A Reputation System for Provably-robust Decision-making in IoT Blockchain Networks

Charles Rawlins, *Member, IEEE,* S. Jagannathan, *Fellow IEEE,* and V. Sriram Siddhardh Nadendla, *Member IEEE*

***Abstract*—Blockchain systems have been successful in discerning truthful information from inter-agent interaction amidst possible attackers or conflicts, which is crucial for the completion of non-trivial tasks in distributed networking. However, the state-of-the-art blockchain protocols are limited to resource-rich applications where reliably-connected nodes within the network are equipped with significant computing power to run lottery-based Proof-of-Work (PoW) consensus. The purpose of this work is to address these challenges for implementation in a severely resource-constrained distributed network with Internet-of-Things (IoT) devices. The contribution of this work is a novel lightweight alternative, called Weight-based Reputation (WBR) scheme, to classify new transactions via modeling blockchain decisions as a distributed machine-learning task. WBR identifies network nodes that are willing to cooperate towards securing ground-truth, showing robustness to adversarial subnetworks that are greater than 50% and reducing collaboration error by 50% compared to other similar schemes. This two-step approach of reputation plus transaction classification for generating blockchain data is treated as a novel method of preventing fraud and double-spending attacks in blockchain networks. To capture adversary influence, a Bayesian game is formulated and implemented to show superior performance to the state-of-the-art along with resource consumption metrics.**

***Index Terms*—Blockchain, Secure Learning, Reputation Systems, Internet of Things**

I. Introduction

BLOCKCHAIN is used to secure arbitrary data in the form of transactions to prevent censorship and conflicts without the trust of a third-party. This approach can be applied to a variety of networking scenarios, including Internet of Things (IoT) networks [1]. Although blockchain provides transparency to an end-user for network activity, a challenge with incorporating blockchain directly on limited nodes like IoT devices [1] is centralization with offloading to distant full-powered servers[2]. While this architecture is inherent to IoT, removing said schemes removes compatibility with blockchain security features like computational-lottery data verification and introduces processing bottlenecks [3]. Traditional blockchain systems realize their security guarantees from their most crucial feature called *distributed consensus*. One of the most well-known blockchain consensus schemes, Proof-of-Work (PoW) [4], introduces a computational race to mine transaction blocks, that uses a majority-rule during verification. Practical use of PoW is secure against small attackers, though not without design criticism of proposed security measures and energy usage, as discussed in [5]–[7]. This ruling prevents recording of false transaction information, commonly introduced through the double-spend attack [8] where a finite token resource is used twice with two different accounts. Double-spending can also apply to data access and generation, such as multiple honest IoT devices having read/write access to cached information with actions stored on the blockchain [9].

A fundamental problem in traditional blockchain consensus design is the passive reaction to conflict resolution. Formulated as a repeated Bernoulli trial [4], the assumed legal majority will eventually outpace an attacking minority through repeated confirmation of new data. However, if double-spend attacks occur rarely, most of the lottery-based computation is wasted on confirming already-legal data. A question this work proposes is: *Can observing past behavior of participating nodes in blockchain networks be used to avoid future conflicts in ledger data?* Methods for proactively avoiding ledger computation based on past observations, such as using machine-prediction, are absent in the related blockchain literature. Blockchain has been explored for recording learning tasks as a backbone for ground-truth [10], but not as the main deciding factor for recorded data.

Lightweight consensus alternatives have been proposed over the past decade in the blockchain-related literature [11]. Reputation quantifies trust between contributing parties exchanging information or assets in social environments [12], [13] in discrete levels, as it can be used for both logging [14], and consensus calculations [15]–[17]. However, there is little mathematical justification for effectively calculating individual reputation values, since it is a non-trivial exercise to provide theoretical guarantees using Partially-observable Markov Decision Process (POMDP) [13] or game theoretical frameworks [18].

Steps towards provable reputation for generic finance networks are shown in [19], but not in a fully-distributed setting. The machine-learning community designs provable-robust techniques for improving robustness in distributed reinforcement learning. While there exists a variety of complex methods to repel adversarial attacks against reinforcement learning (e.g. loss regularization, adversarial training [20]), one of the simplest and proven-effective methods is smoothing to

Charles Rawlins and S. Jagannathan are with the Department of Electrical and Computer Engineering, Missouri University of Science and Technology Rolla, MO, USA. (email: crawlins@mst.edu).

V. Sriram Siddhardh Nadendla is with the Department of Computer Science, Missouri University of Science and Technology Rolla, MO, USA.

Manuscript received X; revised X.

improve generalization [21], [22]. Further, the effort in [23] explored adding additional terms to model-free learning to create a conservative estimate of the optimal learning values and penalizing out of distribution actions. On the contrary, the work in [24] smoothed the agent's interpretation of the environment with adversarial perturbations in generations of state values. A major challenge with designing these systems is comprehensive defense against a variety of attacks and networking scenarios, which are absent in the literature.

