

Dataset Characteristics and Their Impact on Offline Policy Learning of Contextual Multi-Armed Bandits

Piotr Januszewski¹^a, Dominik Grzegorzek¹^b and Paweł Czarnul¹^c

¹*Department of Computer Architecture, Gdańsk University of Technology, Gdańsk, Poland*
{piotr.januszewski, pawel.czarnul}@pg.edu.pl, grzegor.dominik@gmail.com

Keywords: Contextual Multi-Armed Bandits, Offline Policy Learning, Dataset Quality

Abstract: The Contextual Multi-Armed Bandits (CMAB) framework is pivotal for learning to make decisions. However, due to challenges in deploying online algorithms, there is a shift towards offline policy learning, which relies on pre-existing datasets. This study examines the relationship between the quality of these datasets and the performance of offline policy learning algorithms, specifically, Neural Greedy and Neural LCB. Our results demonstrate that Neural LCB can learn from various datasets, while Neural Greedy necessitates extensive coverage of the action-space for effective learning. Moreover, the way data is collected significantly affects offline methods' efficiency. This underscores the critical role of dataset quality in offline policy learning.

1 INTRODUCTION

The Contextual Multi-Armed Bandits (CMAB) framework (Lattimore and Szepesvári, 2020) enables a myriad of services such as recommendation systems (Zhao et al., 2013), personalized healthcare (Komorowski et al., 2018), online advertisement (Li et al., 2010), and resource allocation (Badanidiyuru et al., 2013) to actively learn and adapt from interactions with their respective environments.

However, the challenge of continuously deploying online learning algorithms due to costs, privacy concerns, or constraints on exploration has brought offline policy learning to the forefront (Levine et al., 2020). Offline policy learning, which learns policies from previously collected datasets without interacting with the environment, is beneficial in settings where online interactions are costly, risky, or unethical.


One major determinant of offline policy learning performance is the quality and nature of the offline dataset (Fujimoto et al., 2019). This component is often overlooked, with the majority of studies emphasizing algorithmic improvements. Hence, there exists a research gap in fully understanding the impact of the offline dataset on the performance of the offline policy learning algorithms.


This paper seeks to shed light on the relationship between offline datasets and the performance of offline policy learning algorithms in the contextual


bandits setting. In our empirical investigation, we gather datasets that meet various conditions. For the uniform data coverage assumption (Brandfonbrener et al., 2021), actions need to be chosen uniformly at random. In contrast, the single-policy concentration condition (Nguyen-Tang et al., 2022) necessitates the behavior policy to only cover the target optimal policy. We also explore datasets that extrapolate between these two conditions or that satisfy neither, particularly those that predominantly issue suboptimal actions.

Our work considers two offline methods, Neural Greedy, referred to as the Direct Method in (Dudik et al., 2011), and Neural LCB (Nguyen-Tang et al., 2022). We chose Neural LCB due to its state-of-the-art performance in offline policy learning, offering cutting-edge insights into how different dataset characteristics impact algorithm efficiency. On the other hand, Neural Greedy is known for its simplicity and ease of implementation, and represents a method that practitioners are likely to adopt first (Dutta et al., 2019).

We present a comprehensive empirical investigation that elucidates the relationship between offline dataset characteristics and the performance of offline policy learning algorithms. This analysis specifically contrasts two distinct methods: Neural Greedy and Neural LCB, providing a nuanced understanding of how different dataset conditions influence their efficiency. Our findings offer practical guidance for practitioners in preparing offline datasets and selecting appropriate offline methods, thus contributing to improved performance in real-world applications.

^a <https://orcid.org/0000-0003-3817-3479>

^b <https://orcid.org/0009-0006-0310-7104>

^c <https://orcid.org/0000-0002-4918-9196>

2 PRELIMINARIES

We consider Contextual Multi-Armed Bandits (CMAB) framework (Lattimore and Szepesvári, 2020). Let the context space \mathcal{X} be infinite and the action space \mathcal{A} be finite with $|\mathcal{A}| = K < \infty$. Additionally, let a reward vector $r \in [r_{\min}, r_{\max}]^K$ and a context $x \in \mathcal{X}$ be drawn from a joint distribution ρ . Let \mathcal{H} represent the history of past interactions up to round $t - 1$, where a history $h_{1:t-1}$ is a sequence $h_{1:t-1} = \{(x_1, a_1, r_{a_1}), \dots, (x_{t-1}, a_{t-1}, r_{a_{t-1}})\}$.

We now define a policy $\pi : \mathcal{X} \times \mathcal{H} \rightarrow P(\mathcal{A})$ that maps the current context and the history up to that round to a distribution over actions. At each round t , a contextual bandit observes a context x_t , decides on an action $a_t \sim \pi(x_t, h_{1:t-1})$, and receives an action reward r_{a_t} which is the component of the vector r_t corresponding to the action a_t — this is termed a *bandit feedback*.

For every policy π , the regret R_T after T rounds is defined as:

$$R_T = \sum_{t=1}^T \left[\max_{a \in \mathcal{A}} r_t - r_{a_t} \right]$$

where $x_t, r_t \sim \rho$ and $a_t \sim \pi(x_t, h_{1:t-1})$ at each round t .

The suboptimality of a policy π is defined as its average regret after T rounds:

$$\text{SubOpt}(\pi) = \frac{R_T}{T}$$

In the offline setting, we define a finite dataset consisting of CMAB rounds played with a behavior policy β :

$$\mathcal{S}_\beta = \{(x_i, a_i, r_{a_i})\}_{i=1}^N$$

The goal of an offline CMAB method is to learn an optimal policy $\pi^* = \min_{\pi} \text{SubOpt}(\pi)$, that minimizes the suboptimality, using only the bandit feedback in the dataset \mathcal{S}_β .

3 METHODS

3.1 Behavior Policies

In this section, we briefly describe the four online learning algorithms used in our research to collect datasets: Neural Greedy, Bayes by Backprop, LinUCB, and NeuralUCB.

