Evaluating Performance of Bandit Algorithms in Non-stationary Contextual Environments

Author: Weicheng Hu
Supervisor: Julia Olkhovskaia

[1]**EEMCS, Delft University of Technology, The Netherlands**

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 23, 2024

Name of the student: Weicheng Hu
Final project course: CSE3000 Research Project
Thesis committee: Julia Olkhovskaia, Ranga Rao Venkatesha Prasad

Abstract

This thesis investigates the performance of various bandit algorithms in non-stationary contextual environments, where reward functions change unpredictably over time. Traditional bandit algorithms, designed for stationary settings, often fail in dynamic real-world scenarios. This research evaluated the adaptability and computational performance of popular algorithms such as UCB, LinUCB, and LinEXP3 using a self-implemented bandit framework. Empirical results reveal significant insights into the trade-offs and optimal strategies for applying these algorithms in non-stationary conditions. Notably, LinEXP3 demonstrated superior performance in complex environments due to its ability to incorporate Bayesian posteriors, despite its higher computational cost. The key contributions of this paper include the empirical evaluation of these algorithms and their implementations, with tailored environment settings. The results suggest promising directions for further research, including the incorporation of broader algorithmic ranges like Contextual Thompson Sampling and other reinforcement learning algorithms adapted for linear contextual settings. Additionally, future work should focus on using real-world datasets to validate these algorithms and introducing covariance matrices for context vectors to simulate more realistic learning processes. These findings could influence the design and implementation of bandit algorithms in practical applications such as recommendation systems and financial portfolio management.

1 Introduction

Traditional bandit algorithm: Bandit algorithms are inspired by a decision-making scenario faced by a gambler at a slot machine, where each arm represents a different decision with uncertain rewards[1]. Mathematically speaking, each arm represents an unknown distribution, and the reward is a draw from the distribution of the chosen arm. These distributions can be of any type, such as Bernoulli, Normal, or Uniform. The primary challenge in traditional bandit problems is that the player cannot observe the reward distributions. The aim is to find the arm that will give the highest cumulative reward over time by balancing exploration and exploitation.

Contextual bandit algorithm: Building on the traditional bandit problems, contextual bandit algorithms introduce an additional layer of complexity. Before the selections, the arm vectors drawn from each arm distribution are revealed to the player. The arm vectors are called **context** in this setup. However, there exist some hidden vectors that affect the reward. For example, the reward can be modeled as a simple linear function involving the context:

$$\text{reward} = \mathbf{a} \cdot \mathbf{x} + b \tag{1}$$

where $\mathbf{a}$ is a hidden vector, $\mathbf{x}$ is the context, and b is some negligible noise. The reward function can be anything more complex. Despite the differences, the goal is still to learn a policy that maps contexts to arm pulls in a way that maximizes cumulative reward.

Challenges: While many existing works have addressed the efficiency of traditional bandit algorithms and proposed bandit algorithms in non-stochastic environments[2] and further literature proposed a rich study on contextual bandits[3], most of the previous literature assumed the environment to be stationary, but in more realistic scenarios, the environments are often non-stationary. For example, the hidden vectors that influence the reward could change unexpectedly. Take a **Recommendation System** as an example: It describes a system that needs to decide which advertisements to display to users, different ads provide different rewards: the rewards can be click-through rates or revenue per click, which are not known to the system. The reward of each ad is unlikely to be stationary, instead, different rewards might be given based on when the ad is shown, due to a series of factors, like changing user tastes, competitor actions, or seasonal changes. This raises the main aim of this research: find the bandit algorithm that outperforms the rest when the environment is non-stationary. This research seeks to test some popular bandit algorithms in simulated environments.

Research Questions: Since the study focuses on identifying the most effective MAB algorithms for non-stationary environments, the research questions are closely relevant:

1. Which bandit algorithms best adapt to different non-stationary environments?
2. What are the trade-offs associated with these algorithms in terms of computational performance and algorithm optimum?