Inspired by provable reputation schemes in robust Q-learning[1] techniques proposed in distributed reinforcement learning literature, this work presents a novel approach called Weight-based Reputation (WBR) for generating provable reputation in a distributed IoT network that generates and classifies blockchain transactions. WBR learns reputation of network nodes using distributed Q-learning (e.g. Deep Q Network (DQN) [25], Double DQN (DDQN) [26], Conservative Q-learning (CQL) [27], and Conservative Q-Improvement (CQI) [28]) and creates a lightweight lookup table that IoT networks can update weights dynamically to prevent training-specific attacks, as in [29]. Similar approaches that prevent training on poisoned data include Multi-Krum [30] and Bridge-T [31], which rely on fixed statistical measures that don't consider the underlying data of individual samples. Unlike the WBR approach, both Multi-Krum and Bridge-T suffer from majority attacks where the attacker poisons data at majority of the network's nodes. Moreover, WBR also prevents spam attacks from occurring through network synchronization.

To the best of our knowledge, this is the first work that offers provable machine-learning security for permissionless blockchain systems. Therefore, the first contribution is a model proposing a Bayesian game to detect adversarial disturbances to a collective group task. The second contribution is a robust value function for a meta Q-learning observer to make active decisions about what data to accept in a distributed setting. In addition to theoretical guarantees, simulations in ideal and realistic settings are also explored using the IOTA GoShimmer framework to validate the performance of WBR.

The remainder of the paper is organized as follows. Section II introduces relevant background such as proof of work, IOTA, and potential threats. Section III provides the game theoretic modeling framework, proposed methodology and proof for the WBR scheme. Results in Section IV simulate the scheme under a variety of scenarios in idealistic and realistic setups before presenting conclusions in Section V.

II. Background and Related Works

Computationally-intense decision-making is inherent to traditional blockchain protocol design, for both classic approaches like Bitcoin's PoW and IoT-focused protocols like IOTA. With a data-focused approach, strong security is necessary to avoid introducing more vulnerabilities with data poisoning with distributed reputation. Poisoning-prevention is explored using a Bayesian game in the threat model.

[1]Q-learning is a model-free reinforcement learning scheme that associates a state-action pair (S, A) with the reward function R, to generate a Q-value $Q(S, A)$.

A. Proof of Work and IOTA

Decision-making with PoW in blockchain networks when deconstructed is a simple approach following majority-rule. Each node competes to solve a linear search using $\Omega(N)$ steps [32] with $N = 2^b$ with b input bits. The legal p and malicious $q = (1 - p)$ subnets each work to verify new a new block following a cryptographic-lottery scheme where the probability of winning is reduced to a Bernoulli trial. Each probability is reliant on the amount of computational power of each subnet $p \propto \mu_1$ and $q \propto \mu_2$. This considers the simplest case where both block mining tasks are available, though newer models consider more complex situations [33]. A recent Distributed Ledger Technology (DLT) protocol, IOTA, is designed specifically for use on IoT networks following a similar regime to Bitcoin in its setup for distributed decision-making, also following a majority-rule computation. While the protocol is currently exploring alternatives to PoW [34], the original design also used a lightweight form of PoW to create a similar repeated Bernoulli trial setup.

This work proposes an alternative to the Bernoulli probability in order to remove the computational race introduced with PoW using machine-learning classification. Assuming the distributed nodes each participate in a distributed Stochastic Gradient Descent (SGD) learning task analyzing the transaction data generated from other nodes to produce a percent confidence P_C, creating $\mu_1 = P_C, \mu_2 = (1 - P_C)$. It is assumed calculations are under perfect network and data security to prevent manipulation to falter P_C, along with prevention of data-poisoning/evasion attacks [35] and Sybil spam attacks. Following an approach in the literature [36], the honest group nodes will converge to the ϵ-solution with rate $\mathcal{O}(\frac{1}{K} + \frac{1}{\sqrt{nK}})$ for n nodes and K epochs [36], executing $O(\frac{n}{\epsilon} + \frac{1}{\epsilon^2})$ steps using *strength in numbers*. The intended scenario for this would be with limited IoT devices needing to minimize computation when disconnected from the network as shown in Figure 1. So far, perfect security was assumed, but adversaries would seek to obfuscate capture by stalling or stopping proper distributed training and classification, which this approach addresses.

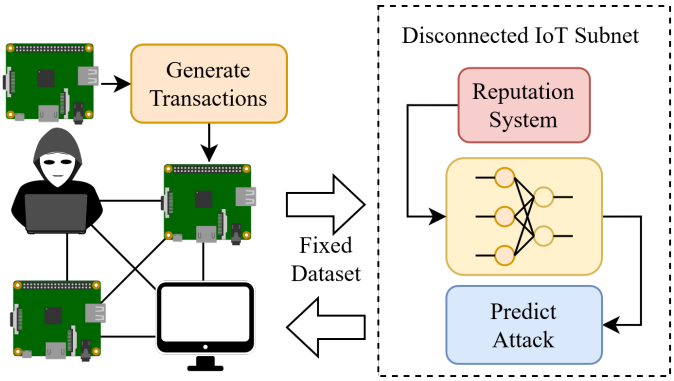


Fig. 1: Offline blockchain reputation example.

B. Threat Modeling and Bayesian Games

For this work, two different data-poisoning attacks are considered that the adversary is capable of performing on the network and distributed parameters shared for weight training. The adversary is assumed to not be capable of performing a man-in-the-middle attacks to modify data between nodes which use secure cryptography for authentication.

