Privacy-Preserving UCB Decision Process Verification via zk-SNARKs

Xikun Jiang[1,3] , **He Lyu**[3] , **Chenhao Ying**[1,2*] , **Yibin Xu**[3] , **Boris Düdder**[3] , **Yuan Luo**[1,2]

[1]Department of Computer Science, Shanghai Jiao Tong University, China
[2]Shanghai Jiao Tong University (Wuxi) Blockchain Advanced Research Center, China
[3]Department of Computer Science, University of Copenhagen, Denmark

{xikunjiang, yingchenhao,yuanluo}@sjtu.edu.cn, lyuhe2000@gmail.com, {xikun, yx, boris.d}@di.ku.dk

Abstract

With the increasingly widespread application of machine learning, how to strike a balance between protecting the privacy of data and algorithm parameters and ensuring the verifiability of machine learning has always been a challenge. This study explores the intersection of reinforcement learning and data privacy, specifically addressing the Multi-Armed Bandit (MAB) problem with the Upper Confidence Bound (UCB) algorithm. We introduce zkUCB, an innovative algorithm that employs the Zero-Knowledge Succinct Non-Interactive Argument of Knowledge (zk-SNARKs) to enhance UCB. zkUCB is carefully designed to safeguard the confidentiality of training data and algorithmic parameters, ensuring transparent UCB decision-making.

Experiments highlight zkUCB's superior performance, attributing its enhanced reward to judicious quantization bit usage that reduces information entropy in the decision-making process. zkUCB's proof size and verification time scale linearly with the execution steps of zkUCB. This showcases zkUCB's adept balance between data security and operational efficiency. This approach contributes significantly to the ongoing discourse on reinforcing data privacy in complex decision-making processes, offering a promising solution for privacy-sensitive applications.

1 Introduction

The integration of reinforcement learning (RL) algorithms in various fields, such as healthcare, autonomous driving, and recommendation systems, signifies a major advancement in artificial intelligence [Yu *et al.*, 2021; Afsar *et al.*, 2022; Kiran *et al.*, 2021]. The Multi-Armed Bandit (MAB) problem, a core RL model, is particularly notable for its efficient decision capabilities. The MAB model has found diverse applications, from optimizing healthcare treatments to enhancing digital user experiences [Patil *et al.*, 2021; Ameko *et al.*, 2020; Gao *et al.*, 2022].
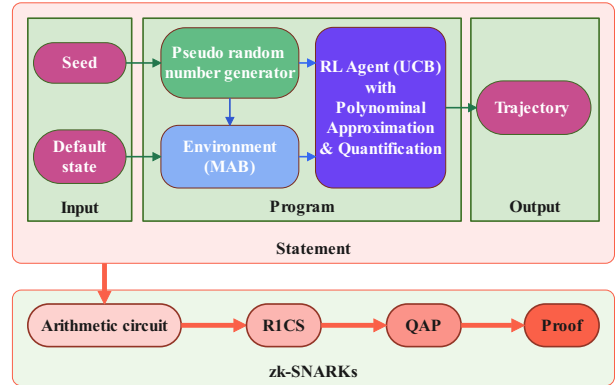
---

*Corresponding author

Figure 1: An illustration of zkUCB. We integrate zk-SNARK with the UBC algorithm. In zkUCB, the complete reinforcement learning process is encapsulated as a statement, which is used to model a deterministic arithmetic circuit that uses addition gates and multiplication gates for arithmetic calculations. The arithmetic circuit then serves as the first step in zk-SNARK execution.

However, the deployment of the MAB model in sensitive areas like healthcare involves significant privacy and security challenges [Zhao *et al.*, 2020; Azize and Basu, 2022]. For instance, in the healthcare domain where AI systems using MAB models, such as AI doctors (AI giving diagnosis to the patients), are tasked with critical responsibilities like adjusting antidepressant dosages [Aziz *et al.*, 2021]. However, this application involves many issues. (1) Data Privacy Risks: It necessitates the long-term accumulation of extensive patient data, encompassing sensitive information like physical conditions and lifestyle habits, posing the risk of privacy breaches; (2) Confidentiality of Algorithm Parameters: The algorithm's parameters, refined through comprehensive experiments and continual enhancements, hold significant value for medical institutions. Therefore, ensuring the confidentiality of these parameters is imperative; (3) Verification by Third Parties: With the involvement of numerous third-party entities such as health insurance companies and government welfare agencies, there is a need to verify the application's execution accuracy. This verification must be conducted without compromising any private data or revealing sensitive algorithm parameters.

In this context, our study introduces an innovative solution

by integrating the zero-knowledge succinct non-interactive arguments of knowledge (zk-SNARKs) [Groth, 2016; Wahby *et al.*, 2018] with the UCB algorithm in the MAB model. This approach enables AI doctors to utilize algorithm parameters and real-time patient data for effective medical advice and medication planning, while simultaneously ensuring that no additional sensitive information is disclosed. To better understand the specific workflow of this application, we give a detailed example of integrating the UCB algorithm and zk-SNARKs and applying it to the AI doctor system. Please refer to Section 1.2 and Figure. 2.

