Neural-Linear Contextual Bandits with Bounded Memory

Anonymous Author(s)
Affiliation
Address
email

Abstract

We consider the stochastic contextual multi-armed bandit problem and propose a hybrid algorithm that utilizes Thompson Sampling (TS) to learn linear actions on top of a non-linear (deep) representation that is being optimized over time. Recently it was shown that this neural-linear approach enjoys superior empirical performance compared to both deep and linear state-of-the art algorithms. This unification is appealing since the deep neural network learns separable representations and the linear approximation provides accurate uncertainty estimates. A practical challenge here is that the representation is assumed to be fixed over time when used by the contextual bandit, while the deep learning based representations change during the optimization. Previous studies addressed this problem by storing the entire data set (in memory), and generating new features (for all the data) at each iteration. Instead, we propose an algorithm that can operate under bounded memory constraints, i.e., without a direct access to all of its previous experience. Specifically, for every batch of data, we use a DNN with fixed weights and perform TS on top of its last layer. We then train the DNN on this batch, ans as a result the representation is changed. To address this change we compute the likelihood of the reward (conditioned on the observed data) given the new features, to match the likelihood of the reward given the old features. Since the likelihood of the reward is Gaussian, this is done by matching the first and second moments of the likelihood. In practice, we approximate this step by using stochastic gradient descent and by solving a semi-definite program. We demonstrate empirically that our algorithm is resistance to catastrophic forgetting, exhibiting superior performance on simulated and real-world data sets over the baseline.

1 Introduction

The use of deep learning methods to solve sequential decision making problems have achieved state-of-the-art results in a variety of challenging, high-dimensional reinforcement learning (RL) domains (Mnih et al., 2015). This success is attributed to the power of deep neural networks (DNN) to learn rich domain representations for approximating the value function or policy (Mnih et al., 2015; Zahavy et al., 2016; Zrihem et al., 2016). Similarly, for contextual bandits, using nonlinear function approximation allows the algorithm to learn better representations and enjoy superior performance (Riquelme et al., 2018; Zahavy et al., 2018). Nevertheless, performing exploration with deep networks end-to-end is still struggling; linear approximations, on the other hand, are more stable and enjoy accurate uncertainty estimates, but require substantial feature engineering to achieve good results. A natural attempt at getting the best of both worlds is to learn a (linear) control policy on top of the representation of the last layer of a DNN, reffered to as a neural-linear method. In RL, This approach was shown to refine the performance of Deep Q Networks (DQNs, Levine et al. (2017)) and improve exploration (Azizzadenesheli et al., 2018; O'Donoghue et al., 2018; Osband et al., 2018). For contextual bandits, Riquelme et al. (2018) proposed a neural-linear Thompson Sampling (TS) algorithm and demonstrated superior performance on multiple data sets.
A practical challenge for using deep representations for contextual bandits is that the features are

Submitted to 32nd Conference on Neural Information Processing Systems (NIPS 2018). Do not distribute.

assumed to be fixed over time when used by the contextual bandit, while deep representations change after every optimization step. Riquelme et al. (2018) tackled this problem by storing the entire data set in a memory buffer, and computing new features for all the data whenever they change the features. Since this approach does not scale to large data sets, they also experimented with a bounded memory buffer, but observed a significant decrease in performance due to "catastrophic forgetting", i.e., a loss of information from previous experience. In a similar line of work, Levine et al. (2017) proposed a batch DRL method that uses the experience replay buffer (with bounded memory) of the DQN to compute better weights to the last layer of the DQN. They observed that by using the weights of the DQN as prior μ_0 when performing this update, their algorithm is more resistance to catastrophic catastrophic forgetting, exhibiting better performance.

In this work, we propose a hybrid algorithm for the stochastic contextual multi-armed bandit problem that performs linear TS on top of a nonlinear (deep) representation. TS is an algorithm for online decision problems where actions are taken sequentially in a manner that must balance between exploiting the current knowledge to maximize immediate performance and exploring new information that may improve future performance. The algorithm addresses a broad range of problems in a computationally efficient manner and is therefore enjoying wide use. Our algorithm utilizes the representation power of deep neural nets, while performing accurate TS using linear Bayesian regression on top of this representation. More specifically, we address the challenge of "catastrophic forgetting" due to finite memory, by matching the moments of the likelihood of the reward conditioned on the data whenever we update the representation. This is done by using the weights of last layer of the DNN as prior to the mean, and using a semi-definite program to approximate the variance. Simulation results show that by using these priors we are able to achieve state-of-the-art results under bounded memory constraints.