The first attack is referred to the 'Uniform' attack where the adversary adds unique uniform noise to individual distributed trainable weight values $W_k \in \mathcal{W}_k$, with $W_k = W_k + u \cdot U(0,1)$ for $k = 0, 1, ..., d$, which adds noise that is independant from the weight variable before distribution. The second attack employs a more intelligent disturbance from the literature using the Maximal Action Distance (MAD) attack that maximizes dependant noise between the current adversarial parameters $\hat{\mathcal{W}}$ and the initial parameters $\mathcal{W}_0$ (see Algorithm 3 in Appendix D in [21]). The added noise maximizes the KL-divergence from the weights in the previous state before sending $\mathcal{W}_k$. With the loss noted as $L_{MAD}(\mathcal{W}_k) = -D_{KL}(\mathcal{W}_0, \mathcal{W}_k)$, the weights are updated before transmission with

$$W_k = W_k + \nabla L_{MAD}(\mathcal{W}_k(t-1)) + \sqrt{\frac{2}{\beta\eta}}\epsilon,$$

where learning rate η, β and ϵ are parameters.

To defend against these two attacks, Bayesian game theoretic formulation will be applied. Bayesian games involve players who don't have complete information about the game being played. Each player takes actions to receive a quantifiable payoff in the form of utility. The players then update their beliefs about the types and behavior of other nodes based on the observed statistical patterns. Since honest nodes H are unaware of the identities or data manipulation strategies of adversarial nodes B, they must determine the behavior of adversaries based on what they observe. Each interaction between an individual honest and attacker node is formulated as a repeated two-player game with the tuple $(N, \mathcal{A}, \Theta, p, U, S)$ where the variables are:

- N nodes in the network with $\{N_i = \text{Honest node}, N_j = \text{Suspicious node}\}$
- $\mathcal{A}$ action space, where $\mathcal{A} = \{A_i = (\text{Accept, Reject}), A_j = (\text{Attack, Not Attack})\}$
- Node type $\mathbf{\Theta} = \{\Theta_H, \Theta_B\}$, where Θ_H indicates honest and Θ_B indicates malicious
- Prior p at each node developed from actions of the other nodes
- Utility $U : A \times \Theta \mapsto \mathbb{R}$ representing payoff for actions taken
- State space $S(t)$, which is defined in Section III

III. Methodology

A network set of $n = |N|$ nodes is executing both a blockchain protocol and simultaneously participating in a distributed learning task to classify transactions. It is assumed every node $N_i \in N$ is part of a fully-connected graph $\mathcal{G}$ and each node has a uniform probability of pinging k other nodes for information that will always answer. The information exchanged in an answer is a single blockchain transaction that is added to a local pool of potential transactions to append to their local blockchain, like IOTA's graph ledger system [3].

Transactions contain $D_i = (\mathscr{W}_i, \mathscr{L}_i)$ to record progress towards the global classification solution in the shared learning task. Training data and class flags are shared between each N_i that is sampled uniformly for $x \in X$. The honest subnetwork $H \in N$ solve a task using distributed SGD [36], optimizing local parameters $\mathcal{W}_i$. Loss for N_i is publicly defined $(\mathcal{W}, X) \mapsto f(\mathcal{W}, X)$ to define error towards the ideal weights

$$\mathcal{W}^* \in \arg\min_{\mathcal{W} \in \mathbb{R}^D} \mathbb{E}(f(\mathcal{W}, X)).$$

Gradients produced from loss f with ∇f that are assumed to be smooth and L-lipschitz. Upper bounds for the global population σ^2 and node sample ς^2 variance for the gradients is also defined.

Individual weight values $W_k \in \mathcal{W}_i, k = 1, ..., d$ follow a Gaussian process (approaching the distribution as $d \to \infty$ [37]). It is assumed that $W_k \sim \mathcal{N}(\mu_{W,k}, \sigma_{W,k})$. H seeks to find

$$\min_{\mathcal{W}_i : i \in H} \frac{1}{H} \sum_{i=1}^{|H|} \nabla f_i(\mathcal{W}_i, X) \tag{1}$$

without knowing the identities of the other nodes in H. As part of distributed SGD [36], each node updates $\mathcal{W}_i$ using a separate symmetric doubly stochastic scaling matrix $C \in \mathbb{R}^{n \times n}$, where $C_{ij} \in [0,1]\ \forall i, j, C_{ij} = C_{ji}$ and $\sum_j C_{ij} = 1$. $C_{ij} = 0$ notes a disconnection from N_i to N_j. Weights are aggregated from each available node (see Algorithm 1 in [36]) at time t with

$$W_i(t) = \sum_{j=1}^{n} C_{ij} W_j(t)$$

before taking a standard SGD learning step

$$W_i(t+1) = W_i(t) - \alpha \nabla f(X_i)$$

with learning rate α. It is assumed $\forall t, \alpha(t) \leq \alpha(t+1)$.

A. Bayesian Game Formulation

To model whether an adversary has perturbed a distributed weight update stored on the blockchain, the approach this work takes considers each weight update as a potentially large discrete vector for processing and detecting anomalies. This approach is similar to some non-distributed adversarial training techniques [38], but these approaches primarily use vector inputs of gradients for training on adversarial input samples, while this approach looks for perturbations.

Participating nodes with $N_i \in H$ and $N_j \in N$ always publish data $D_j = \mathcal{W}_j, \mathscr{L}_j$ containing an array of weights and loss for a mini-batch update taken in training a classifier, and receiving nodes can $\mathcal{A}_{i \cup j} = \{(\text{Accept, Reject})\}$, while $\mathcal{A}_j(\text{Attack, Not Attack})\}$. From the data published and publicly recorded, types for a node $\mathbf{\Theta} = \{\Theta_H, \Theta_B\}$ are determined along with p. U and S are defined below.