Contributions: The main contributions of this research are the empirical evaluation of several leading bandit algorithms in changing linear contextual environments and a supplementary implementation that supports adaptability in non-stationary conditions. These contributions may provide insights that could influence future algorithm design focuses. On top of this, an existing implementation in applied settings.

Conclusions: Despite that in theory, it is expected LinEXP3 and LinUCB should outperform EXP3 and UCB1, however, this is not always the case.

Structure: The rest of the paper is organized as follows: Section 2, the paper will discuss the methodologies and introduce the problem description. In section 3 the paper will present the experimental setups and main results. Section 4 will share the conclusion of this paper. Section 5 discusses the ethical parts of the study. Finally, Section 6 denotes some possible improvements and future works for researchers who are willing to continue the study.

2 Methodologies

2.1 Methodology and Background

To address the research question, this study used Python and several of its popular libraries: Pandas for data preprocessing and Matplotlib for data visualization. Additionally, the study heavily relied on the SMPyBandits framework[4] by Lilian

Besson, which provides implementations of various bandit algorithms and environments. For the specific purposes of this study, the library was slightly modified to support contextual environments and newly developed algorithms that were not part of the original framework.

2.2 Formal Problem Description

The experiment involves a linear contextual bandit setup, where an agent interacts with a dynamic environment over multiple rounds. This section formally describes the problem and introduces related concepts.

Problem Setup

Consider an adversarial interaction between a learner (agent) and a set of k context vectors $\{\mathbf{x}_{t,a} \mid a = 1, 2, \ldots, k\} \subseteq \mathbb{R}^d$, where the context vectors are generated from some independent and identically distributed (i.i.d.) distributions. The interaction proceeds in discrete time steps $t = 1, 2, \ldots, T$. At each time step t, the following steps are executed:

1. **Policy-based Selection**:
   - The agent selects a context vector $\mathbf{x}_{t,a_t}$ from the set of available context vectors $\{\mathbf{x}_{t,1}, \mathbf{x}_{t,2}, \ldots, \mathbf{x}_{t,k}\}$ based on its policy $\pi_t(x)$. The context vectors are revealed to the agent.
2. **Hidden Reward Vector**:
   - Independently of the choice of the context vector, there exists a hidden reward vector $\boldsymbol{\theta}_t \in \mathbb{R}^d$ that is unknown to the agent. The hidden reward vector $\boldsymbol{\theta}_t$ is non-stationary.
3. **Reward Observation**:
   - The agent observes the reward r_{t,a_t}, which is obtained by taking the dot product of the selected context vector $\mathbf{x}_{t,a_t}$ and the hidden reward vector $\boldsymbol{\theta}_t$:

$$r_{t,a_t} = \mathbf{x}_{t,a_t} \cdot \boldsymbol{\theta}_t$$

4. **Learning**: Based on the observed reward r_{t,a_t}, the agent updates its weights for each arm and learns a better decision-making strategy.
5. **Objective**:
   - The agent's objective is to minimize the cumulative regret over T time steps. There are differences in how regrets are calculated between papers, in this paper, the **best-fixed policy regret** would be used:

$$R_T = \sum_{t=1}^{T} \left(r_{t,\pi_T^*(x)} - r_{t,\pi_T(x)} \right)$$

   where an assumption is made about a theoretically optimal policy $\pi_T^*(x)$ which always selects context vectors that yield the highest possible rewards in every t, the cumulative regret is then equal to the cumulative reward achieved by $\pi_T^*(x)$ minus the cumulative reward achieved by the policy of choice.