2 Background

2.1 Problem setting

We consider the stochastic contextual multi-armed bandit problem with N arms. At time t a context vector $b(t) \in \mathbb{R}^d$, is revealed. These context vectors are chosen by an adversary in an adaptive manner after observing the arms played and their rewards up to time $t-1$, i.e. history $H_{t-1} = \{a(\tau), r_{a(\tau)}(\tau), b(\tau), \tau = 1, ..., t-1\}$, where $a(\tau)$ denotes the arm played at time τ. The contexts $b(t)$ are assumed to be **realizable**, i.e., the reward for arm i at time t is generated from an (unknown) distribution s.t. $\mathbb{E}\left[r_i(t)|\{b(t)\}_{i=1}^N, H_{t-1}\right] = \mathbb{E}\left[r_i(t)|b(t)\right] = b(t)^T\mu_i$ where $\mu_i \in \mathbb{R}^d$ is a fixed but unknown parameter. An algorithm for this problem needs to choose, at every time t, an arm $a(t)$ to play, with the knowledge of history H_{t-1} and current context $b(t)$. Let $a^*(t)$ denote the optimal arm at time t, i.e. $a^*(t) = \arg\max_i b(t)^T\mu_i$, and let $\Delta_i(t)$ the difference between the mean rewards of the optimal arm and of arm i at time t, i.e., $\Delta_i(t) = b(t)^T\mu_{a^*(t)} - b(t)^T\mu_{a(t)}$. The objective is to minimize the total regret $R(T) = \sum_{t=1}$ regret(t) in time T, where the regret at time t is defined as regret$(t) = \Delta_{a(t)}(t)$ and the time horizon T is finite but possibly unknown. Finally, we assume that the noise η_i, t is conditionally R-sub-Gaussian.

2.2 Thompson Sampling for linear contextual bandits

We briefly discuss the Thompson Sampling algorithm for linear contextual bandits (Algorithm 1, Agrawal & Goyal (2013)). Suppose that the likelihood of reward $r_i(t)$, given context $b(t)$ and parameter μ_i, were given by the pdf of Gaussian distribution $N(b(t)^T\mu_i, v^2)$, and let $B_i(t) = I_d + \sum_{\tau=1}^{t-1} b(\tau)b(\tau)^T, \mu_i\hat{(t)} = B(t)^{-1}\sum_{\tau=1}^{t-1} b(\tau)r_{a(\tau)}(\tau)$. For this case, it is know that using a Gaussian prior at time t for μ $(N(\hat{\mu}(t), v^2B(t)^{-1}))$, allows to compute the posterior distribution at

Algorithm 1 Thompson Sampling for Contextual bandits

Set $B = I_d$, $\hat{\mu} = 0_d$, $f = 0_d$
for $t = 1, 2, \ldots,$
  Sample $\tilde{\mu}(t)$ from distribution $N(\hat{\mu}, v^2B^{-1})$
  Play arm $a(t) := \arg\max_i b(t)^T\tilde{\mu}(t)$, and observe reward r_t
  Update $B_{a(t)} = B_{a(t)} + b(t)b(t)^T$, $f_{a(t)} = f_{a(t)} + b(t)r_t$, $\hat{\mu}_{a(t)} = B_{a(t)}^{-1}f_{a(t)}$

time $t+1$ analytically as, $Pr(\tilde{\mu}|r_i(t)) \sim Pr(r_i(t)|\tilde{\mu})Pr(\tilde{\mu}) \sim N(\mu(t+1), v^2B(t+1)^{-1})$. At each time step t, the algorithm generates a sample $\tilde{\mu}(t)$ from the posterior distribution $N(\hat{\mu}(t), v^2B(t)^{-1})$, plays the arm i that maximizes $b(t)^T\mu_i(t)$ and updates the posterior.