Generally speaking, applying zk-SNARKs involves converting the corresponding learning algorithm or program into a format compatible with zero-knowledge proofs. The conversion process involves first converting the algorithm or program into a deterministic arithmetic circuit and then converting that arithmetic circuit into a Rank-1 constrained system (R1CS), which is essentially a system of linear equations. These R1CS are then converted into Quadratic Arithmetic Programs (QAPs), thereby converting the linear equations into polynomial forms. QAP can then be used to generate the corresponding zk-SNARK proof, which is used to verify the output of the algorithm while protecting the confidentiality of the underlying data and the internal mechanisms of the algorithm.

Research Challenges. Thus, to integrate the UCB algorithm with zk-SNARKs to ensure the data and parameters' confidentiality while supporting secure UCB decision process verification, we need to follow the above conversion process to convert the UCB algorithm into a format compatible with zero-knowledge proofs, thus facilitating efficient verification. However, this integration presents three major challenges: (1) The inherent randomness of the UCB complicates this task; the challenge is how to transform this randomness into the deterministic process necessary for constructing the arithmetic circuit; (2) Given that zk-SNARK relies on polynomials in finite fields Operations, how to convert the non-polynomial operations of the UCB (such as logarithms and non-integer powers) into polynomial operations is the second challenge. (3) To meet the finite field requirements of R1CS, how to adjust the floating point numbers in the UCB to meet the finite field requirements of R1CS is the third challenge.

To overcome these challenges, we carefully designed the conversion process to convert UCB into a format compatible with zero-knowledge proofs. As shown in Figure. 1, we eliminate randomness by inputting seeds into the pseudo-random number generator and solve the second and third challenges respectively through polynomial approximation and quantization operations. Details on how to solve the challenge can be found in Section. 3.1. In zkUCB, we encapsulate all input, output, environmental, and the intermediate process of the UCB into one statement, and then use this statement to build a deterministic arithmetic circuit for subsequent execution of zk-SNARK.

1.1 Contributions

In this article, we address key challenges in merging UCB with zk-SNARKs, focusing on data confidentiality and transparent decision process verification. We employ
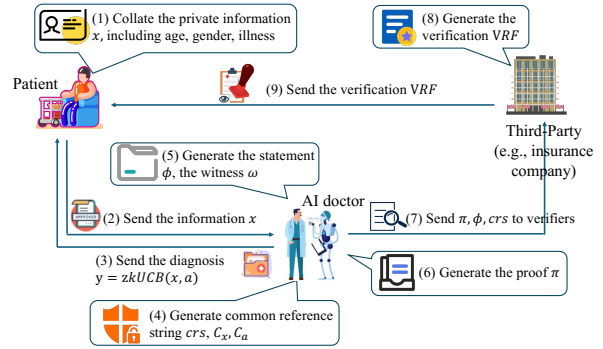

Figure 2: Application of zk-SNARKs in the AI Doctor System Using UCB Algorithm.

Groth16 [Groth, 2016] for its efficient zk-SNARKs implementation, notable for its fast verification process and suitability for large blockchain networks. Additionally, Groth16's proofs are compact, facilitating easy transmission and verification, crucial for large-scale applications. Our main contributions include:

1. **A new framework:** We introduce the Zero-Knowledge Proof into the MAB problem, a novel integration of zk-SNARKs (specify to Groth 16) with the UCB algorithm (an approach for addressing the MAB problem).

2. **Theory that links UCB and zero-knowledge proofs:** We convert the UCB algorithm into arithmetic circuits compatible with zero-knowledge proofs and overcome the challenges of randomness, non-polynomial operations, and floating point numbers in UCB. This transformation secures precise algorithm output validation and maintains data and parameter confidentiality, resulting in a transparent decision process verification.

3. **A throughout evaluation:** Our experiments demonstrate that zkUCB outperforms the standard UCB in terms of reward. This means that our algorithm makes decisions that are closer to optimal than before the improvements. This improvement is because appropriate quantization bits can effectively reduce information entropy in the decision-making process, enhancing rewards. Particularly noteworthy is the fact that the proof size for zkUCB is proportional to the number of execution steps within the algorithm, suggesting that zkUCB maintains a manageable scale even when dealing with large datasets. This aspect is crucial in applications where balancing data security and operational efficiency is paramount.

1.2 Use Case

Figure. 2 illustrates the application of zk-SNARKs to the AI doctor using the UCB algorithm. Here, the AI doctor functions as the prover, while the verifier could be either the patient themselves or a third party, such as an insurance company. Initially, the AI doctor gathers the patient's information, denoted as x, and computes a diagnostic result y using the $zkUCB$ model and parameters a. Subsequently, the

AI doctor creates a common reference string crs and formulates commitments C_x and C_a, based on x and a respectively. The next step involves the AI doctor generating a statement $\phi = (C_x, C_a, y)$ and a witness $w = (x, a)$, with the latter serving as a confidential input in the subsequent phase. The AI doctor then produces a proof $\pi = (crs, \phi, w)$ and dispatches π, the crs, and the statement ϕ to the third party for verification $VRF = (crs, \phi, \pi)$. After verification, the results VRF are relayed to the patient. This ensures that commitments to the parameters and private data, along with the computational results and generated proofs, are presented to third parties for verification, all while safeguarding the integrity and confidentiality of both the data and algorithm.

1.3 Organization

The subsequent sections of this paper are structured as follows: In Section 2, we provide essential preliminaries. Section 3 delves into an in-depth discussion of zkUCB. Moving forward to Section 4, we present a thorough examination of comprehensive experimental results and analysis. In Section 5, we review the related work, highlighting the distinctions between our approach and existing methodologies. Finally, Section 6 encapsulates the paper with concluding remarks.