Algorithms

The study explores various bandit algorithms adapted to the contextual setting. Some state-of-art algorithms in this domain include but are not limited to:

- **UCB[5]**:
  - The basic UCB algorithm that uses upper confidence bounds to balance exploration and exploitation. It was used as one of the baseline agents for comparison.
- **EXP3[2]**:
  - The EXP3 algorithm is designed for adversarial settings, where the reward distributions are not assumed to be stationary. It adapts to the observed rewards and balances exploration and exploitation probabilistically.
- **LinUCB[3]**:
  - An adapted UCB algorithm that models the reward as a linear function of the context vector and uses upper confidence bounds to balance exploration and exploitation.
- **LinEXP3[6]**:
  - A new probabilistic algorithm built upon EXP3[2] and LinUCB. It uses both context information and Bayesian inference to make decisions.

3 Experimental Setup and Results

To know how well a policy learns and how it could be helpful in the real world, selected algorithms are run against an artificial dataset to verify their reliability and theoretical correctness.

3.1 Environment: Artificial data

To simulate recurring events, we consider several coordinates of the reward vector:

1. $\theta_{t,a}^i = |\sin(w_i t + \phi_i)|$
2. $\theta_{t,a}^i = |\cos(w_i t + \phi_i)|$
3. $\theta_{t,a}^i = |\log(t) \cdot (w_i + \phi_i)|$

where w_i and ϕ_i are fixed vector chosen for each direction $i \in [d]$, t could be seen as the current timestamp.

The reason for choosing sin and cos is because they could model seasonal changes. For example, presenting ice cream to users in summer is generally more attractive than presenting them in winter. Likewise, the same context may give desirable or undesirable results based on the period, which represents periodic changing environments. Logarithm could be seen as a slowly changing environment.

Ten 3d-context vectors with different characteristics are prepared. The definition of characteristics is based on the values in each dimension. For example, low value in the first dimension and high in the second and third; medium value across all directions to include more variances. For each iteration, the algorithm takes context $x_{t,a}$.

To ensure the accuracy of the algorithms and provide a better understanding, Each algorithm is executed in different environmental settings for 20 realizations, 5000 epochs each, with their cumulative regrets, and corresponding normal distributions plotted.

3.2 Algorithm setup

For each algorithm, attempts are made to fine-tune the parameters, to make them perform optimally in the specific environment settings. A parameter is optimal if the algorithm reaches the highest average cumulative reward. This section will discuss the choices of values and why these values are chosen. A random policy that always chooses arms randomly is added as a baseline agent.

1. **UCB**: In the past years, many papers proposed different versions of UCB. This study will stick to UCB1 which was proposed by Auer[7]. This is because UCB1 does not introduce additional parameters, it chooses arms according to a fixed formula. This not only reduces the cost and time complexity, but also produces no variances between realizations, which makes it a perfect baseline policy.

2. **EXP3**: A first attempt is made to tune γ on a logarithmic scale. However, for γ ranged in $[10^{-6}, 10^{-1}]$, EXP3 behaves randomly. For γ above 0.95, the agent converges early, which leads to a local minimum. We then did some rounds for grid search for γ between 0.1 and 0.8 and found 0.15 the most optimal. Since EXP3 chooses arms according to Bayesian posteriors, to prevent overflows during the weights calculation, log-sum-exp trick[8] is applied to the probability calculation of the original algorithm. The plot of the first round's grid search results is shown in Figure 1 and 2.

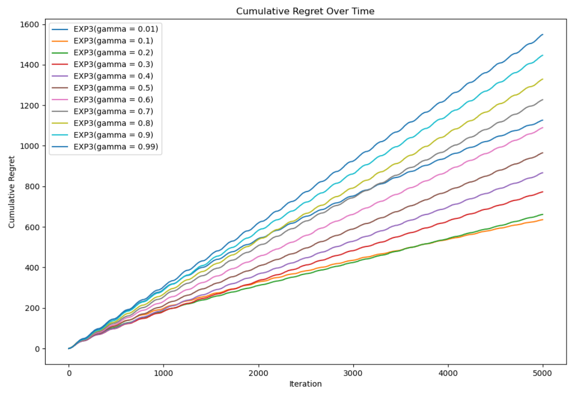

Figure 1: EXP3 Parameter Grid Search in Trignometric Environment

3. **LinUCB**: In LinUCB, there is one controllable parameter α, which is an exploration term. Again, if α is set too high, LinUCB is more likely to explore and behave randomly. On the other hand, if α is set too low, the algorithm might exploit too much, potentially missing out on better arms that haven't been explored enough. After some rounds of grid search for α, we found that α=0.3 performs the best overall. The plot of the first round's grid search results is shown in Figure 3 and 4.