3 Method

3.1 TS with changing features

We present our algorithm that learns a nonlinear representation of the context using a DNN, and explores by performing linear TS using deep representations. Denote the DNN at time t by D, given context $b(t)$ we denote the activations of the last hidden layer of this neural net as $\phi(t) = \text{LastLayerActivations}(D(b(t)))$. $\phi(t)$ is typically of lower dimensions than $b(t)$ and can potentially be realizeable (even if $b(t)$ is not) since the neural net is a global function approximator. Every L iterations, the neural net is trained to minimize the mean squared error from the reward of the played arm, similar to (Riquelme et al., 2018). After this optimization step, the weights of the network change, and so does its activations, which we denote by $\psi(t) = \text{LastLayerActivations}(D(b(t)))$. For analysis purposes, we will assume that all the representations that are produced by the neural net are **realizeable**[1], i.e., $\mathbb{E}\left[r_i(t)|\phi(t)\right] = \phi(t)^T\mu_i = \psi(t)^T\beta_i = \mathbb{E}\left[r_i(t)|\psi(t)\right]$.
Next, observe that under the realizeablity assumption, the likelihood of the reward is invariant to the choice of representation , i.e. $N(\phi(t)^T\mu_i, v^2) \sim N(\psi(t)^T\beta_i, v^2)$. Thus, if we can guarantee that the estimation of this likelihood given the new features will be exactly as with the old features, then the algorithm should be resilient to the change of parametrization. Recall (see Section 2.2) that the estimation of the reward at time t, given by $\phi(t)^T\tilde{\mu}_{a(t)}(t)$ is a Gaussian random variable with mean $\phi(t)^T\hat{\mu}_{a(t)}(t)$ and standard deviation $vs_{i,t}$, where, $s_{i,t} = \sqrt{\phi(t)^T\Phi_i(t)^{-1}\phi(t)}$ is the standard deviation of estimate $b(t)^T\hat{\mu}(t)$. Thus, to approximate the likelihood it is enough to estimate its moments $\phi(t)^T\hat{\mu}_{a(t)}(t), vs_{i,t}$. The following lemma summarizes this intuition; a technical proof appears in the supplementary material.

Lemma 1. *For the stochastic contextual bandit problem with linear payoff functions, given a set of realizeable feature functions $\{\phi^i\}_{i=1}^t$, performing TS with features ϕ_i gives the same regret as performing TS with changing features $\{\phi^i\}_{i=1}^t$ while matching the moments of the likelihood whenever the features change.*

3.2 Implementation

Our algorithm (Algorithm 2), preforms TS using the last layer activations of a DNN as features, and learns new weights for the DNN. The DNN is trained by minimizing the mean squared error using stochastic gradient descent, where the labels are the observed rewards. Our algorithm is very similar to the vanilla TS (Algorithm 1, (Agrawal & Goyal, 2013)) with two main differences. First, we perform TS while updating the posterior both for μ, the mean of the estimate, and v, its variance. This is done following Bayesian linear regression equations, as suggested in Riquelme et al. (2018). Second, whenever the features are updated, we compute an approximation for the moments of the new features in the following manner.

Approximation of the variance $s_{t,i}$: we wish to find a solution Ψ such that $s_{t,i} = \phi(i)^t(\Phi)^{-1}\phi(i) = \psi(i)^t(\Psi)^{-1}\psi(i)$. We are given a set of samples of the old and new features $\{\phi(i), \psi(i)\}_{i=1}^n$ from the memory buffer (of size n) and the inverse covariance matrix Φ^{-1} (which was estimated using "old" features ϕ at previous iterations), and the goal is to estimate $\Psi^{-1} = \Phi_{0,a}^{-1}$. Using the cyclic property of the trace, this is equivalent to $\Psi^{-1}(\psi(i)\psi(i)^T) = s_{t,i}$. Next, we define X to be a vector of size m in the vector space of dxd symetric matrices, i.e., its i-th element is a matrix corresponding to $\psi(i)\psi(i)^T$. We also define S to be a vector of dimension m with scalar elements $s_{t,i}$. Ψ^{-1} can be found by solving a semidefinite program (Equation 1). Note that $\text{Trace}(X^T\Psi^{-1})$ is an inner product over the vector space of symmetric matrices, known as the Frobenius inner product. In practice, we use cvxpy (Diamond & Boyd, 2016).