2 Preliminaries

2.1 UCB Algorithm in MAB Problems

The Multi-Armed Bandit (MAB) problem involves sequentially choosing from options with uncertain rewards to maximize cumulative rewards. The Upper Confidence Bound (UCB) algorithm addresses this challenge by balancing exploration (testing new options) and exploitation (leveraging known options). It achieves this equilibrium by constructing upper confidence bounds for each option, reflecting both anticipated rewards and uncertainty.

In this study, we focus on UCB1, a specific UCB algorithm version. The algorithm's workflow is outlined in Alg. 1. It begins with an "Initialization" phase, playing each arm i once to establish initial reward estimates (x_i). Following this, the algorithm updates the average reward $\bar{x}_i$ for each arm i after it is played. Subsequently, the algorithm calculates a UCB value $\bar{x}_i + \sqrt{\frac{2 \cdot \ln n}{n_i}}$ for each arm i and selects the arm with the highest UCB value. This strategic choice guides the algorithm toward achieving an optimal balance between exploring new arms and exploiting known ones.

2.2 A General zk-SNARKs Scheme

Zero-Knowledge Succinct Non-Interactive Argument of Knowledge (zk-SNARKs) [Chen *et al.*, 2022] is a novel cryptographic protocol allowing a prover to authenticate a statement's truth to a verifier without extra information disclosure. zk-SNARKs stand out for their non-interactive nature, eliminating the need for continuous communication typical in traditional proofs. Their succinct proofs, small and fast to verify, remain efficient regardless of the complexity of the proven statement, enhancing their practicality in diverse applications.

At first, the zk-SNARKs scheme involves converting computational statements (like the execution of a program) into a

---

Algorithm 1 UCB1

1: **Initialization:** Play each arm once
2: **Loop:** Play arm j that maximizes $\bar{x}_j + 2\sqrt{\frac{\ln n}{n_j}}$, where $\bar{x}_j$ is the average reward obtained from arm j, n_j is the number of times arm j has been played so far, and n is the overall number of plays done so far.

---

mathematical form (QAP) that can be efficiently proven and verified, as shown in Figure. 1. Next, the workflow of zk-SNARKs mainly involves three stages: setup, proof, and verification, as follows:

1. **Setup Phase:** zk-SNARKs scheme employs the setup algorithm to generate a common reference string (crs) including the evaluation key and verification key, which is used for proof generation and proof verification.

2. **Proof Generation:** the prover employs the prove algorithm $Prove(\cdot)$ computes proof π using the secret input w (also named witness), the statement ϕ, and the crs. I.e., $\pi \leftarrow Prove(crs, \phi, w)$, where the witness comprises private input essential for specific computations, such as input data and parameters for an ML model. It also encompasses all intermediate computational steps and results in converting high-level programs to QAP.

3. **Proof Verification:** the verifier uses the verify algorithm $Verify(\cdot)$ to check the proof π against the crs to confirm the statement ϕ's truth without additional interaction. I.e., $0/1 \leftarrow Verify(crs, \phi, \pi)$, where "1" and "0" denote verification successful and failed, respectively.

The zk-SNARK scheme can guarantee four properties, which are:

1. **Completeness:** given a true statement ϕ and a proof π, both from the prover, an argument is complete if the probability of successful verification is 1:

$$Pr[Verify(crs, \phi, \pi) = 1] = 1 \qquad (1)$$

2. **Knowledge Soundness:** given a statement ϕ from a Probabilistic Polynomial-Time (PPT) adversary $\mathcal{A}$, who can forge the witness w to generate a proof π. An argument is knowledge soundness if the probability that π generated by $\mathcal{A}$ is successfully verified is almost 0:

$$Pr[Verify(crs, \phi, \pi) = 1 | (\phi, \pi, w) \leftarrow \mathcal{A}] \approx 0 \quad (2)$$

3. **Succinctness:** an argument is succinct if the size of the proof satisfies

$$|\pi| \leq poly(k) poly \log(|\phi| + |w|) \qquad (3)$$

where k refers to the security parameter, which dictates the system's security level and determines key sizes and cryptographic algorithm complexity.

4. **Zero-knowledge:** given a simulation trapdoor td for generating a simulated witness w, a simulator $Sim(\cdot)$ for creating a simulated proof that can pass the verification process without the true witness. Facing a malicious verifier $\mathcal{A}$, an argument is zero-knowledge if $\mathcal{A}$ cannot

discern whether the prover or the simulator generated the proof, i.e.,

$$Pr[\mathcal{A}(crs, \phi, \pi) = 1 | \pi \leftarrow Prove(crs, \phi, w)] \\ = Pr[\mathcal{A}(crs, \phi, \pi) = 1 | \pi \leftarrow Sim(crs, \phi, td)] \quad (4)$$

This indistinguishability denotes that the prover does not leak any information other than the truth of the statement, ensuring the essential security and privacy features of the zk-SNARKs scheme.

From the above, zk-SNARK facilitates a secure, private, and efficient verification process in computing environments, making it invaluable in environments where data confidentiality and integrity are critical.