4. **LinEXP3**: There is a learning rate parameter η and an exploration term γ. For the ideal setting, η is set to 0.5 and

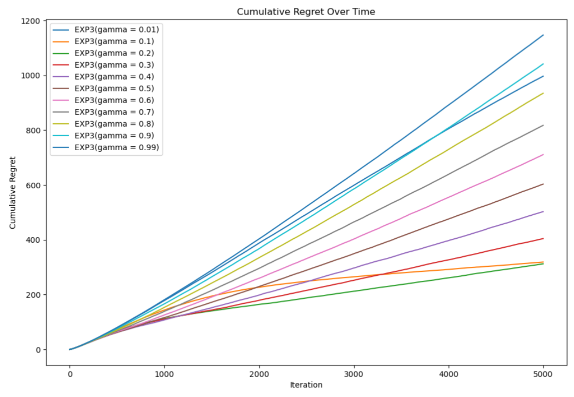

Figure 2: EXP3 Parameter Grid Search in Logarithmic Environment

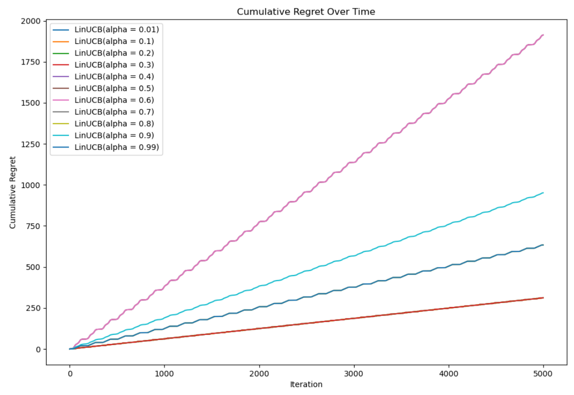

Figure 3: LinUCB Parameter Grid Search in Trignometric Environment

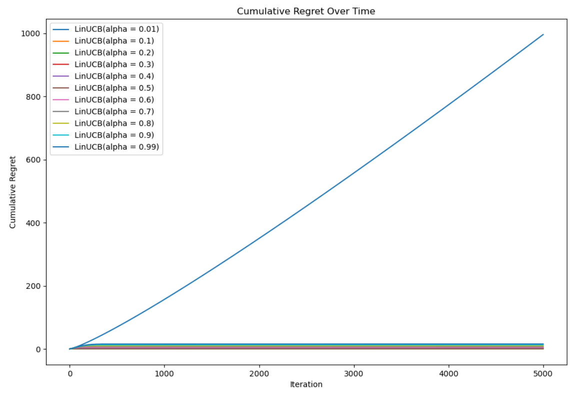

Figure 4: LinUCB Parameter Grid Search in Logarithmic Environment

γ to 0.2. If η is low or γ is high, the algorithm will perform randomly; If η is high or γ is low, the algorithm will quickly converge to a local minimum. Again to prevent overflows during the weight calculation, log-sum-exp trick is applied. Moreover, the literature LinEXP3 proposed a sampling method called MGR[6] which included some tuneable parameters, namely **M**, the number of samples taken from the context distribution, and β, which is used to calculate variances. In the experiment, β is set to 0.5 according to the optimal value formula in the paper, but **M** is only set to 1500 instead of the optimal value, due to limited computational resources. The plots for grid searches will not be displayed as there are too many parameters to tune, but they can be found in the TU repository.