$$\underset{\Psi^{-1}}{\text{minimize}} \quad ||\text{Trace}(X^T\Psi^{-1}) - S||^2 \quad \text{subject to} \quad \Psi \succeq 0. \tag{1}$$

[1] the realizability assumption is standard in the existing literature on contextual multi-armed bandits(Chu et al., 2011; Abbasi-Yadkori et al., 2011; Agrawal & Goyal, 2013)

Algorithm 2 Thompson Sampling for Deep Contextual Bandits with Bounded Memory

Set $\forall a : \Phi_{0,a} = I_d, \hat{\mu}_{0,a} = 0_d, \Phi_a = 0_{dxd}, f_a = 0_d$, ▷ $\phi(t) \in \mathbb{R}^d$
Initialize Replay Buffer E, and DNN D
Define $\phi(t) \leftarrow \text{LastLayerActivations}(D(b(t)))$
for $t = 1, 2, \ldots,$:
  $\hat{\mu}_{a(t)} = (\Phi_{0,a(t)} + \Phi_{a(t)})^{-1}(\mu_{0,a(t)} + f_{a(t)})$ ▷ Bayesian Linear Regression
  Sample $b(t)$, evaluate $\phi(t)$
  $\forall a$, sample $\tilde{\mu}_a(t)$ from the posterior distribution $N\left(\hat{\mu}_a, v^2(\Phi_{0,a} + \Phi_a)^{-1}\right)$
  Play arm $a(t) := \arg\max_i \phi(t)^T \tilde{\mu}_a(t)$, and observe reward r_t
  If E is full, remove the first tuple in E with $a = a(t)$ ▷ round robin
  Store $\{b(t), a(t), r_t\}$ in E
  Update: $\Phi_{a(t)} = \Phi_{a(t)} + \phi(t)\phi(t)^T$, $f_{a(t)} = f_{a(t)} + \phi(t)r_t$,
  if $(t \bmod L) = 0$: ▷ Change the representation and approximate the likelihood
    $\forall a$ set $\phi_a = \{\text{LastLayerActivations}(D(b(i))) : i \in E, a(i) = a\}$
    Train DNN for L steps
    for $a \in A$:
      Set $\psi_a = \{\text{LastLayerActivations}(D(b(i))) : i \in E, a(i) = a\}$
      Solve for $\Phi_{0,a}^{-1}$ using EQ 1 with $\psi_a, \phi_a, \Phi_a^{-1}$ ▷ Approximate the variance
      Set $\hat{\mu}_{0,a} \leftarrow \text{LastLayerWeights}(D)_a$ ▷ Approximate the mean
      Set $\Phi_a = 0_{dxd}, f_a = 0_d$

Approximation of the mean $\hat{\mu}_{0,a}$: Recall that the DNN is trained using sgd by minimizing the MSE, where the labels are samples of the reward. Thus, given the new last layer activations ψ, the weights of the last layer of the DNN make a good approximation to the true mean. This approach was shown empirically to make a good approximation (Levine et al., 2017), as the DNN was optimized online by observing the entire data.

4 Experiments

We evaluate our approach on several synthetic and real-world data sets; for each data set, we present the cumulative reward achieved by the different algorithms averaged over 50 runs. These algorithms include: (1) linear TS (Agrawal & Goyal (2013), Algorithm 1) using the raw context as feature, with an additional uncertainty in the variance (Riquelme et al., 2018). (2) Neural-Linear (Riquelme et al., 2018). (3) Neural-Linear with finite memory, a version of Algorithm 2 that does not use prior calculations when changing the feature representation. (4) Neural-Linear with finite memory and calculation of a prior only for the mean (following Levine et al. (2017)). (5) Neural-Linear with finite memory, with both priors computed according to Algorithm 2. Each run was performed for 5000 steps, where the finite memory methods (3-5) used a memory buffer of size 100 for each action (there are 4-8 actions depending on the data set).