Each node has a positive probability associated with each action, known as a mixed strategy. Scenarios for an adversary where attacks consistently occur is called a pure strategy, which is explored experimentally in Section IV. Each N_i seeks to maximize using D to solve for (1), while N_j seeks to

obfuscate learning in the opposite direction. By determining the type for each N_j with Θ_j, N_i can adjust influence by controlling factors in matrix C, with $C_{ij} = 0$ for Θ_B and $C_{ij} = 1$ for Θ_H. The state space for the game contains a mixture of ancillary information to record the connectivity and identity of N_j along with the adversary-manipulated $\mathcal{W}_j$.

Using past behavior from N_j with S, N_i creates p_i. As part of this scenario, H make modifications to received data in $\mathcal{W}_j$. It is assumed N_i has threshold D_T to determine if samples $\mathcal{W}_j$ are out-of-distribution from $\mathcal{W}_i$ for anomaly detection. Unpredictable random noise is also added $\mathcal{W}_j + \mathcal{N}(0, \sigma_\epsilon)$ to remove minute changes and scenarios where attackers try to predict the update distribution of H with parameter σ_ϵ.

Referring back to the group task stated in (1), each node is assumed to have a vector of noise $\bar{Y}$ that differentiates their own learning experience and is centered around W^*. For N_i, this is represented for each weight value as $Y_{i,k} \in \bar{Y}_i$. For Θ_H, this is $Y_{H,k} = |W_k - W_k^*| + \mathcal{N}(0, \sigma_\epsilon)$, and for Θ_B this is $Y_{B,k} = |W_k - W^*k| + \mathcal{N}(0, \sigma_\epsilon) + \epsilon_i$.

For $\mathbf{\Theta}$, in order for N_i to accept D_j, each node would have to submit $\bar{Y}_j$ where all of $Y_{j,k}$ are within D_T. This is represented probabilistically as a binomial distribution for each weight value for each time the game is played T as a form of utility

$$U_j = \sum_{t=1}^{T} \mathbb{1}\{\sum_{k=1}^{d} \binom{d}{i} p_1^i (1-p_1)^{d-i} | Y_{j,k} \leq Y_{i,k}\}$$

where $p_1 = P(Y_{j,k} \leq Y_{i,k})$. If $\Theta_j = \Theta_H$, then this develops into expected utility where

$$p_2 = P(Y_{j,k} \leq Y_{i,k} - \mathcal{N}(0, \sigma_\epsilon))$$

$$U_j = \sum_{t=1}^{T} \sum_{k=1}^{d} \binom{d}{i} (p_1 p_2)^i (1 - p_1 p_2)^{d-i}$$

$$\mathbb{E}(U_j) = \sum_{t=1}^{T} d p_1 p_2. \tag{2}$$

For Θ_B a similar situation follows, where

$$p_3 = P(Y_{j,k} \leq Y_{i,k} - \mathcal{N}(0, \sigma_\epsilon) - \epsilon_j)$$

$$U_j = \sum_{t=1}^{T} \sum_{k=0}^{d} \binom{d}{i} (p_1 p_3)^i (1 - p_1 p_3)^{d-i}$$

$$\mathbb{E}(U_j) = \sum_{t=1}^{T} d p_1 p_3 \tag{3}$$

Since $p_2 \geq p_3 \rightarrow \mathbb{E}(U_j) \geq \mathbb{E}(U_j)$ if $\epsilon_i \geq 0$, it is expected that Θ_H would gain more utility than Θ_B for deviating from Y_i, but calculating or observing this utility change is difficult with large k.

The optimal action steps for N_i and N_j are now identified, regardless of type, to further differentiate the behaviour of Θ_H and Θ_B. This is done by determining the existence of a Bayesian Nash Equilibrium (BNE), a point in the repeated game where both players' expected utility is maximized, which follows similar methodology to Theorem 1 in [39].

Theorem 1. *For a BNE in the two-player scenario described for decentralized SGD, all types in $\mathbf{\Theta}$ seek to obtain a noise vector $Y_j \in \bar{Y}_j$ where the optimal Cumulative Distribution Function (CDF) of Y_j for variable in range d is*

$$F_{Y_j}^* = \frac{1}{2} F_{Y_j}(\mathcal{N}(0, \epsilon_k)) \tag{4}$$

if $\mathcal{N}(0, \sigma_k), Y_j$ and ϵ_k are independant random variables. If ϵ_k is not independant of Y_j, then

$$F_{Y_j}^* = \frac{P_\epsilon}{P_\epsilon + 1} F_{Y_j}(\mathcal{N}(0, \epsilon_k)), \tag{5}$$

where $P_\epsilon = \frac{\partial}{\partial F_{Y_j}} \epsilon_k$.