The Neural Greedy method was also used as one of the baselines in (Blundell et al., 2015). The Neural Greedy method learns from observations at each round to estimate the reward vectors r with a reward function $f(x; \theta)$. The reward function $f(x; \theta)$ is a neural network with parameters θ . It is then used in place

of the actual reward to evaluate the policy value and to maximize it by selecting the highest valued action in the policy $\pi(x) = \arg\max_a f_a(x; \theta)$, where $f_a(x; \theta)$ is the component corresponding to action a of the output from the neural network. This approach requires an accurate model of rewards, which can be a restrictive assumption in some cases. However, when the model of rewards is accurate, it can provide a reliable estimate of the policy value.

The Bayes by Backprop is an algorithm proposed in (Blundell et al., 2015) that regularizes the neural network model weights by minimizing a compression cost. It introduces uncertainty in the weights of the neural network, which can improve generalization in non-linear regression problems. The uncertainty is modeled by a variational posterior $q_\theta(w)$ over the weights w of the network, where θ is the variational parameters learned by the algorithm. The compression cost is given by the KL divergence between the variational posterior and a prior $p(w)$, and is minimized by adjusting the variational parameters. In areas where the network is less confident in its predictions (high uncertainty), Bayes by Backprop explores more, gathering more data to reduce the uncertainty. In areas where the network is more confident in its predictions (low uncertainty), Bayes by Backprop exploits its knowledge, making decisions that maximize the expected reward. This approach can help to focus more on approximating the reward in areas that are important for the policy (high uncertainty), leading to more accurate estimates of the policy value and better decision-making overall. This can be particularly beneficial in addressing the limitations of Neural Greedy, where the reward estimate might focus on approximating the reward in areas that are irrelevant to the policy (low uncertainty). The policy used by Bayes by Backprop is given by $\pi(x) = \arg\max_a f_a(x; w)$, where $f_a(x; w)$ is the component corresponding to action a of the output from the neural network with weights $w \sim q_\theta$, sampled from the variational posterior, and input context x . Effectively, this policy selects actions that maximize the expected output of the network under the variational posterior over the weights.

The LinUCB method is described in (Li et al., 2010). The authors model personalized recommendation of news articles as a contextual bandit problem. The LinUCB algorithm assumes that the expected payoff of an arm is linear in its d -dimensional feature vector with some unknown coefficient vector. The algorithm uses ridge regression to estimate the coefficients θ and computes a confidence interval for the expected payoff of each arm. The arm with the highest upper confidence bound (UCB) is selected in

each round. The policy used by LinUCB is given by $\pi(x) = \operatorname{argmax}_a x^T \theta_a + \alpha \sqrt{x^T A_a^{-1} x}$, where θ_a is the ridge regression estimate of the coefficients for action a , A_a is the design matrix for action a , and α is a hyperparameter controlling the exploration-exploitation trade-off.

The NeuralUCB method, introduced in (Zhou et al., 2020), extends the LinUCB approach by using a neural network $f(x; \theta)$ to model the reward function, allowing it to handle more complex, non-linear relationships between the context and the expected reward. This is a significant departure from the LinUCB method, which assumes a linear relationship. The policy used by NeuralUCB is given by $\pi(x) = \operatorname{argmax}_a f_a(x; \theta) + \text{UCB}(x, a, \theta)$, where $\text{UCB}(x, a, \theta)$ is an exploration bonus described in the paper.

3.2 Offline Methods

In this section, we briefly describe the two offline policy learning algorithms trained on the collected datasets: Neural Greedy and NeuralCB.

The offline variant of the Neural Greedy method is used in the context of Offline Contextual Multi-Armed Bandits (CMAB). Unlike its online counterpart, the offline Neural Greedy operates under the constraint that it cannot interact with the environment or collect more data. It aims to learn an optimal policy solely from a fixed offline dataset of bandit feedback, denoted as S_β . The method uses a neural network to estimate the reward vectors r from the observations in the dataset S_β . The policy $\pi(x)$ is then defined as the action that maximizes the estimated reward, i.e., $\pi(x) = \operatorname{argmax}_a f_a(x; \theta)$, where $f_a(x; \theta)$ is the component corresponding to action a of the output from the neural network.

The NeuralCB method is a novel approach to Offline Contextual Multi-Armed Bandits (CMAB) introduced by (Zhou et al., 2020). It uses a neural network to model any bounded reward function without assuming any functional form. The key feature of NeuralCB is its pessimistic formulation, which constructs a lower confidence bound of the reward functions for decision-making. The lower confidence bound is computed for each context and action based on the current network parameter, and the network is updated by minimizing a regularized squared loss function using stochastic gradient descent (SGD). The policy used by NeuralCB is given by $\pi(x) = \operatorname{argmax}_a f_a(x; \theta) - \text{LCB}(x, a, \theta)$, where $\text{LCB}(x, a, \theta)$ is the lower confidence bound for action a at context x with network parameters θ .

The offline policy learning faces issues with the distributional shift problem (Levine et al., 2020), par-

ticularly due to the behavior policy mismatch. This problem arises when the policy used to collect the dataset (the behavior policy) is different from the learned policy by the offline method, leading to situations in which the learned policy exploits actions about which it has little to no data. Neural Greedy addresses this problem by making assumptions about *the uniform data coverage* (Brandfonbrener et al., 2021; Xie et al., 2023). Specifically, Neural Greedy assumes that the behavior policy has already explored the entire action space to a sufficient degree. However, this assumption may not always hold in practice. To address this limitation, NeuralCB utilizes a pessimism principle that constructs lower confidence bounds for the reward functions for conservative decision-making (Rashidinejad et al., 2022). By doing so, NeuralCB reduces the requirement for uniform data coverage to *the single-policy concentration condition*, which ensures that the behavior policy only needs to cover the target optimal policy, rather than the entire action space.

In our approach, we adopt the batch mode training inspired by NeuralCB. We replay the dataset S_β in the order it was collected. During each replay of a round t , the reward model is updated using 100 steps of Stochastic Gradient Descent (SGD) on a random batch of size 50, sampled from the subset $S_\beta^{(1:t)}$ of the dataset accumulated from round 1 to t .

3.3 Problems

The algorithms are evaluated on real-world problems obtained from the UCI Machine Learning Repository (Markelle et al.,), including Mushroom, Shuttle, Adult, and MNIST. These problems are diverse in terms of size, dominant actions, and stochastic versus deterministic rewards. Details on each problem can be found in Table 1.