3 zkUCB

This section explores the integration of zk-SNARKs into the UCB algorithm. For efficiency, we utilize Zokrates, a toolkit providing both a circuit compiler and a proof system. This allows us to create a corresponding program code straightforwardly, which compiles the UCB algorithm into R1CS and uses Zokrates to generate CRS and zero-knowledge proofs. Next, we will first delve into how we tackled the implementation challenges of zkUCB. This will be followed by an overview of the zkUCB workflow, offering insights into its operational dynamics.

3.1 Overcoming Challenges in Integrating zkUCB

In our research, we tackled the inherent challenges in implementing zkUCB by employing strategic solutions as follows.

Pseudo-random Number Generator
One of the fundamental challenges in integrating the UCB algorithm with zk-SNARKs arises from the need for randomness in MAB problems. Particularly, the UCB algorithm often requires random variables to decide between arms with identical maximum expected rewards. Moreover, in reinforcement learning scenarios, employing random variables with specific probability distributions is crucial for modeling uncertainty, optimizing policies, and facilitating adaptive decision-making in stochastic environments. However, zk-SNARKs demand determinism in the compiled programs, presenting a significant hurdle in blending reinforcement learning with zero-knowledge proofs. To overcome this, we incorporated a statistical pseudorandom number generator. This inclusion allows for the deterministic encoding of the UCB algorithm under the MAB model into a digital circuit, thereby transforming a fundamentally stochastic system into a deterministic one.

In the field of stochastic processes, the generation of discrete random variables conforming to a uniform distribution can be achieved by inputting a seed parameter into a linear congruential generator (LCG) denoted as F. The seed is a parameter determined by the user. In the execution of F, there is also a second input parameter u, which is the upper bound of its output. Consequently, the sequence of random numbers $\{X_1, ..., X_n\}$ with length n is generated by $X_i = F(seed, u), 1 \leq i \leq n$. Every X_i belongs to the uniform distribution with mean u and range of $(0, 2u)$.

Algorithm 2 zkUCB

1: **Initialization:** Play each arm once
2: **Loop:** Play arm j that maximizes $q\bar{x}_j + 2q \left\lfloor \sqrt{\frac{\lfloor \ln n \rfloor}{n_j}} \right\rfloor$, where $\bar{x}_j$ represents the average reward obtained from arm j, n_j is the number of how many times the arm j has been played to date, n denotes the overall number of plays done so far, and $\lfloor \cdot \rfloor$ indicates rounding down to the nearest integer.

Non-polynomial Functions
In the context of zk-SNARK, all programs must be performed by polynomial operations in finite fields. A critical aspect of zkUCB involves converting non-polynomial operations in the UCB algorithm, such as logarithms and non-integer powers, into polynomial forms. Our solution for handling logarithmic functions is the implementation of piecewise linear approximation by focusing on discrete intervals of positive integers. After performing the approximation, and rounding down results to the nearest integer, we achieved a practical and statistically robust approximation. Similarly, for approximating non-integer powers, particularly square roots as required in the UCB algorithm, we utilized Newton's method, fixed the number of iterations to 20, and rounded down results to the nearest integer. These strategies allow us to approximate these functions using polynomial methods, ensuring compatibility with the computational constraints of zk-SNARKs while maintaining uniform time and space complexity.

Quantization for Floating Point Numbers
The UCB algorithm predominantly deals with real numbers, which inherently exist in a continuous space. Conversely, zk-SNARKs operate within finite fields, necessitating discrete computations. To bridge this gap, we employed a quantization process that maps the floating point numbers to positive integers. This process involves multiplying each element by a predetermined scaling factor (q). However, as depicted in Alg. 2, directly applying proportional scaling to n and N_j is not viable. Such scaling significantly alters the value of n away from zero, leading to a negligible derivative of $\ln n$, which in turn disrupts the crucial balance between exploration and exploitation. This imbalance is particularly evident in the learning agent's reduced inclination towards exploratory actions initially. Moreover, if proportional scaling is straightforwardly applied to $\ln n$ and N_j, the results remain essentially identical to those in the unscaled scenario. Yet, the scaled $\bar{x}_j$ becomes disproportionately larger than the other unscaled components, once again disturbing the exploration-exploitation equilibrium. By applying the scaling parameter uniformly to both the prior and subsequent components, we streamline the computations and attain more favorable results.

In our study, we assume that the reward from each arm follows a normal distribution. To make zkUCB satisfy this requirement, we use sliding window sampling to sample data from a uniform distribution to simulate a normal distribution based on the principle of the Central Limit Theorem. In zkUCB, we set the mean and the range to be q and $(0, 2q)$,

respectively for the uniform distribution. We repeat this sampling process for 20 times and generate the average denoted as $(\bar{x})$. Obviously, $(\bar{x})$ belongs to a normal distribution with mean q and range $(0, 2q)$. Since the mean of normal distribution satisfies the additivity, this $(\bar{x})$ can be used to calculate the actual reward in zkUCB.

3.2 The Workflow of zkUCB

This subsection shows how the zkUCB scheme works.

- **Setup** $(crs, td) \leftarrow Setup(zkUCB)$

  In this stage, the setup algorithm generates a common reference string crs and a simulation trapdoor td. Where crs includes the evaluation key and the verification key, td is used to generate a simulated witness when checking the zero-knowledge properties of zkUCB to further generate simulated proofs π.