Proof. Given the expected utility for each player type derives in (2) and (3), each player seeks to maximize their own utility with the following

$$\text{maximize } U(F_i^*, F_j^*) \geq U(F_i, F_j^*)$$

to form a BNE. Θ_H seeks to change Y_j for each weight value with the constraint

$$maximize \sum_{t=0}^{T} d p_1 p_2$$

$$p_2 \leq \int_0^{Y_i} (Y_j + \mathcal{N}(0, \epsilon)) d F_{Y_j}, \text{and } p_1, p_2 \geq 0$$

while Θ_B seeks to maximize utility with the proceeding constraint

$$maximize \sum_{t=0}^{T} d p_1 p_3$$

$$p_3 \leq \int_0^{Y_i} (Y_j + \mathcal{N}(0, \epsilon) + \epsilon_k) d F_{Y_j}, \text{and } p_1, p_3 \geq 0.$$

Assuming the utility is convex and the Karush-Kuhn-Tucker (KKT) conditions apply [40], the Lagrangian is used to solve for ideal values of the success probabilities for each type. For independant ϵ_k and Y_j with the first Lagrangian multiplier being $\lambda_1 = d$ (assuming p_1 and p_2 are nonzero), the solution becomes as follows:

$$d \frac{\partial}{\partial F_{Y_j}} p_1 + \lambda_1 \frac{\partial}{\partial F_{Y_j}} p_2 = d \frac{\partial}{\partial F_{Y_j}} p_1 + d \frac{\partial}{\partial F_{Y_j}} p_2 = 0.$$

Taking the derivative of the integral representing the probabilities

$$F_{Y_j}(Y_i - \mathcal{N}(0, \epsilon_k)) + F_{Y_j}(Y_i) = 2F_j(Y_i) - F_{Y_j}(\mathcal{N}(0, \epsilon_k)) = 0$$

$$F_{Y_j}^* = \frac{1}{2} F_{Y_j}(\mathcal{N}(0, \epsilon_k))$$

and thus concludes the proof for the case where ϵ_k is independant of Y_j. If ϵ_k is not independant of Y_j, then $\lambda_1 = \frac{d}{\frac{\partial}{\partial F_{Y_j}} \epsilon_k}$ and

$$d \frac{\partial}{\partial F_{Y_j}} p_1 + \frac{d}{\frac{\partial}{\partial F_{Y_j}} \epsilon_k} \frac{\partial}{\partial F_{Y_j}} p_2 = 0.$$

Solving in a similar manner for the independant case, the final equation results in

$$F_{Y_j}^* = \frac{P_\epsilon}{P_\epsilon + 1} F_{Y_j}(\mathcal{N}(0, \epsilon_k)),$$

where $P_\epsilon = \frac{\partial}{\partial F_{Y_j}} \epsilon_k$. Thus concludes the proof for the dependent case.

□

For the case where ϵ_k is independant of Y_j, each type in $\mathbf{\Theta}$ is seeking to reduce $\bar{Y}_j$ to the same bound to maximize utility. A key factor of determining Θ_j is observing and remembering how quickly each node reduces $\bar{Y}_j$. Since $p_2 \geq p_3$ if $\epsilon_k \geq 0$, then a node with Θ_B will reduce $\bar{Y}_j$ slower that Θ_H. How each node accomplishes this is dependant on the algorithm chosen to optimize parameters. To determine whether N_j is behaving maliciously, a bound for differentiating the two types of behavior is determined. Since the Distributed Parallel SGD (DPSGD) algorithm and the expected rates of convergence are well-understood, a known boundary for determining whether a node is behaving maliciously can be determined from the theoretical analysis [36] as given in Theorem 2.

For the case when ϵ_k is dependent on Y_k, Θ_B will seek to reduce their perturbation to a tighter bound compared to the independant case that is dependant on the magnitude of ϵ_k and will have the same F^* if $|\frac{\partial}{\partial F_{Y_j}}\epsilon_k| = 1$. Since N_i is adding noise to each vector transmitted from N_j, it is assumed that Θ_B would want to find an effective perturbation where $|\frac{\partial}{\partial F_{Y_j}}\epsilon_k| \geq 1$ as smaller noise would be overshadowed by $\mathcal{N}(0, \epsilon_k)$, so N_i can identify Θ_j in a similar manner to the independant case. $\|\cdot\|^2$ denotes the squared ℓ_2 norm.

Theorem 2. *For the DPSGD algorithm [36], there exists a target that is used to differentiate Θ_H and Θ_B based on the rate of distributed convergence to $\mathcal{W}^*$. For an $n = 2$ scenario with learning rate α and local gradients ∇f_i, the following target can conservatively estimate honest behavior*

$$\alpha^2 \mathbb{E}(Var(\nabla f_i)) \leq \mathbb{E}\left\| \frac{1}{n}\sum_{i=1}^{n} W_i^* - W_i \right\|^2, \tag{6}$$

for an estimator of the gradient variance with small bias.

Proof. Following Theorem 1 in [36] starting at (12), the expected deviation from the ideal weights for a distributed task can be found by

$$\mathbb{E}\left\| \frac{\sum_{i=1}^{n} W_i^*}{n} - W_i \right\|^2 \leq \frac{4\alpha^2\sigma^2}{1-\lambda_2} + \frac{36\alpha^2\varsigma^2}{(1-\sqrt{\lambda_2})^2} + F(\lambda_2). \tag{7}$$

$F(\lambda_2)$ includes some lengthy terms that get cancelled with the proceeding steps. For a two node network, the matrix of connectivity constants C has the eigenvalues $\lambda(C) = (1, \lambda_2)$, where $\lambda_2 = (C_{ii} - C_{ij})^2$. The values of C_{ij} indicate the contribution for N_j to distributed learning. For a value of $\lambda_2 \approx 0$ with the difference in shared weights being small, this indicates the contribution of a node with Θ_H. Plugging $\lambda_2 \approx 0$ into (7) and $n = 2$:

$$\begin{aligned}\mathbb{E}\left\| \frac{\sum_{i=1}^{n} W_i*}{n} - W_i \right\|^2 &\leq \frac{4\alpha^2\sigma^2}{1-\lambda_2} + \frac{36\alpha^2\varsigma^2}{(1-\sqrt{\lambda_2})^2} + F(\lambda_2) \\ &= \alpha^2(4\sigma^2 + 36\varsigma^2)\end{aligned}$$