The Mushroom problem contains equal numbers of edible and poisonous mushroom examples. Each mushroom is represented by a set of attributes. The learner's task is to choose from two actions: to eat a given mushroom or not. If the mushroom is edible and the learner chooses to eat it, they receive a reward of +5. On the other hand, if the mushroom is poisonous and the learner chooses to eat it, there is a probability of 0.5 that they receive a reward of +5 and an equal probability that they receive a penalty of -35 instead. If the learner chooses not to eat the mushroom, they always receive a reward of 0.

All the other problems are K -class classification problems:

1. The Adult problem contains personal information from the US Census Bureau database including

Table 1: Problems characteristics

	size	context dimension	number of actions	dominant action	stochastic rewards
Adult	45,222	94	14	>80% of data belongs to half of actions	no
MNIST	70,000	784	10	balanced	no
Mushroom	8,124	22	2	balanced	yes
Shuttle	43,500	9	7	~80% of data belong to one action	no

occupations that we take for actions.

2. The MNIST problem contains images of various handwritten digits from 0 to 9.
3. The Shuttle problem contains data about a space shuttle flight where the goal is to predict the state of the radiator subsystem of the shuttle.

We convert these problems into K -armed contextual bandit problems based on the methodology proposed by (Nguyen-Tang et al., 2022). Specifically, the learner receives a reward of 1 if it selects the action y , where y is the correct label and 0 otherwise.

3.4 Datasets

We collect datasets that satisfy the uniform data coverage assumption, the single-policy concentration condition, extrapolate between the two, or satisfy neither condition. To do this we run the following behavior policies to collect the datasets:

1. Uniform and stationary policies that fulfill the uniform data coverage assumption
 - full exploration which issues uniformly random actions (we later refer to it as uniformly random data).
2. Non-uniform, but stationary policies that fulfill the single-policy concentration condition
 - full exploitation which issues only optimal actions, meaning minimizing the suboptimality.
3. Non-uniform, but still stationary policies that extrapolate between the two conditions
 - ϵ -greedy which issues uniformly random actions with probability ϵ and optimal actions otherwise.
4. Non-uniform and not-stationary policies that might not satisfy any of the two conditions are described in detail in Section 3.1
 - Neural Greedy,
 - LinUCB,
 - NeuralUCB,
 - Bayes by Backprop.

It is crucial to note that the last category includes learning algorithms that: 1) will gradually make less and less diverse decisions and 2) may or may not converge to the optimal solution. That is why they might not satisfy any of the aforementioned conditions.

We run each behavior policy 10 times, with different seeds, for 15,000 rounds in each problem and record observed contexts, issued actions, and received action rewards which establish our offline datasets.

4 RESULTS & DISCUSSION

Based on the experimental results, it was observed that the performance of offline contextual bandits algorithms is heavily dependent on the nature of datasets used for training and the specific characteristics of the problem being addressed. In this section, we explore this observation from different angles.

4.1 Offline training on datasets with varying degrees of exploration

Our study investigates whether the NeuralCB and Neural Greedy algorithms learn better from datasets collected with the ϵ -greedy behavior policy, which mixes the optimal actions with the uniformly random actions, or the full exploration behavior policy, which always picks the actions uniformly at random. Table 2 shows that the NeuralCB algorithm performs better when trained on datasets coming from behavior policies that are more focused on optimal actions, with one exception. In Figure 5, the NeuralCB algorithm performed better when trained on uniformly random data in the Mushroom problem. This exception might be related to the method's failure to learn from the optimal actions in this particular problem, as described in Section 4.2. Figure 7 reveals that NeuralCB can fail to achieve stable convergence with the dataset coming from the full exploration behavior policy, which is the case in the Shuttle problem. Table 2 shows that the Neural Greedy offline method performed better when trained on uniformly random data. This was expected, as Neural Greedy requires uniform data coverage for efficient learning under dis-

Table 2: Which random behavior policy collects datasets on which the offline method achieves better performance (AOC)?

	Adult	MNIST	Mushroom	Shuttle
NeuraLCB	e-greedy	e-greedy	u. random	e-greedy
Neural Greedy	u. random	u. random	u. random	u. random

Table 3: Which offline method achieves better performance (AOC) for each dataset?

	Adult	MNIST	Mushroom	Shuttle
e-greedy	NeuraLCB	NeuraLCB	NeuraLCB	NeuraLCB
u. random	Neural Greedy	Neural Greedy	Neural Greedy	Neural Greedy
greedy	NeuraLCB	NeuraLCB	both cannot learn	NeuraLCB
Bayes by Backprop	NeuraLCB	Neural Greedy	NeuraLCB	NeuraLCB
LinUCB	NeuraLCB	Neural Greedy	NeuraLCB / both cannot learn	NeuraLCB
Neural Greedy	NeuraLCB / draw	Neural Greedy	NeuraLCB	NeuraLCB
NeuralUCB	NeuraLCB	NeuraLCB	NeuraLCB	NeuraLCB

tributional shift (Nguyen-Tang et al., 2022). In Figure 8, in the Shuttle problem, the dataset coming from the uniformly random behavior policy is the only one that results in the Neural Greedy convergence.

4.2 Offline training on datasets with only optimal actions

We investigate whether the offline methods can learn from datasets collected with the full exploitation behavior policy, which always picks the optimal actions. Based on the results obtained, it has been observed that the Neural Greedy algorithm is unable to learn from datasets consisting solely of optimal actions. In contrast, the NeuraLCB algorithm has been found to be capable of learning on such datasets. Nevertheless, its performance still benefits from some level of exploration as can be seen in Figures 1 and 3. The Mushroom problem is one exception in which NeuraLCB cannot learn from the dataset collected by the full exploitation behavior policy. This is due to the fact that this dataset does not include the result of eating a poisonous mushroom, leading offline method to learn a suboptimal policy which thinks that eating mushrooms has a higher reward than not eating them, regardless of their type. Figure 5 shows that even a small level of exploration, such as 10% in the ϵ -greedy collected dataset, can mitigate this problem. These findings suggest that offline training of contextual bandits from optimal actions can be challenging, and at least some level of exploration may be necessary for effective training.