- **Proof Generation** $\pi \leftarrow Prove(crs, \phi, w)$

  In the proof generation stage, the prove algorithm computes proof π using the secret input w (also named witness), the statement ϕ, and the crs. Where the statement ϕ is constructed using several elements, including the operations of a pseudo-random number generator determined by a seed, the expected rewards of each arm, the reward received after playing arm during each round, and the implementation of the UCB algorithm under the MAB model. The witness w serves as private input, including the input data, parameters, all intermediate computational steps, and results in converting high-level programs to QAP.

- **Proof Verification** $0/1 \leftarrow Verify(crs, \phi, \pi)$

  In the proof verification phase, the verifier uses the verify algorithm to assess the proof π against the crs, thereby ascertaining the truth of the statement ϕ. Where "1" signifies successful verification, and "0" denotes failure.

Theorem 1. *The above zkUCB scheme satisfies a non-interactive zero-knowledge argument of knowledge with completeness and perfect zero-knowledge. It has computational knowledge soundness against a Probabilistic Polynomial-Time adversary.*

Proof. We prove this theorem by analyzing how the zkUCB scheme guarantees the properties mentioned in the theorem, as shown below.

Completeness: for the true statement ϕ and the proof π, both from the prover, the zkUCB is complete if the probability of successful verification is 1:

$$Pr[Verify(crs, \phi, \pi) = 1] = 1 \quad (5)$$

Knowledge Soundness: given a statement ϕ^* from a Probabilistic Polynomial-Time (PPT) adversary $\mathcal{A}$, who can forge the witness w^* to generate a proof π^*. the zkUCB has computational knowledge soundness if the probability that ϕ^* is successfully verified is close to 0 and negligible:

$$Pr[Verify(crs, \phi^*, \pi^*) = 1 | (\phi^*, \pi^*, w) \leftarrow \mathcal{A}] \approx 0 \quad (6)$$

Zero-knowledge: given a simulation trapdoor td for generating a simulated witness w', a simulator $Sim(\cdot)$ for creating a simulated proof π' that can pass the verification process without the true witness w. Facing a malicious verifier $\mathcal{A}$, the zkUCB is zero-knowledge if the malicious verifier $\mathcal{A}$ cannot discern whether the prover or the simulator generated the proof, i.e.,

$$Pr[\mathcal{A}(crs, \phi, \pi) = 1 | \pi \leftarrow Prove(crs, \phi, w)]$$
$$= Pr[\mathcal{A}(crs, \phi, \pi') = 1 | \pi' \leftarrow Sim(crs, \phi, td)] \quad (7)$$

This indistinguishability characteristic ensures that the prover reveals no information beyond the veracity of the statement ϕ. Consequently, neither a well-behaved verifier nor a malicious verifier can glean any additional knowledge from the statement ϕ. This ensures the inherent security and privacy properties integral to the zkUCB scheme. $\square$

In summary, zkUCB fulfills the criteria of completeness, perfect zero-knowledge, and computational knowledge soundness, against probabilistic polynomial-time adversaries. This makes zkUCB stand as a highly secure, private, and efficient framework, adept at facilitating transparent verification processes in computing environments. It is especially promising in scenarios where data confidentiality and integrity are of paramount importance.

4 Experimental Evaluation and Analysis

4.1 Experimental Objectives

In this section, we conduct a comprehensive assessment of the effectiveness of zkUCB. Our evaluation focuses on several key aspects:

1. *Reward Comparison:* We analyze the performance disparities between the standard UCB algorithm and zkUCB, with a particular emphasis on the average reward generated during each step across 100 iterations over 200 steps.

2. *Time Analysis:* We delve into various time-related metrics for zkUCB, including setup time, compile time (the duration required for converting high-level programs to R1CS), compute witness time, generate proof time, and verify time. These measurements are conducted over 200 steps, with results recorded at intervals of 10 steps.

3. *Components Size:* We examine the sizes of essential components for zkUCB, including the proving key size, proof size, verifying key size, and witness size. These metrics are assessed over 200 steps, with measurements recorded at 10-step intervals.

4.2 Experimental Setup

Our experiment leverages ZoKrates, as it enables the selection of a proof system, elliptic curves for pairing, and zk-SNARKs backend at our discretion. In this context, we have opted for the ALT_BN128 elliptic curve. The Groth-16 protocol is chosen as the proof scheme. Additionally, the Ark backend is selected. The rationale for employing this specific configuration is attributed to its well-structured design,
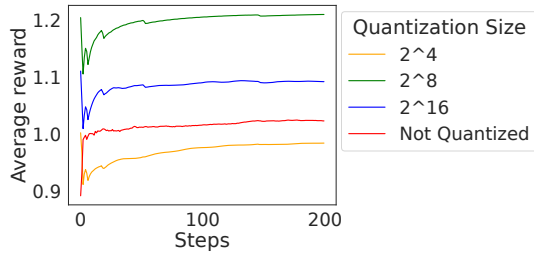
Figure 3: Average reward generated during each round by UCB and zkUCB over 100 iterations

making it among the more favored options in current practical implementations. Specifically, we operate the following two settings for different experimental purposes.