With the above bound defined for cooperating with a node with honest behavior, a conservative target below the bound is now defined. With the constant terms generated multiplying σ and ς, a Maximum-liklihood Estimator (MLE) is created for ς where $\mathbb{E}(Var(\nabla f_i)$. This estimator allows the local node to approximate a parameter of the overall distribution of the global gradients based on its own local experience. This estimator is always below the honest bound derived above as long as the bias from the estimator is smaller than the constant factors. We now define a conservative target below the bound provided in the above equation, where

$$\alpha^2 \mathbb{E}(Var(\nabla f_i)) \leq \alpha^2(4\sigma^2 + 36\varsigma^2)$$

□

With the conservative bound determined in Theorem 2, there is now a quantifiable way of determining if N_j is manipulating D_j and not converging to a BNE as fast as Θ_H and thus the type of N_j. One of the key benefits of this approach is that for any $\epsilon_i \geq 0$, suspicion can be generated about Θ_j .

B. Reputation System

Note N_i develops p_i by observing S_j from N_j. For forming a solid prediction about blockchain data with complex distributions, it is assumed d is somewhat large for $\mathcal{W}_k$, creating a large state-space that cannot be reduced to a lookup table for anomaly detection. To create connections and cutoffs with entries in C, where $\{C_{ii} = 1, C_{ij} = 0\}$ is set for Θ_B and $\{C_{ii} = 1 - C_{ij}, 1 > C_{ij} > 0\}$ for Θ_H, deep reinforcement Q-learning was chosen to reduce the state space. Similar approaches to this work, such as meta-learning [41], learn how to optimize an inner classifier with selection of parameters based on performance on many trials.

This setup follows optimization for an 'inner' learner provided an 'outer' learner observes performance and explores parameters spaces, which has been explored with reinforcement Q-learning [42]. Standard greedy Q-learning approaches address issues concerning balancing short-term and long-term goals with the Bellman operator $Q(S, A) = R(S, A) + \gamma \max_A Q * (S^*, A)$ where γ is long-term bias towards the largest value Q^*. In ultralight forms of Q-learning with small state spaces, a simple lookup table can be used to determine the optimal action for a given state. In scenarios where the number of variables or state-space is large like this blockchain scenario, deep neural networks are used to identify the current state [25], [28].

A method for forming reputation about different nodes in a blockchain network is proposed with the following, called Weight-based Reputation (WBR), and is shown in Algorithm 1. Reputation is first formed from a value function given by (8). From there, the reputation values are used to generate Q-values for reinforcement learning. An overview of calculations is shown in Fig. 2, with $A = \{i, j\}$ denoting calculations from both N_i, N_j.

For determining Θ_j, N_i creates a biased opinion about learning from distributed data parsed from the blockchain. N_i knows that its own data is a reliable ground truth, so to develop p_i it compares perturbed values of D_j to its own past experience as well as other nodes. An $N_j \in H$ will follow similar patterns to N_i, which N_i can determine with the following

$$Rep_j = ||S_i - S_j|| + ||W_i - W_j| - \alpha^2 \mathbb{E}(Var(\nabla f_i))\| \tag{8}$$

which contains the target derived for determining adversarial behavior in Theorem 2 and a state vector created for N_j with

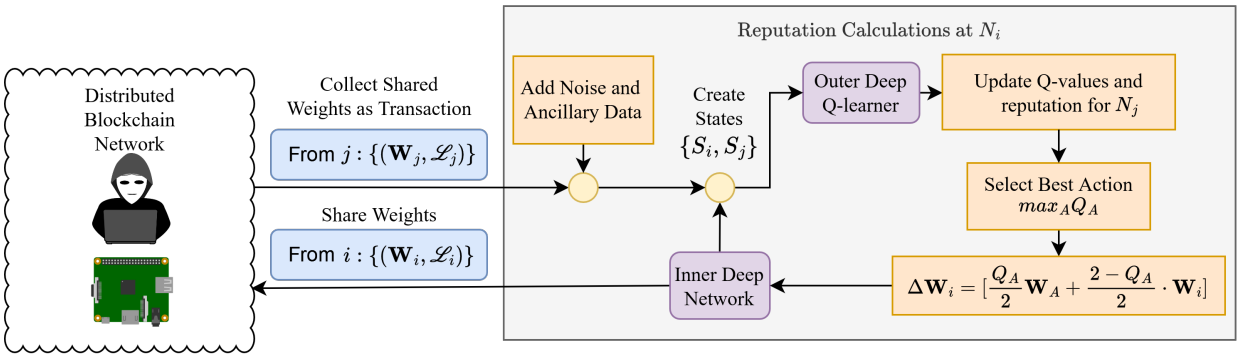


Fig. 2: Calculation overview for determining reputed nodes.