Results

For trigonometric environments combined, the average cumulative regret is calculated and illustrated in Figure 5. As for the logarithm environment, the average cumulative regret is calculated and shown in Figure 6.

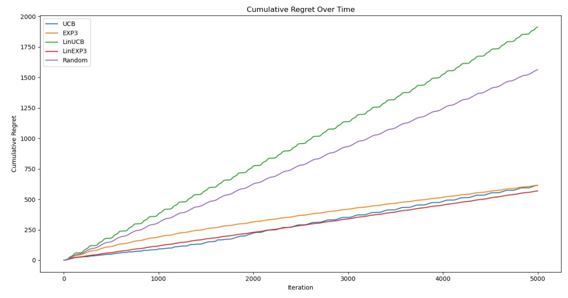

Figure 5: Performance of algorithms in trigonometric environment

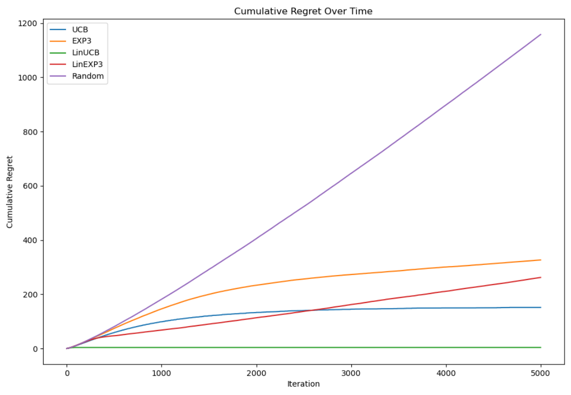

Figure 6: Performance of algorithms in logarithmic environment

Additionally, the normal distributions of the results can be found in Figure 7 and 8 respectively.

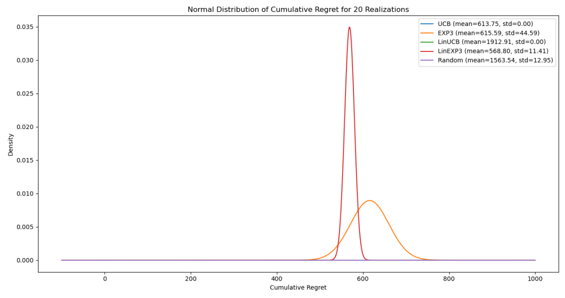

Figure 7: Normal distributions in trigonometric environment

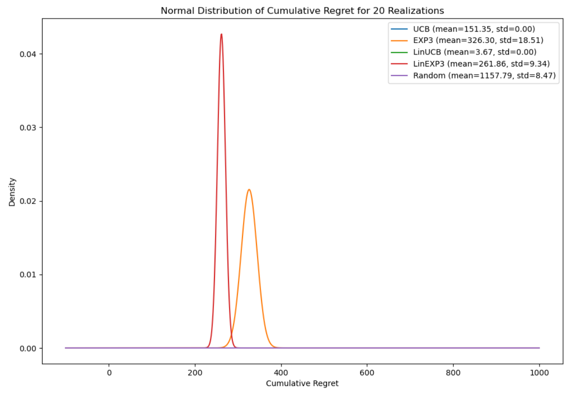

Figure 8: Normal distributions in logarithmic environment

As observed, the standard deviation of UCB1 and LinUCB is zero since they choose arms according to a fixed formula. LinEXP3 has a moderate density curve, indicating standard variability. As for EXP3, it has a wider density curve, showing that there is a higher variability, which implies the algorithm's performance is less consistent.
The time taken per algorithm in experimental settings is shown in Table 1, the numbers are gotten by summing up the time taken for the algorithms to finish in all environments, then normalized to the second decimal place for a better visual comparison:

Algorithm	**Avg. Computation Time**
UCB1	0.02
LinUCB	0.08
EXP3	0.02
LinEXP3	0.87
Random Policy	0.01

Table 1: Computation costs per algorithm

4 Conclusions

From the theoretical hypotheses, algorithms not adapted to contextual environments perform worse than contextual algorithms. However, the experimental results have shown this is not the case. The outcomes can be attributed to the following factors:

1. **UCB1**: This algorithm does not utilize contextual information when making decisions, which is crucial for environments where context significantly impacts the reward. Therefore, in contextual environments, it often converges slower than LinUCB. Figure 6 also verified this fact. In non-stationary environments where the best arm changes frequently, UCB1 performed worse, but better than LinUCB in the experiments because contexts could sometimes provide misleading information.

2. **LinUCB**: This algorithm leverages contextual information to inform decisions. In the logarithmic environment, it converges the fastest, because it finds the best arm immediately by observing the contexts. Unfortunately, in non-stationary settings, where the best arm keeps changing, LinUCB becomes stuck in a local minimum quickly, repeatedly selecting the arm it believes to be the best, which might turn out to be the worst in later stages, causing LinUCB to perform even worse than a random selection.

3. **EXP3**: This algorithm is designed for adversarial settings, meaning it does not assume that the reward distributions are stationary. Instead, it continuously adapts to the observed rewards, making it capable of quickly responding to changes in environments. In both trigonometric and logarithmic environments LinUCB adapts well, with slightly better performance in trigonometric environments due to its probabilistic manner of arm selection. However, it does not utilize contextual information, where some useful data could be inferred.

4. **LinEXP3**: Among the algorithms, LinEXP3 demonstrated superior performance in the more complex trigonometric environment. Like EXP3, LinEXP3 accounts for non-stationarity by incorporating Bayesian posteriors. This approach allows LinEXP3 to occasionally select sub-optimal arms, resulting in lower cumulative regret over time. While LinUCB and UCB1 assume a fixed optimal arm, leading to suboptimal performance in these environments. On top of adaptability in non-stationary environments, LinEXP3 also infers contextual information, which makes it capable of making more informed choices. However, to improve the estimator of the hidden reward vectors, LinEXP3 uses a sampling method called Geometric Matrix Resampling [6], which requires sampling many times from the context distributions during every selection. As the number of samples grows with the complexity of environments, it could be computationally expensive. In real-time systems, where immediate feedback is crucial, LinEXP3 might not always be the most suitable choice. Due to the computation resource constraint mentioned earlier, the number of samplings per round is decreased. It is expected that LinEXP3 would perform even better if resources allow.

The normal distribution graph shows the variability and performance of the different algorithms. Using the data from the normal distributions, their 95% confidence intervals can be calculated to show the range within which the true performance is expected to lie in both environments:

Trigonometric Environment

Algorithm	Lower Bound	Upper Bound
UCB	613.75	613.75
EXP3	596.05	635.13
LinUCB	1912.91	1912.91
LinEXP3	563.80	573.80
Random	1557.86	1569.22

Table 2: Confidence Intervals for Trigonometric Environment

Logarithmic Environment

Algorithm	Lower Bound	Upper Bound
UCB	151.35	151.35
EXP3	318.19	334.41
LinUCB	3.67	3.67
LinEXP3	257.77	265.95
Random	1154.08	1161.50

Table 3: Confidence Intervals for Logarithmic Environment

This way, it is more straightforward to observe that:

- **UCB1 and LinUCB**: These provide highly consistent performance with zero variability due to their deterministic arm selection mechanisms. They are stable and predictable.
- **EXP3**: This algorithm shows significant variability, reflecting its probabilistic nature and capability to adapt to changing environments.
- **LinEXP3**: This algorithm displays moderate variability, indicating that while it incorporates contextual information, the randomness in its decision-making process introduces some unpredictability in performance.