4.3 Offline training on datasets collected by learning behavior policies

In this experiment, we look at the offline methods' performance when trained on datasets collected by

behavioral policies that start from the random initialization and are actively learning to issue better actions.

As Figures 1 and 2 show, in the Adult problem, there is not much difference in performance across Bayes by Backprop, LinUCB, Neural Greedy, or NeuralUCB behavior policies used for datasets collection. Neither performance of the offline methods trained on these datasets differ.

In the MNIST problem, Figures 3 and 4 nicely show the situation where, although behavior policies' performance significantly differs, the NeuraLCB and Neural Greedy offline methods achieve similar performance across collected datasets – in most cases exceeding the performance of the corresponding behavior policies. This suggests that the offline methods can pick the optimal solution from a variety of datasets. However, the slower the behavior policies learn the slower the NeuraLCB algorithm learns on their collected datasets.

For the Mushroom problem, in Figures 5 and 6, all the behavior policies reach similar, near-optimal performance, with the LinUCB reaching it much quicker. The NeuraLCB algorithm quickly converges on the optimal policy across the datasets besides the one collected by the LinUCB in which it struggles to learn. This is most probably because the LinUCB behavior policy quickly starts to issue mostly optimal actions which causes difficulties described in Section 4.2. For the Neural Greedy algorithm the situation is similar, but it does not achieve the optimal performance, it maintains little bias. It is worth noting that Neural Greedy can learn the optimal policy from the uniformly random data. This suggests that the lack of ability to directly model the uncertainty about the less common actions, in this case, the eating of inedible mushrooms in datasets collected by mostly optimal behavior policies, contributes to the worse per-

formance of Neural Greedy.

Figures 5 and 6 present the results in the Shuttle problem. Behavior policies gain similar final performance but at different rates except for the LinUCB which quickly converges on the worse solution than other learning behavior policies. Similarly, as in the MNIST problem, the NeuraLCB algorithm learns slower if the behavior policy learns slowly. However, it reaches the same final performance across datasets except for the dataset collected by the LinUCB from which NeuraLCB is missing optimal actions in the dataset. Generally, the faster behavior policies converge on some solution, either optimal (see the Neural Greedy behavior policy) or suboptimal (see the LinUCB behavior policy) the more problems the Neural Greedy algorithm has to learn from such collected datasets. This confirms that the Neural Greedy algorithm requires a substantial degree of exploration in the behavior policies. This conclusion is further reinforced by the observation, that Neural Greedy recovers around the middle of the training on the dataset collected by the Bayes by Backprop behavior policy, just when the Bayes by Backprop again started to pick more explorative actions. This exploration increase can be observed as the slight Bayes by Backprop performance deterioration.

4.4 Can NeuraLCB learn where Neural Greedy fails and vice versa?

At the high level, the aim of these experiments was to investigate if NeuraLCB can learn from datasets where Neural Greedy fails and vice versa. The results in Figures 7 and 8 show that, in the Shuttle problem, NeuraLCB learns well from all datasets, while Neural Greedy can only learn from the uniformly random data. Across all problems but the Mushroom, NeuraLCB can learn from the datasets consisting solely of optimal actions, whereas Neural Greedy fails to do so. Interestingly, the study found that NeuraLCB was not always better than Neural Greedy, see Table 3. Particularly, when trained on the uniformly random data which is the most visible in the Adult problem, in Figures 1 and 2, and in the MNIST problem, in Figures 3 and 4. However, we argue that it is unreasonable to expect the deployed policies to be uniformly random in their decisions in real-world scenarios.

5 RELATED WORK

The existing literature provides valuable insights into offline policy learning, contextual bandits, and behavior policies. However, the specific exploration of how

different behavior policies influence the performance of offline policy learning algorithms in the contextual bandits setting appears to be a novel and underexplored aspect.

5.1 Offline Policy Learning

Offline policy learning has been a subject of interest in various domains. Recent works have proposed frameworks for offline reinforcement learning (Fujimoto et al., 2019), with advances like conservative Q-learning (Kumar et al., 2020), general methods for data reuse (Xiao and Wang, 2021), and methods to bridge the sim-to-real gap (Rashidinejad et al., 2022). Uncertainty-aware approaches have also gained traction (Hu et al., 2023), with methodologies like diffusion models being proposed (Wang et al., 2023). Moreover, information-theoretic considerations in offline policy learning have been explored in (Chen and Jiang, 2019). However, these works primarily focus on general frameworks and methodologies rather than the nuanced influence of different behavior policies on offline policy learning.

5.2 Contextual Bandits

Contextual bandits have been applied to diverse areas such as interactive recommendation (Li et al., 2010) and energy optimization (Vannella et al., 2023). The domain has seen studies that investigated offline policy optimization (Nguyen-Tang et al., 2022) and those that delve into the relationship of offline optimization with overparametrized models (Brandfonbrener et al., 2021). Furthermore, Joachims et al. have studied the use of a counterfactual risk minimization approach for training deep networks with logged bandit feedback (Joachims et al., 2018). While these works consider aspects of offline datasets, they do not delve into the specific impact of various behavior policies on offline policy learning performance.

5.3 Off-Policy Evaluation

There is significant research related to off-policy evaluation on offline datasets, touching on aspects like doubly robust policy evaluation (Dudik et al., 2011), distributionally robust policy gradients (Yang et al., 2023), and methodologies like variance-minimizing augmentation logging (Tucker and Joachims, 2023). The literature has also included works on neural contextual bandits with UCB-based exploration (Zhou et al., 2020) and PAC-Bayesian approaches (Sakhi et al., 2022). However, a comprehensive examination of how different behavior policies directly influence

the efficiency and effectiveness of offline policy learning algorithms remains an underexplored area.