We can see that on six out of eight data sets, using the prior computations, significantly improved the performance of the algorithm. Surprisingly, for the financial and stalog data sets, the bounded memory algorithm (with both priors) outperformed even the full memory algorithms. We believe that this is because real world data sets are not necessarily stationary (in particular financial data sets), and in these cases, it is better to focus on more recent experience.

	Full memory		Bounded memory, Neural-Linear		
	Linear	Neural-Linear	No Prior	μ Prior	Both Priors
Linear	6711.2 ± 166.3	6467.7 ± 168.7	6298.1 ± 171.2	6397.5 ± 175.3	**6650.1 ± 175.9**
Mushroom	11022 ± 774	10880 ± 853	7613 ± 1670	9442 ± 1351	**10923 ± 839**
Financial	4588 ± 587	4389 ± 584	4225 ± 594	4311 ± 598	**4597 ± 597**
Jester	14080 ± 2240	12819± 2135	**11114 ± 2050**	10996 ± 2013	9624 ± 2186
Statlog	4483 ± 353	4781 ± 274	4623 ± 276	4681 ± 285	**4825 ± 305**
Adult	960 ± 883	861 ± 861	689 ± 863	695 ± 880	**810 ± 898**
Covertype	3054 ± 557	2898 ± 545	2334 ± 603	2347 ± 615	**2828 ± 593**
Census	1924 ± 794	2136 ± 723	**1895 ± 737**	1873 ± 757	1853 ± 801

Table 1: Experimental results

References

Abbasi-Yadkori, Yasin, Pal, David, and Szepesvari, Csaba. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems*, pp. 2312–2320, 2011.

Agrawal, Shipra and Goyal, Navin. Thompson sampling for contextual bandits with linear payoffs. In *International Conference on Machine Learning*, pp. 127–135, 2013.

Azizzadenesheli, Kamyar, Brunskill, Emma, and Anandkumar, Animashree. Efficient exploration through bayesian deep q-networks. *arXiv preprint arXiv:1802.04412*, 2018.

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

Diamond, Steven and Boyd, Stephen. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

Levine, Nir, Zahavy, Tom, Mankowitz, Daniel J, Tamar, Aviv, and Mannor, Shie. Shallow updates for deep reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 3135–3145, 2017.

Mnih, Volodymyr, Kavukcuoglu, Koray, Silver, David, Rusu, Andrei A, Veness, Joel, Bellemare, Marc G, Graves, Alex, Riedmiller, Martin, Fidjeland, Andreas K, Ostrovski, Georg, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

O'Donoghue, Brendan, Osband, Ian, Munos, Remi, and Mnih, Volodymyr. The uncertainty bellman equation and exploration. *International Conference on Machine Learning*, 2018.

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

Riquelme, Carlos, Tucker, George, and Snoek, Jasper. Deep bayesian bandits showdown. In *International Conference on Learning Representations*, 2018.

Zahavy, Tom, Ben-Zrihem, Nir, and Mannor, Shie. Graying the black box: Understanding dqns. In *International Conference on Machine Learning*, pp. 1899–1908, 2016.

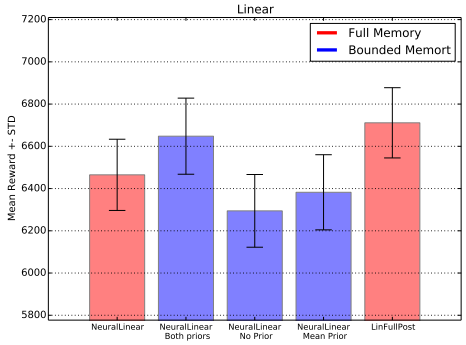
Zahavy, Tom, Haroush, Matan, Merlis, Nadav, Mankowitz, Daniel J, and Mannor, Shie. Learn what not to learn: Action elimination with deep reinforcement learning. *Advances in Neural Information Processing Systems*, 2018.

Zrihem, Nir Ben, Zahavy, Tom, and Mannor, Shie. Visualizing dynamics: from t-sne to semi-mdps. *arXiv preprint arXiv:1606.07112*, 2016.