Algorithm 1 WBR Reputation System

1: Initalize: Local Inner Learner L_I, Outer Q-learner L_O
2: Nodes $A = \{i, j\} \in N$, Initialize N_i: $\sigma_\epsilon, k, Rep_{j,0} = 0$
3: **for** t in T **do**
4: Obtain $D_i = (\mathcal{W}_i, \mathscr{L}_i)$ from L_I
5: Receive from k nodes: $D_j = (\mathcal{W}_j + \mathcal{N}(0, \sigma_\epsilon), \mathcal{L}_j)$
6: Calculate Rep_A with (8)
7: Scale $Rep_A = min(Rep_A)/Rep_A$
8: $Q(A) \leftarrow (1-\alpha)Q(A) + \alpha[Rep_A + \gamma max_A Q^*]$
9: $Rep_A = \frac{Q(A)-Q_{Min}}{Q_{Max}-Q_{Min}}$
10: $\mathbf{W}_I = \mathbf{W}_I - \alpha[\frac{Q_A}{2}\nabla f_A + \frac{2-Q_A}{2}\nabla f_i]$
11: **end for**

S_j defined with the game previously. Creating a reward based on both the state distance as well as the conservative bound prevents malicious nodes from slightly modifying their own distributed weights and trying to falsely increase their reward by reducing their distance from W_i. Rep_j is also scaled based on the minimum reward to prevent Sybil adversaries from repeatedly trying to game the reward. Since contact cannot be cutoff from the honest nodes with the current threat model, malicious Sybil adversaries will not be able to skew the reward distribution by spamming garbage data within the acceptable boundaries.

Some of the design choices used in the reputed system were inspired from related robustness techniques in the literature, such as randomized smoothing [20] and weighted distance to prevent influence of Sybil adversaries [30], [43]. Where this technique benefits from other approaches is that it allows for a direct learning about adversary behavior, as opposed to creating a fixed statistical bound with other approaches that is decided globally for all nodes. This allows N_i to decide a case-by-case basis for determining the influence of N_j to distributed learning. In addition, the deviation of suspicious nodes can be easily quantified and observed with the reputation values and stored in the blockchain for auditing and viewing. Similar approaches developed alongside for generating reputation based on weight contribution to a group task assume a centralized server aggregates weights like ByGars [44].

C. Robustness Guarantees

The amount of error for this approach for determining $\mathcal{W}^*$ is entirely dependant on the Q-learning scheme chosen to resolve the state-space and find the optimal action. For simple Q-learning algorithms that operate on a finite state-space, similar to the multi-arm bandit problem, regret analysis can be used to find the amount of lost reward between the optimal and suboptimal actions and create a formula to describe the security strength [45]. For deep Q-learning schemes, there is not a universal metric to compare security for a reinforcement learning setting. It is possible to frame the security strength with this approach in the context of the performance of an arbitrary deep Q-learning scheme. For this approach it is assumed that a Θ_B node will always produce a suboptimal Q-value, while Θ_H will produce a Q-value in the neighborhood of Q^*. A single time-step error bound is defined in Theorem 3 as follows for a Q-learning approach.

Theorem 3. *The expected error for weight updates introduced in Algorithm 1 for both types of nodes with possible additive adversarial noise bound* $\mathbf{E}$ *is found by*

$$\mathbb{E}\left\|\frac{\sum_{i=1}^{n} W_i*}{n} - W_i(t)\right\|^2 \leq \alpha^2(4\sigma^2 + 36\sigma^2) + \frac{\alpha Q_E}{2}\mathbb{E}\left\|\nabla f_i + \mathbf{E}\right\|^2 \quad (9)$$

where $Q_E = (Q(\hat{s}_a, a) - Q_{Min})/(Q^* - Q_{Min})$ *is the scaled error created from the Q-learning scheme and* $\hat{s}$ *is the state locally perturbed with the noise added in Line 5.*

Proof. Using the bound from Theorem 1 in [36] for $n = 2$ and the triangle inequality where N_i is assumed to produce optimal weights

$$\begin{aligned}\mathbb{E}\left\|\frac{\sum_{i=1}^{n} W_i*}{n} - W_a\right\|^2 &= \mathbb{E}||(W^*(t-1) - \alpha\nabla f_i) - \\ &(W_i(t-1) - \alpha(\frac{2-Q_E}{2}\nabla f_i - \frac{Q_E}{2}\nabla f_j)||^2 \\ &\leq \alpha^2(4\sigma^2 + 36\sigma^2) + \left\|\frac{\alpha Q_E}{2}(\nabla f_i + \nabla f_j)\right\|^2\end{aligned}$$

The second term is bounded assuming the adversary follows the same distribution of gradients as N_i with strictly additional

noise

$$\mathbb{E}\left\|\frac{\alpha Q_E}{2}(\nabla f_i + \nabla f_j)\right\|^2 \leq \frac{\alpha Q_E}{2}\mathbb{E}\left\|\nabla f_i + \mathbf{E}\right\|^2$$

Plugging in the above bound into the previous equation, this concludes the proof. □

This bound is influenced asymptotically by the error of the Q-learning scheme to find Q^*. This approach then provides a robust boundary depending on the Q-learning algorithm chosen. For a standard DQN, a definitive bound can be made following Theorem 6.1 in [46] to produce an upper bound on the overall error. The bound in (9) indicates that the adversarial influence is directly determined by the opinion of the Q-learning scheme. Standard approaches in estimating the Q-values like DQN would expect to have a higher error rate. Approaches with a conservative estimate or assumptions about the Q-value distribution, such as CQL [27] that learns lower-bounded Q-distributions and aims to make conservative estimates, will have tighter bounds and better performance.

