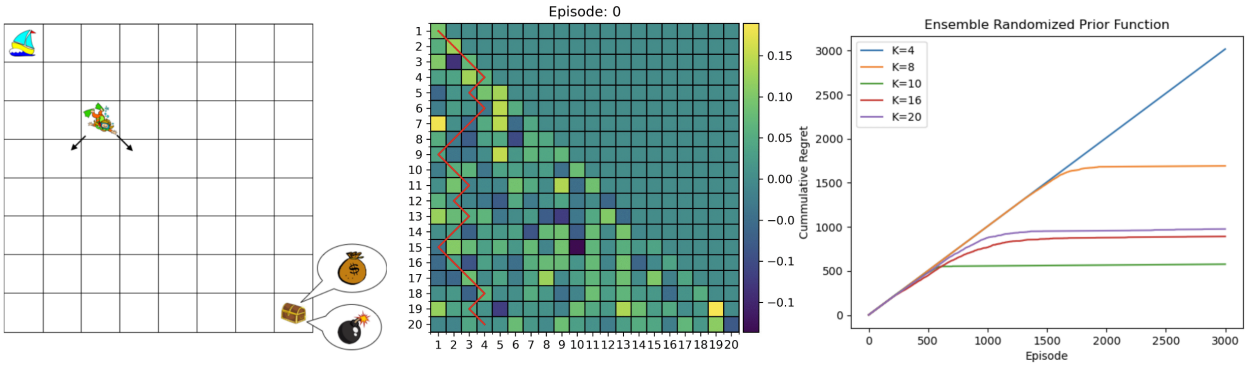


Figure 1: Left is a figure from [4] demonstrating mechanism of Deep Sea Environment. Middle is a heat map generated by aggregating value function associated to each state from the ensemble. Here the red path shows the trajectory taken by the current episode's policy. Moreover, ensemble size K here is equal to 10. Right is a sensitivity analysis on ensemble size hyper-parameter.

Finally, one interesting observation is that the effect of ensemble size was not monotonic which contradicts with the behaviour in the linear tabular case as mentioned in previous sections. The optimal regret were achieved at $K = 10$ for this specific environment. To our knowledge, it is worth mentioning that there is yet no theoretical guarantee on the regret bounds for either of Ensemble Randomized Prior or Ensemble Randomized Least Square in the linear tabular case.

For the sake of completeness and clarity of this report we also include the comparison of the previously mentioned methods on Deep Sea environment from [3]. Let us define the learning time of an algorithm to be the first episode at which average cumulative regret becomes less than 0.5. The following figure demonstrates how these algorithms scale with N where the dashed line is the exponential baseline curve:

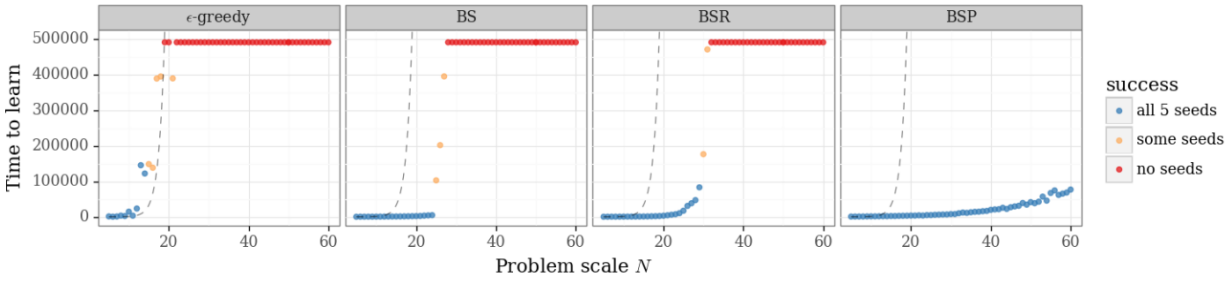


Figure 2: From left to right methods used were $\epsilon-greedy$, Bootstrap DQN, Bootstrap Randomized Least Squared, and Bootstrap Randomized Prior Function. Ensemble size were $K = 5$ and Q-networks had one hidden layer with 20 neurons. Moreover, bootstrap probability p was 0.5

5 Conclusion

In this report, we have tried to establish a more concrete connection with online-decision problems to motivate Thompson Sampling approach. We have included approximate methods to overcome the difficulties induced by intractability of posterior sampling. Methods provided, had theoretical guarantees in the tabular case even with function approximation. Finally, a comparison simulation were conducted to demonstrate the effectiveness of these algorithms. In addition, two main questions arose during reading the reference papers which was also addressed in the previous sections. Firstly that these methods seems to rely heavily on ADAM optimizer and it would be interesting to follow the dynamics of this optimization method in the linear tabular case and compute regret bounds. Secondly, extended versions of ensemble sampling (Bootstrap DQN + Randomized Prior Function) seems to be controversial in the sense that ensemble size behaves differently for linear tabular and functional approximation setting.

References

[1] Exploration Strategies in Deep Reinforcement Learning

[2] Osband, I., Blundell, C., Pritzel, A. and Van Roy, B., 2016. Deep exploration via bootstrapped DQN. In Advances in neural information processing systems (pp. 4026-4034).

[3] Osband, I., Aslanides, J. and Cassirer, A., 2018. Randomized prior functions for deep reinforcement learning. In Advances in Neural Information Processing Systems (pp. 8617-8629).

[4] Osband, I., Van Roy, B., Russo, D.J. and Wen, Z., 2019. Deep Exploration via Randomized Value Functions. Journal of Machine Learning Research, 20(124), pp.1-62.

[5] Russo, D., Van Roy, B., Kazerouni, A., Osband, I. and Wen, Z., 2017. A tutorial on thompson sampling. arXiv preprint arXiv:1707.02038.

[6] Lu, X. and Van Roy, B., 2017. Ensemble sampling. Advances in neural information processing systems, 30, pp.3258-3266.

[7] Russo, D., 2019. Worst-case regret bounds for exploration via randomized value functions. In Advances in Neural Information Processing Systems (pp. 14433-14443).

[8] Dwaracherla, V., Van Roy, B., 2020. Langevin DQN. [Preprint] Available from: https://arxiv.org/abs/2002.07282

[9] Zanette, A., Brandfonbrener, D., Brunskill, E., Pirotta, M. and Lazaric, A., 2020, June. Frequentist regret bounds for randomized least-squares value iteration. In International Conference on Artificial Intelligence and Statistics (pp. 1954-1964). PMLR.