Setting I
This experiment is under a three-armed MAB model and runs a UCB algorithm with 200 steps, where each arm has initially expected rewards set at 0.9, 1.0, and 1.1, respectively. We set three different levels of quantization (2^4, 2^8, 2^{16}) and calculate the average reward across 100 iterations. To adapt to zk-SNARKs, we perform quantization operations on the UCB algorithm. To compare the expected reward of the quantized zkUCB with the non-quantified version of the standard UCB, we multiply the reward generated by the zkUCB by the reciprocal of the quantified scaling ratio, so that the reward generated by zkUCB and the reward generated by ordinary UCB belong to a same distribution, keeping the baseline consistent. The theoretical basis for this is the additivity of expectations derived from random variables.

Setting II
Maintaining the same foundational parameters as in Setting I, this experiment executes a UCB algorithm with various steps (ranging from 20 to 200, with increments of 10 steps per interval) for 100 iterations to average the results.

4.3 Results and Analysis

Analysis of Reward
Figure. 3 compares the rewards generated by the standard UCB algorithm and zkUCB in each round. It demonstrates that zkUCB with smaller quantization bits slightly underperforms compared to the standard UCB. In contrast, zkUCB configured with larger quantization bits generally exceeds the performance of the standard UCB. Notably, the optimal performance is observed with zkUCB at a medium quantization level. This improvement is because appropriate quantization bits can effectively reduce information entropy in the decision-making process, enhancing rewards. However, overly high quantization may reduce information entropy excessively, causing a loss of vital information, which in turn leads to performance degradation of zkUCB. Consequently, a well-calibrated quantization level that optimally reduces information entropy is key to maximizing zkUCB's performance.

Time Efficiency Across Stages in zkUCB
Figure. 4 and Figure. 5 display the average execution times for various stages of zkUCB with different quantization bits
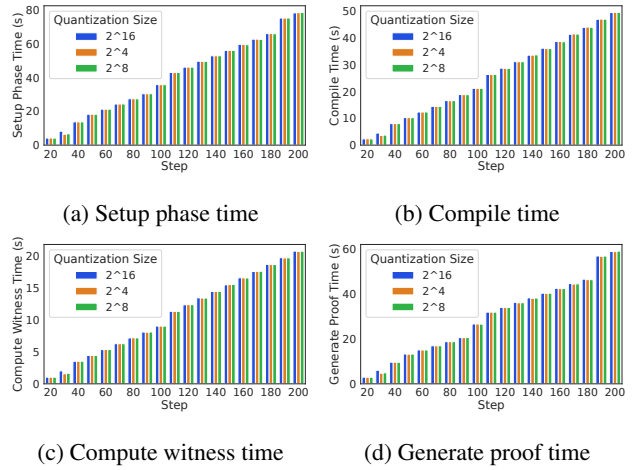

(a) Setup phase time

(b) Compile time

(c) Compute witness time

(d) Generate proof time

Figure 4: The average time during various phases
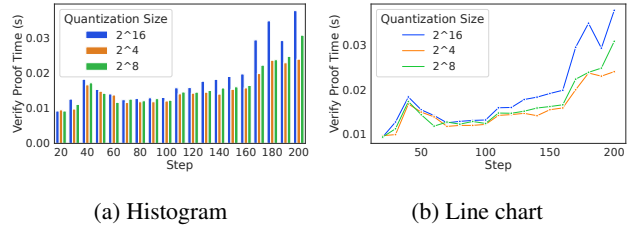

(a) Histogram

(b) Line chart

Figure 5: The average time for verifying proof

over a range of steps. Figure. 4 highlights consistent performance in setup, compilation, witness calculation, and proof generation times across various quantization sizes, showing these times are influenced more by the number of algorithm steps than by quantization bits. Figure. 5, however, reveals a notable difference in verification times for zkUCB with different quantization bits; larger quantization numbers result in longer verification times, which also scale with the number of algorithm steps. These findings demonstrate zkUCB's promising application potential and scalability, especially considering the minimal verification times, which are within milliseconds.

Components Size in zkUCB

Figure. 6 displays the sizes of various components of zkUCB with different quantization bits across a sequence of steps, including witness size, proving (evaluation) key size, verifying (verification) key size, and proof size. The results reveal that the quantization bits have a negligible impact on the sizes of these zkUCB components. Instead, the size of these components is more influenced by the number of algorithm steps, showing a proportional relationship to these steps. Notably, even within 200 steps, the proof size remains relatively modest, only in the tens of kilobytes range, highlighting zkUCB's considerable potential for broad application and scalability.
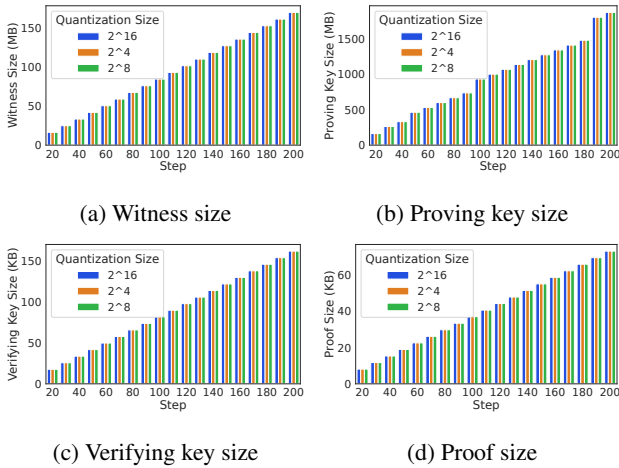
(a) Witness size

(b) Proving key size

(c) Verifying key size

(d) Proof size

Figure 6: The average time during various phases

5 Related Work

Here we first discuss the state-of-art for the verifiable computation then we move to discuss its implementation within machine learning. Finally, we discuss the shortcomings and how our approach could assist.