Another noticeable trend in logarithmic environments is that LinEXP3 found the best arm faster than UCB1, at about 300 iterations. However, due to its probabilistic way of selecting arms, it still occasionally chooses suboptimal arms, causing it not to converge, at about 2600 iterations UCB starts to stay at the best arm which causes it to beat LinEXP3 by then. To cope with this situation, one could introduce a method called learning rate decay[9] to reduce exploration over time.
In conclusion, to answer the research question in the beginning:

1. Which bandit algorithms best adapt to different non-stationary environments?
2. What are the trade-offs associated with these algorithms in terms of computational performance and algorithm optimum?

The answer to the questions is that the best bandit algorithm depends on the nature of the specific environments and constraints of the application context. UCB1 and LinUCB are suitable for stable environments where the best arm is fixed and stability is desired. EXP3 offers better adaptability across various environments, albeit with higher variance, making it a good choice when the environment is unknown. LinEXP3, on the other hand, demonstrates reliable performance in environments where the optimal arm changes frequently, but it does not pull away from other algorithms in simpler environments. Given the performance and as observable in Table 1, which indicates LinEXP3 uses nearly 90% out of total resources to run, other algorithms may still be preferable due to their time efficiency. Consequently, LinEXP3 is better in environments where higher precision is the most required.

5 Responsible Research

This study adheres to the principles of responsible research, emphasizing ethical considerations and reproducibility. The research process is guided by the FAIR principles, which stands for Findable, Accessible, Interoperable, and Reusable, to enhance transparency and impact:

1. **FAIR Principles**:
   - **Findable**: All code and data generated during this research will be deposited in public repositories such as the TU Repository[10] and the author's GitHub Repository. These resources will be findable by detailed metadata and unique identifiers to allow easy access for other researchers.
   - **Accessible**: The data and code are openly accessible without restrictive barriers. This paper will be provided as a reference to ensure that other researchers can easily understand and utilize the repository.
   - **Interoperable**: The research outputs adhere to standardized formats, and utilize popular Python frameworks such as MatPlotLib and NumPy. This promotes compatibility with other datasets and tools, which leads to broader usage and integration.
   - **Reusable**: Methodologies and metadata are provided in the paper to maximize the reusability of the code, and to enable other researchers to replicate, verify, and build upon the work.
2. **Data Privacy**: The study employs artificial data that has no connection to real-world data, in compliance with GDPR[11]. This approach eliminates the risk of privacy breaches and safeguards sensitive information.
3. **Proper Citation**: All sources of data, information, and code that influenced this research are properly cited in IEEE style. The references list includes details for each source, ensuring proper acknowledgment and facilitating further investigation.
4. **Use of LLM**: Large Language Models (LLMs), specifically ChatGPT[12], have been utilized responsibly in this research. LLMs mainly assisted in code debugging and rephrasing the paper into a more academic writing style. In some use cases where LLM are asked about related literature, this kind of use was carefully managed, with human researchers evaluating and validating all generated responses to avoid inaccuracies, such as non-existent references.

6 Discussions and Future Work

This study has explored the application and performance of contextual bandit algorithms in various settings. It addressed key research questions regarding the adaptation and performance of these algorithms in non-stationary linear contextual settings. The findings suggest several avenues for future research and improvements.

Firstly, incorporating a broader range of algorithms, such as Contextual Thompson Sampling, could enhance the reliability of the analysis. Further exploration into other reinforcement learning algorithms that can be adapted for linear contextual settings is also warranted, in this way proposing new algorithms inspired by earlier works could potentially improve the performance of contextual bandit algorithms. For instance, combining the settings of LinEXP3 with elements from algorithms like Exp4, as described by Beygelzimer et al. (2011) [13], could be a good direction.