6 CONCLUSIONS

Our study demonstrates the critical impact of dataset characteristics on offline policy learning of Contextual Multi-Armed Bandits, offering key insights for their practical application. The Neural Greedy algorithm requires datasets with a substantial degree of exploration for effective learning. In practical scenarios, however, it is unreasonable to anticipate the deployed policies to consistently make highly exploratory decisions e.g. based on uniformly random criteria. We advise employing the NeuraLCB method, as it can learn effectively from datasets collected by behavior policies that leverage problem-specific knowledge. It has the better performance, the more optimal the actions in the dataset. Nonetheless, we show NeuraLCB still benefits from some exploratory actions. We recommend ensuring that each action gets chosen multiple times in the datasets.

Future work shall tune the proportion of exploratory actions to the optimal ones for the best performance. Experiments with learning behavioral policies in Section 4.3 could also be extended by dropping early, uninformed decisions and checking how this influences the offline methods performance.

Our investigation adds a new dimension to the body of knowledge concerning offline policy learning. While algorithms undoubtedly form the learning engine, our research underscores the importance of fuel quality — the offline dataset — for the journey toward efficient offline policy learning and decision-making. We hope to contribute to the ongoing dialogue on improving the implementation of offline policy learning in real-world scenarios.

REFERENCES

- Badanidiyuru, A., Kleinberg, R., and Slivkins, A. (2013). Bandits with Knapsacks. IEEE Computer Society.
- Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. (2015). Weight Uncertainty in Neural Networks.
- Brandfonbrener, D., Whitney, W. F., Ranganath, R., and Bruna, J. (2021). Offline Contextual Bandits with Overparameterized Models.
- Chen, J. and Jiang, N. (2019). Information-Theoretic Considerations in Batch Reinforcement Learning. PMLR.
- Dudik, M., Langford, J., and Li, L. (2011). Doubly Robust Policy Evaluation and Learning.
- Dutta, P., Cheuk, M. K., Kim, J. S., and Mascaro, M. (2019). Automl for contextual bandits. *CoRR*, abs/1909.03212.
- Fujimoto, S., Meger, D., and Precup, D. (2019). Off-Policy Deep Reinforcement Learning without Exploration. PMLR.
- Hu, B., Xiao, Y., Zhang, S., and Liu, B. (2023). A Data-Driven Solution for Energy Management Strategy of Hybrid Electric Vehicles Based on Uncertainty-Aware Model-Based Offline Reinforcement Learning.
- Joachims, T., Swaminathan, A., and de Rijke, M. (2018). Deep learning with logged bandit feedback. OpenReview.net.
- Komorowski, M., Celi, L. A., Badawi, O., Gordon, A. C., and Faisal, A. A. (2018). The Artificial Intelligence Clinician learns optimal treatment strategies for sepsis in intensive care.
- Kumar, A., Zhou, A., Tucker, G., and Levine, S. (2020). Conservative Q-Learning for Offline Reinforcement Learning. Curran Associates, Inc.
- Lattimore, T. and Szepesvári, C. (2020). *Bandit Algorithms*. Cambridge University Press.
- Levine, S., Kumar, A., Tucker, G., and Fu, J. (2020). Offline reinforcement learning: Tutorial, review, and perspectives on open problems.
- Li, L., Chu, W., Langford, J., and Schapire, R. E. (2010). A contextual-bandit approach to personalized news article recommendation.
- Markelle, K., Longjohn, R., and Nottingham, K. The uci machine learning repository.
- Nguyen-Tang, T., Gupta, S., Nguyen, A. T., and Venkatesh, S. (2022). Offline neural contextual bandits: Pessimism, optimization and generalization.
- Rashidinejad, P., Zhu, B., Ma, C., Jiao, J., and Russell, S. (2022). Bridging offline reinforcement learning and imitation learning: A tale of pessimism.
- Sakhi, O., Chopin, N., and Alquier, P. (2022). Pac-bayesian offline contextual bandits with guarantees.
- Tucker, A. D. and Joachims, T. (2023). Variance-minimizing augmentation logging for counterfactual evaluation in contextual bandits.
- Vannella, F., Jeong, J., and Proutière, A. (2023). Off-policy learning in contextual bandits for remote electrical tilt optimization.
- Wang, Z., Hunt, J. J., and Zhou, M. (2023). Diffusion policies as an expressive policy class for offline reinforcement learning.
- Xiao, T. and Wang, D. (2021). A general offline reinforcement learning framework for interactive recommendation.
- Xie, T., Foster, D. J., Bai, Y., Jiang, N., and Kakade, S. M. (2023). The role of coverage in online reinforcement learning.
- Yang, Z., Guo, Y., Xu, P., Liu, A., and Anandkumar, A. (2023). Distributionally robust policy gradient for offline contextual bandits.
- Zhao, X., Zhang, W., and Wang, J. (2013). Interactive collaborative filtering.
- Zhou, D., Li, L., and Gu, Q. (2020). Neural contextual bandits with ucb-based exploration.