Verifiable Computation. Verifiable computation is a key research area essential for validating the correctness of outsourced computations, particularly in cloud services. It focuses on verifying third-party computations without needing to re-execute them fully. This field encompasses a blend of cryptographic and non-cryptographic approaches to ensure computational integrity and data authenticity. Cryptographic methods include Zero-Knowledge Proofs [Yang and Wang, 2022; Mouris and Tsoutsos, 2021; Wahby *et al.*, 2018], Interactive Proofs (IP) [Baelde *et al.*, 2021], and Homomorphic Encryption (HE) [Kim *et al.*, 2023], which offer robust security guarantees. In contrast, non-cryptographic methods, notably Authenticated Data Structures (ADS) [Sun *et al.*, 2020], concentrate on maintaining data integrity. Three primary streams dominate this field: ADS, which ensures data authenticity; IP, which facilitates interactive validation; and zk-SNARKs [Groth, 2016; Wahby *et al.*, 2018], renowned for efficient, non-interactive proof generation. These diverse methods highlight the field's complexity and richness, underpinning its significance in areas ranging from cloud computing to privacy-preserving protocols. This comprehensive approach to verifying computational processes without complete re-execution sets the foundation for advanced research and practical applications in computational integrity.

Verifiable Machine Learning. The integration of verifiable computation with machine learning (ML) stands as a transformative advancement, poised to redefine data security and integrity in AI-driven applications. This convergence is not only enhancing the trustworthiness of ML models but is also unlocking new avenues for secure and private data analysis, particularly in sensitive sectors like healthcare, finance, and government. Traditional ML models often operate as black boxes, with limited ability to verify the internal pro-

cesses and outputs. The advent of verifiable machine learning aims to bring transparency and reliability into this domain. Homomorphic encryption (HE) [Li *et al.*, 2020] and garbled circuits (GC) [Jawurek *et al.*, 2013] are prevalent methods in this sphere, enabling secure computation of encrypted data. While these methods offer robust privacy guarantees, they often come with computational overheads. Another trend is the use of Zero-Knowledge Proofs in ML [Weng *et al.*, 2021; Xing *et al.*, 2023], which allows the validation of model predictions without revealing the underlying data or model parameters. This is particularly useful in scenarios where data confidentiality is paramount. Current research integrating ZKP has primarily concentrated on specific algorithms, including CNN [Liu *et al.*, 2021; Lee *et al.*, 2024], DNN [Ghodsi *et al.*, 2017], linear regression, logistic regression, neural networks, support vector machines, K-means [Zhao *et al.*, 2021], and decision trees [Zhang *et al.*, 2020].

Shortcomings: Despite the promising potential of ZKP for enhancing privacy, their application in Reinforcement Learning (RL) is underexplored. This is mainly due to RL's complexities and high computational demands in dynamic environments. We focus on the MAB problem, a prevalent model in critical domains like healthcare and recommendation systems, where maintaining data integrity is of paramount importance. We aim to apply ZKP's privacy-preserving capabilities to the UCB-based MAB framework, achieving a trustworthy and confidential decision process.

6 Conclusion

This study presents a significant advancement in the integration of reinforcement learning with data privacy through the development of the zkUCB algorithm. By integrating zk-SNARKs within the UCB algorithm, zkUCB ensures the confidentiality and security of sensitive data and parameters, while facilitating a transparent decision-making process. Our comprehensive experimental analysis confirms the superiority of zkUCB over the standard UCB algorithm, particularly in terms of reward optimization and operational efficiency. This improvement is because appropriate quantization bits can effectively reduce information entropy in the decision-making process, enhancing rewards. In addition, the scalability and practicality of zkUCB are demonstrated through its linear proof size and verification times, underlining its potential applicability in a wide range of real-world scenarios.

Acknowledgements

References

[Afsar *et al.*, 2022] M Mehdi Afsar, Trafford Crump, and Behrouz Far. Reinforcement learning based recommender systems: A survey. *ACM Computing Surveys*, 55(7):1–38, 2022.

[Ameko *et al.*, 2020] Mawulolo K Ameko, Miranda L Beltzer, Lihua Cai, Mehdi Boukhechba, Bethany A Teachman, and Laura E Barnes. Offline contextual multi-armed bandits for mobile health interventions: A case study on emotion regulation. In *Proceedings of the 14th ACM Conference on Recommender Systems*, pages 249–258, 2020.

[Aziz *et al.*, 2021] Maryam Aziz, Emilie Kaufmann, and Marie-Karelle Riviere. On multi-armed bandit designs for dose-finding clinical trials. *The Journal of Machine Learning Research*, 22(1):686–723, 2021.

[Azize and Basu, 2022] Achraf Azize and Debabrota Basu. When privacy meets partial information: A refined analysis of differentially private bandits. *Advances in Neural Information Processing Systems*, 35:32199–32210, 2022.

[Baelde *et al.*, 2021] David Baelde, Stéphanie Delaune, Charlie Jacomme, Adrien Koutsos, and Solène Moreau. An interactive prover for protocol verification in the computational model. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 537–554. IEEE, 2021.