5 Supplementary material: proof

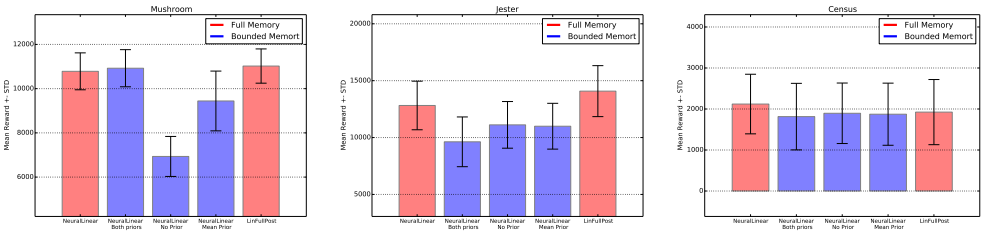
Proof. We follow the proof of (Agrawal & Goyal, 2013). When the features change, we will denote the features before the change by $\phi(t)$ and after it by $\psi(t)$. By definition of $\tilde{\mu}(t)$, the marginal distribution of the estimate $\theta_i(t) = \phi(t)^T \tilde{\mu}_i(t)$ is Gaussian with mean $\phi(t)^T \hat{\mu}(t)$ and standard deviation $\nu s_{t,i} = \nu \sqrt{\phi_i(t) \Phi(t)^{-} 1 \phi_i(t)}$. Since we match the moments of this estimate, then estimate given the new features $\psi(t)$ have exactly the same moments and distribution. Since the distribution of the estimate did not change, its concentration and anti-concentration bounds do not change which implies that lemma's 1&2 from Agrawal & Goyal (2013) still holds. Similarly, since the variance of the estimate did not change, saturated (unsaturated) arms remain saturated (unsaturated) by definition, thus, lemma 3 holds as well. We define the filtration $\mathcal{F}_{t-1}$, as the union ofd the history until time $t-1$, and the context at time t, i.e., $\mathcal{F}_{t-1} = \{\mathcal{H}_{t-1}, \phi^i(t)\}$. The history at time t is defined to be $\mathcal{H}_t = \{a(\tau), r_{a(\tau)}, \phi^{i(\tau)}(\tau), i = 1, ..., N, \tau = 1, .., t\}$, i.e., the sequence of actions rewards and (time dependent) features. As a consequence, lemma 4 and the proof of the theorem holds as well. □

6 Supplementary material: additional figures



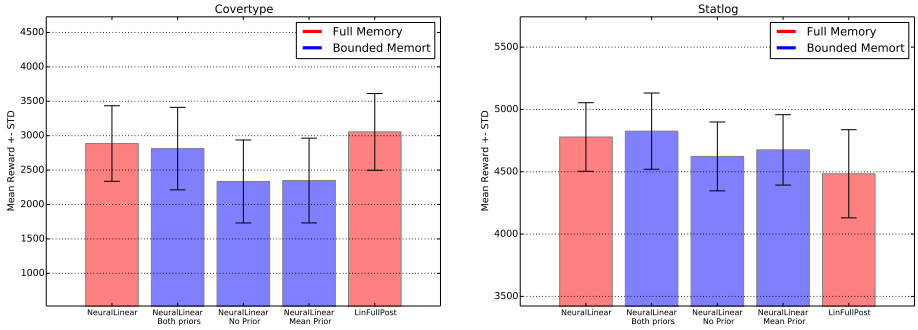
(a) Linear

Figure 1: Results on synthetic, linear data set.



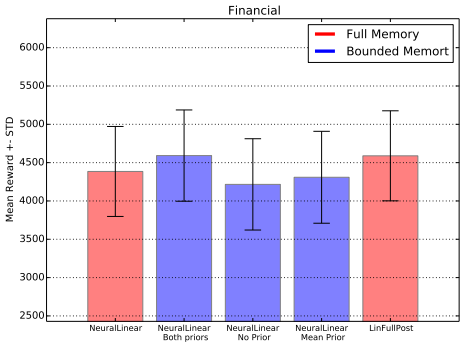
(a) Mushroom (b) Jester (c) Census

Figure 2: Linear data sets.



(a) Covertype (b) Statlog

Figure 3: Nonlinear data sets.



(a) Financial

Figure 4: Non stationary data set