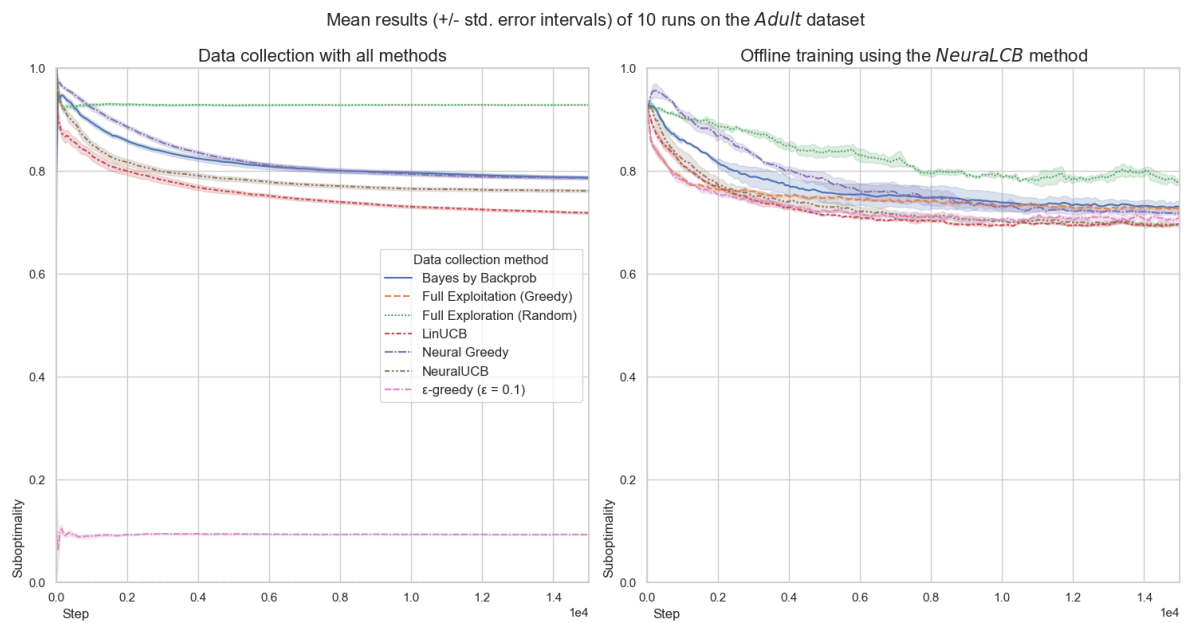


Figure 1: The *NeuralLCB* offline method results in the *Adult* problem.

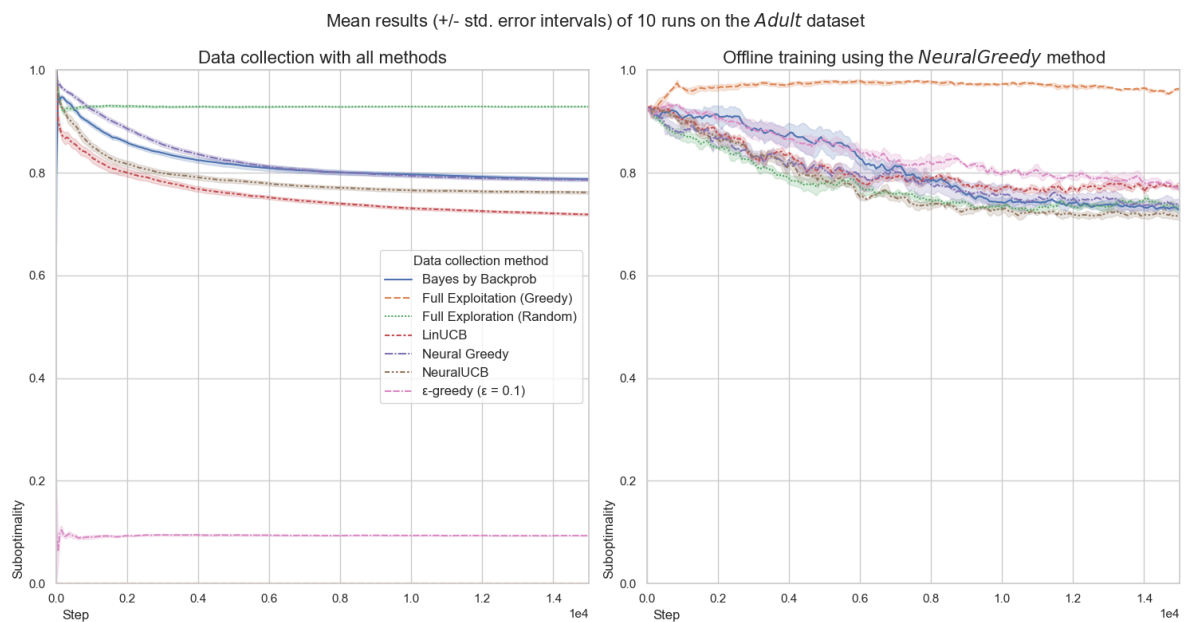


Figure 2: The *NeuralGreedy* offline method results in the *Adult* problem.

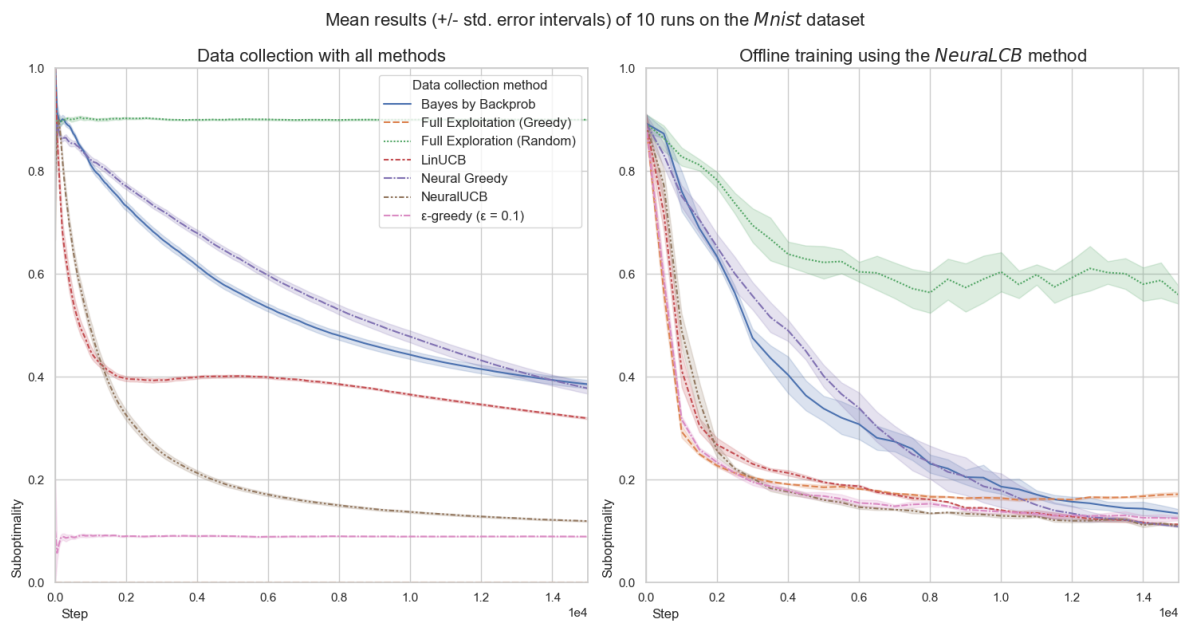


Figure 3: The NeuralLCB offline method results in the MNIST problem.