[Chen *et al.*, 2022] Thomas Chen, Hui Lu, Teeramet Kunpittaya, and Alan Luo. A review of zk-snarks. *arXiv preprint arXiv:2202.06877*, 2022.

[Gao *et al.*, 2022] Guoju Gao, Sijie Huang, He Huang, Mingjun Xiao, Jie Wu, Yu-E Sun, and Sheng Zhang. Combination of auction theory and multi-armed bandits: Model, algorithm, and application. *IEEE Transactions on Mobile Computing*, 2022.

[Ghodsi *et al.*, 2017] Zahra Ghodsi, Tianyu Gu, and Siddharth Garg. Safetynets: Verifiable execution of deep neural networks on an untrusted cloud. *Advances in Neural Information Processing Systems*, 30, 2017.

[Groth, 2016] Jens Groth. On the size of pairing-based non-interactive arguments. In *Advances in Cryptology–EUROCRYPT 2016: 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II 35*, pages 305–326. Springer, 2016.

[Jawurek *et al.*, 2013] Marek Jawurek, Florian Kerschbaum, and Claudio Orlandi. Zero-knowledge using garbled circuits: how to prove non-algebraic statements efficiently. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 955–966, 2013.

[Kim *et al.*, 2023] Taechan Kim, Hyesun Kwak, Dongwon Lee, Jinyeong Seo, and Yongsoo Song. Asymptotically faster multi-key homomorphic encryption from homomorphic gadget decomposition. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, pages 726–740, 2023.

[Kiran *et al.*, 2021] B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A Al Sallab, Senthil Yogamani, and Patrick Pérez. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 23(6):4909–4926, 2021.

[Lee *et al.*, 2024] Seunghwa Lee, Hankyung Ko, Jihye Kim, and Hyunok Oh. vcnn: Verifiable convolutional neural network based on zk-snarks. *IEEE Transactions on Dependable and Secure Computing*, 2024.

[Li *et al.*, 2020] Jing Li, Xiaohui Kuang, Shujie Lin, Xu Ma, and Yi Tang. Privacy preservation for machine learning training and classification based on homomorphic encryption schemes. *Information Sciences*, 526:166–179, 2020.

[Liu *et al.*, 2021] Tianyi Liu, Xiang Xie, and Yupeng Zhang. Zkcnn: Zero knowledge proofs for convolutional neural network predictions and accuracy. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 2968–2985, 2021.

[Mouris and Tsoutsos, 2021] Dimitris Mouris and Nektarios Georgios Tsoutsos. Zilch: A framework for deploying transparent zero-knowledge proofs. *IEEE Transactions on Information Forensics and Security*, 16:3269–3284, 2021.

[Patil *et al.*, 2021] Vishakha Patil, Ganesh Ghalme, Vineet Nair, and Yadati Narahari. Achieving fairness in the stochastic multi-armed bandit problem. *The Journal of Machine Learning Research*, 22(1):7885–7915, 2021.

[Sun *et al.*, 2020] Yi Sun, Qian Liu, Xingyuan Chen, and Xuehui Du. An adaptive authenticated data structure with privacy-preserving for big data stream in cloud. *IEEE Transactions on Information Forensics and Security*, 15:3295–3310, 2020.

[Wahby *et al.*, 2018] Riad S Wahby, Ioanna Tzialla, Abhi Shelat, Justin Thaler, and Michael Walfish. Doubly-efficient zksnarks without trusted setup. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 926–943. IEEE, 2018.

[Weng *et al.*, 2021] Chenkai Weng, Kang Yang, Xiang Xie, Jonathan Katz, and Xiao Wang. Mystique: Efficient conversions for {Zero-Knowledge} proofs with applications to machine learning. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 501–518, 2021.

[Xing *et al.*, 2023] Zhibo Xing, Zijian Zhang, Jiamou Liu, Ziang Zhang, Meng Li, Liehuang Zhu, and Giovanni Russello. Zero-knowledge proof meets machine learning in verifiability: A survey. *arXiv preprint arXiv:2310.14848*, 2023.

[Yang and Wang, 2022] Kang Yang and Xiao Wang. Non-interactive zero-knowledge proofs to multiple verifiers. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 517–546. Springer, 2022.

[Yu *et al.*, 2021] Chao Yu, Jiming Liu, Shamim Nemati, and Guosheng Yin. Reinforcement learning in healthcare: A survey. *ACM Computing Surveys (CSUR)*, 55(1):1–36, 2021.

[Zhang *et al.*, 2020] Jiaheng Zhang, Zhiyong Fang, Yupeng Zhang, and Dawn Song. Zero knowledge proofs for decision tree predictions and accuracy. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 2039–2053, 2020.

[Zhao *et al.*, 2020] Hui Zhao, Mingjun Xiao, Jie Wu, Yun Xu, He Huang, and Sheng Zhang. Differentially private unknown worker recruitment for mobile crowdsensing using multi-armed bandits. *IEEE Transactions on Mobile Computing*, 20(9):2779–2794, 2020.

[Zhao *et al.*, 2021] Lingchen Zhao, Qian Wang, Cong Wang, Qi Li, Chao Shen, and Bo Feng. Veriml: Enabling integrity assurances and fair payments for machine learning as a service. *IEEE Transactions on Parallel and Distributed Systems*, 32(10):2524–2540, 2021.