Moreover, the current study used artificial data to validate the algorithms, which are not directly correlated to real-world scenarios. Using the artificial data allows for controlled experimentation and initial proof of concept. However, it lacks the complexity and variability found in real-world data. This can limit the applicability and generalizability of the findings. In future research, it is recommended to use real-world datasets to validate the algorithms. One example dataset is the Criteo Attribution Modelling dataset [14] available on Kaggle. However, due to the complexity of the dataset, it is not suitable to use it in this scope due to limited computational resources and analysis complexity. The given instance is only an illustration, it is recommended to find a similar but simpler dataset.

If future researchers wish to continue using artificial data, it is a good idea to introduce covariance matrices for the context vectors. Because the current study only cares about the expectation of cumulative regret, the contexts are set to be fixed for convenience. However, variances might introduce a bigger impact on the outcome of experiments. Therefore, changing the fixed contexts to normal distributions with different variances could add another layer of complexity, and denote a more realistic learning process.

Acknowledgements

Lastly, I would like to express my deepest gratitude to my supervisor, **Julia Olkhovskaia**, for all her guidance in every meeting, her on-time feedback, and her patience. My research skills are heavily sharpened throughout the process with the help. I would also like to thank my research group, they provided invaluable assistance to me in their spare time. Their willingness to answer my questions and offer coding advice helped me overcome numerous challenges. Their suggestions and ideas were crucial in developing the coding aspects of this research.

References

[1] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. MIT Press, 2018.

[2] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire, “The nonstochastic multiarmed bandit problem,” *SIAM journal on computing*, vol. 32, no. 1, pp. 48–77, 2002.

[3] W. Chu, L. Li, L. Reyzin, and R. Schapire, “Contextual bandits with linear payoff functions,” in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. JMLR Workshop and Conference Proceedings, 2011, pp. 208–214.

[4] L. Besson, “SMPyBandits: an Open-Source Research Framework for Single and Multi-Players Multi-Arms Bandits (MAB) Algorithms in Python,” Online at: GitHub.com/SMPyBandits/SMPyBandits, 2018, code at https://github.com/SMPyBandits/SMPyBandits/, documentation at https://smpybandits.github.io/. [Online]. Available: https://github.com/SMPyBandits/SMPyBandits/

[5] P. Auer, “Using confidence bounds for exploitation-exploration trade-offs,” *Journal of Machine Learning Research*, vol. 3, pp. 397–422, 2002.

[6] G. Neu and J. Olkhovskaya, “Efficient and robust algorithms for adversarial linear contextual bandits,” in *Conference on Learning Theory*. PMLR, 2020, pp. 3049–3068.

[7] P. Auer, N. Cesa-Bianchi, and P. Fischer, “Finite-time analysis of the multiarmed bandit problem,” *Machine learning*, vol. 47, no. 2-3, pp. 235–256, 2002.

[8] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[9] L. Bottou, “Online learning and stochastic approximations,” in *Online Learning in Neural Networks*, D. Saad, Ed. Cambridge University Press, 1998, pp. 9–42.

[10] “Tu delft repository,” 2024, accessed: 2024-06-03. [Online]. Available: https://repository.tudelft.nl/

[11] E. Parliament and C. of the European Union, “Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data (general data protection regulation),” 2016, accessed: 2024-06-03. [Online]. Available: https://eur-lex.europa.eu/eli/reg/2016/679/oj

[12] OpenAI, “Gpt-4,” https://www.openai.com/research/gpt-4, 2023.

[13] A. Beygelzimer, J. Langford, L. Li, L. Reyzin, and R. E. Schapire, “Contextual bandit algorithms with supervised learning guarantees,” in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. JMLR Workshop and Conference Proceedings, 2011, pp. 19–26.

[14] Diemert Eustache, Meynet Julien, P. Galland, and D. Lefortier, “Attribution modeling increases efficiency of bidding in display advertising,” in *Proceedings of the AdKDD and TargetAd Workshop, KDD, Halifax, NS, Canada, August, 14, 2017*. ACM, 2017, p. To appear.