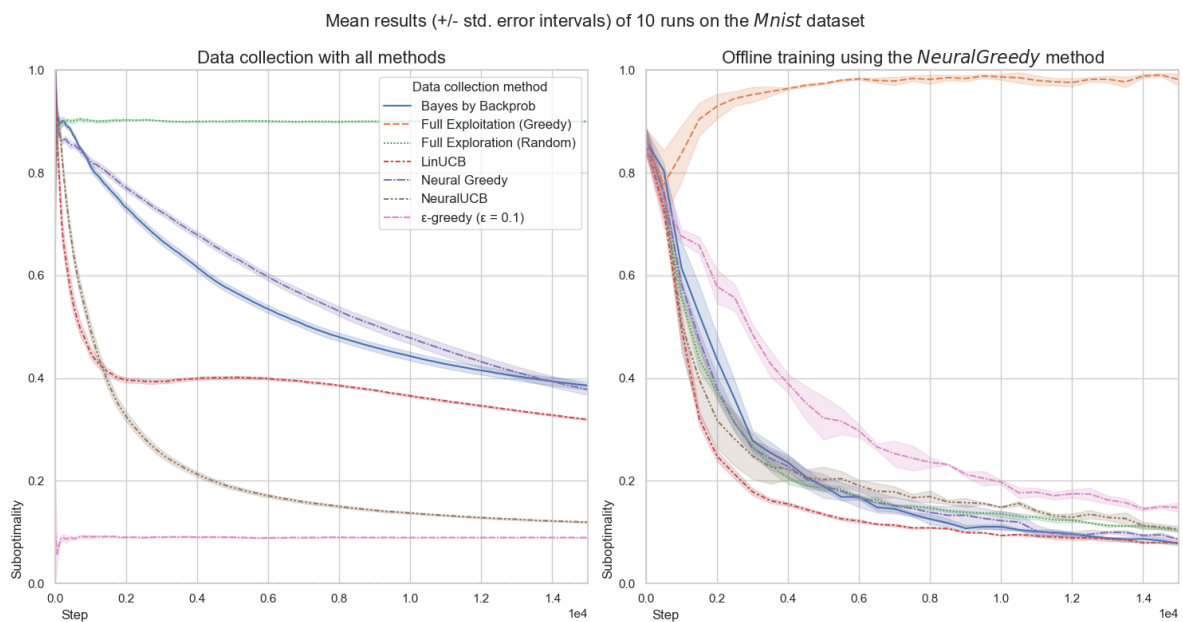


Figure 4: The NeuralGreedy offline method results in the MNIST problem.

Mean results (+/- std. error intervals) of 10 runs on the *Mushroom* dataset

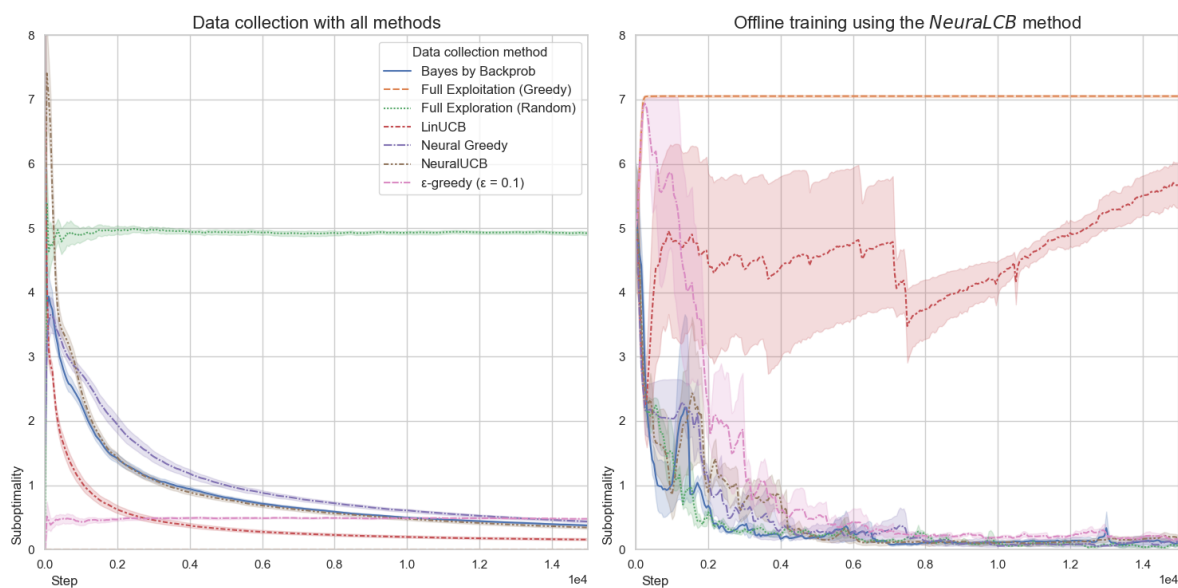


Figure 5: The NeuralLCB offline method results in the Mushroom problem.

Mean results (+/- std. error intervals) of 10 runs on the *Mushroom* dataset

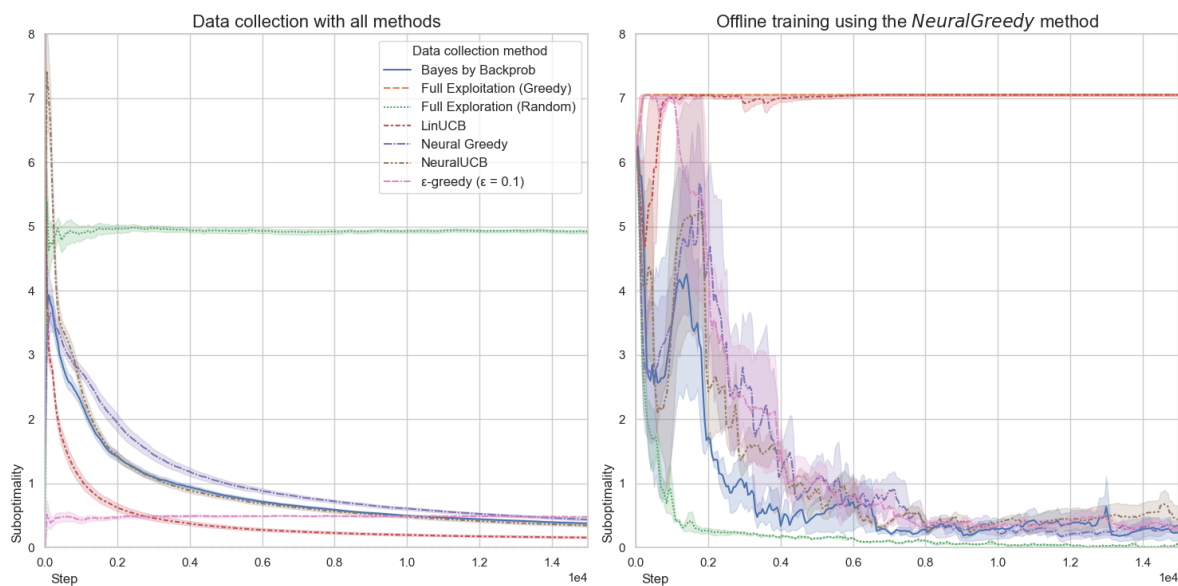


Figure 6: The NeuralGreedy offline method results in the Mushroom problem.

Mean results (+/- std. error intervals) of 10 runs on the *Statlog* dataset

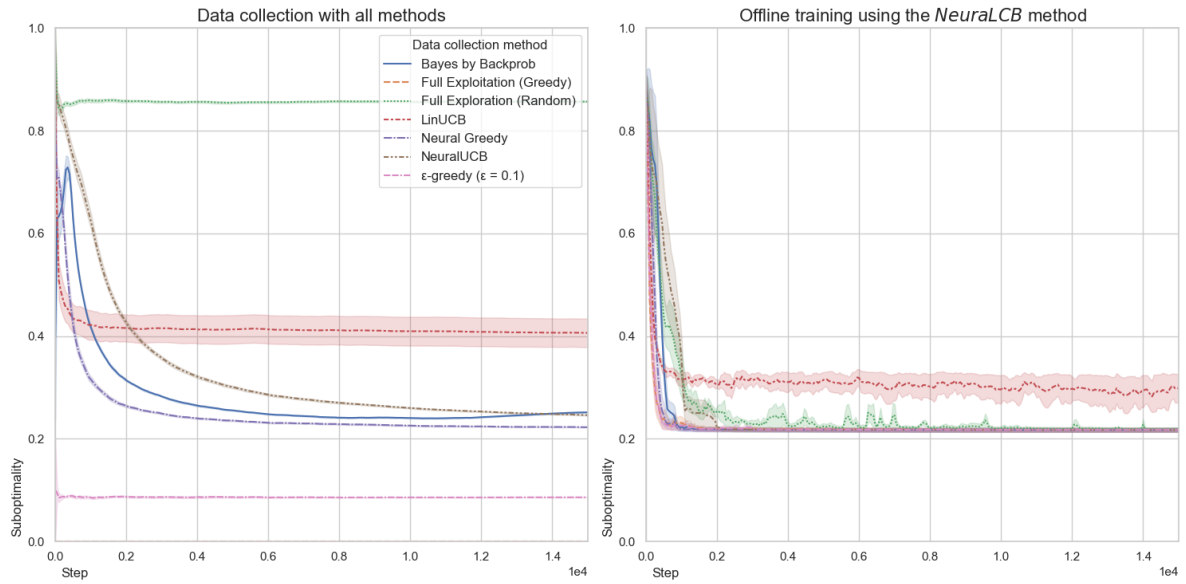


Figure 7: The NeuralLCB offline method results in the Shuttle problem.

Mean results (+/- std. error intervals) of 10 runs on the *Statlog* dataset

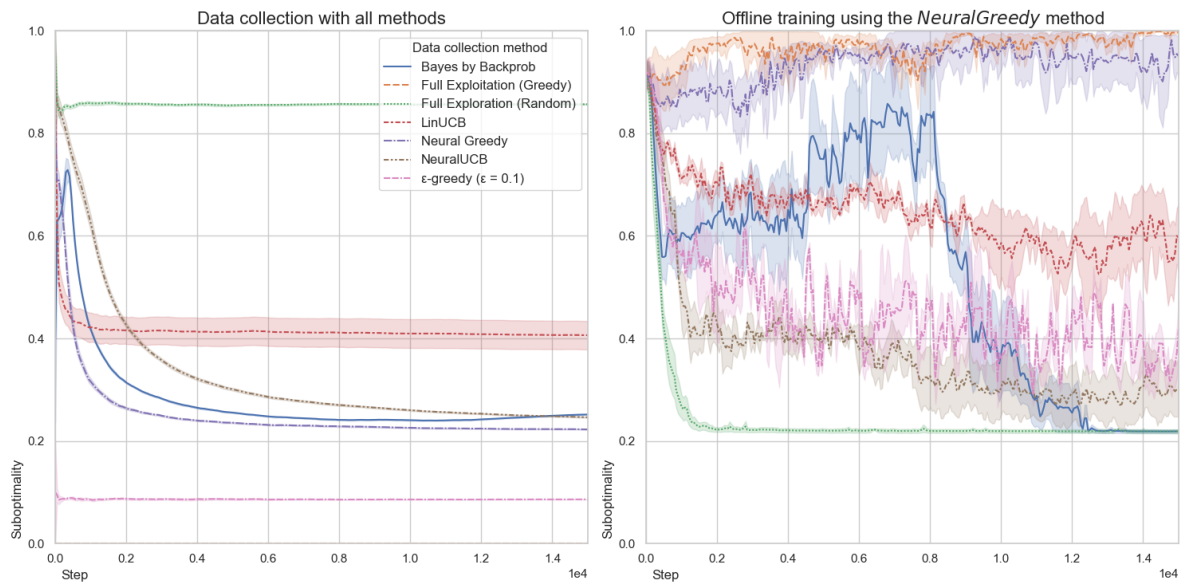


Figure 8: The NeuralGreedy offline method results in the Shuttle problem.