IV. Results and Discussion

Two simulation setups were created to evaluate WBR. The first was an idealistic setup created in Python 3.9 using PyTorch 1.10 to classify an Ethereum fraud dataset [47]. A repository for the idealistic environment is provided [2]. The second setup contained the same Python environment in a realistic setting using the IOTA GoShimmer 0.9.8 framework[3] to simulate a small blockchain network for confirming and distributing transactions in an IoT-like environment. This setup was executed with Ubuntu 20.04 LTS with 128 GB DDR4 RAM and 48 Intel Xeon Gold 6136 CPUs at 3.00GHz using Docker 20.10.21. The classifier and WBR were executed in parallel to GoShimmer with communication through a custom web API to a custom data application for generating transactions. Each node was limited to 10 CPU threads with 4 GB of RAM each, mimicking that of IoT device resources specialized for machine-learning[4].

Parameters for both setups are shown in Table I, with parameter notation for compared approaches kept the same from their original works for consistency. Network simulation consisted of a fixed number of nodes with adversarial percentage f conducting various attacks. Results from each experiment were validated with results averaged over 10 trials similar to the published literature [27]. The inner classifier on each node was a two hidden layer Multi-layer Perceptron (MLP) network that was optimized using DPSGD with learning rate α. For a train/test split percentage, each node uses an equal training with different sampling seeds, and each node communicates with k other nodes requesting shared parameters. Several q-learning schemes were tested equally among nodes for different trials: CQI [28], DQN [25], DDQN [26], CQL [27]. Each q-learning scheme used a moving average as the MLE for reward in (8) with number of samples A_m. Each q-learning algorithm was used in conjunction with WBR (noted as DQN-WBR, etc.) and shared values for q-learning learning rate α and bias γ.

The CQI approach is an interpretable decision tree for finding the state-space representation that would be practical for auditing in a blockchain protocol, while the other approaches are deep black-box q-learning algorithms. CQI has a starting threshold H_S to split tree nodes on Q-value changes with decay factor D and a fixed tree depth maximum. Tree nodes are visited recursively with decay factor d, with the least visited nodes trimmed given a parameter percentage. DDQN and CQL can be considered variations on the DQN algorithm for using temporal-difference deep q-learning, with each approach optimized using Adam with rate α and two hidden-layers with a batch parameter.

For evaluating the security of the reputation scheme to other similar approaches in the literature, the Multi-Krum [30] and Bridge-T [31] robustness techniques were implemented for comparison. Both the Multi-krum and Bridge-T methods use a lightweight form of averaging to remove weight outliers for local learning. Multi-Krum uses the Krum operator to calculate the distance of a new weight from the local vector and averages m samples, while the Bridge-T method removes outliers and assumes there is a fixed number of adversaries b. Two attacks, outlined in Section II-B were conducted to test robustness of various techniques, the first being the Uniform attack with magnitude u and the MAD [21] attack with parameters η, β, ϵ.

TABLE I: Simulation parameters

System	Parameter
Net. Sim.	Nodes $= 30$, Time $= 3000$, $f = 0.3$, Trials $= 10$
Inner Learner Task	Train % $= 0.8$, Neurons $= [120, 50]$, DPSGD $\alpha = 0.7$
Reputation System	$\sigma_\epsilon = 0.001$, $k = 9$, $A_m = 10$
Q-learning	$\alpha = 0.7$, $\gamma = 0.99$
CQI [28]	$H_S = 1$, $d = 0.9$, $D = 0.999$, Trim % $= 0.1$, Depth $= 3$
DQN [28], DDQN [26], CQL [27]	Adam $\alpha = 0.001$, Neurons $= [150, 50]$, Batch $=$ |Nodes|
Attacks	Uniform $u = 1E-5$, MAD $\epsilon = U(0,1)$ MAD $\alpha = 1E-5$, MAD $\beta = 1E-5$
Multi-krum	$m = 3$
Bridge-T	$b = 3$

A control scenario for the idealistic setup plotting inner learning loss for all approaches is shown in Fig. 3 without attacks.

A. Ideal Environment Results

Following the considered adversarial attacks outlined in Section III, each data poisoning attack was tested in conjunction with two strategies. These policies, called 'pure' and 'mixed', follow their respective game-theoretic origins where an adversary consistently chooses an action to attack or has a positive probability associated with the attack. For the mixed strategy, $P(\text{Attack}) = \mathbb{E}(Rep_j)$ for N_j. Each approach is an attempt at the optimal solution proposed in Theorem 1,

[2]Github link to be added in the final journal version.
[3]GoShimmer Wiki
[4]Jetson AGX Xavier Series

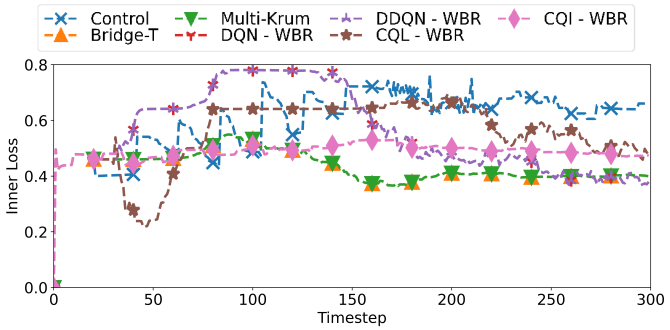
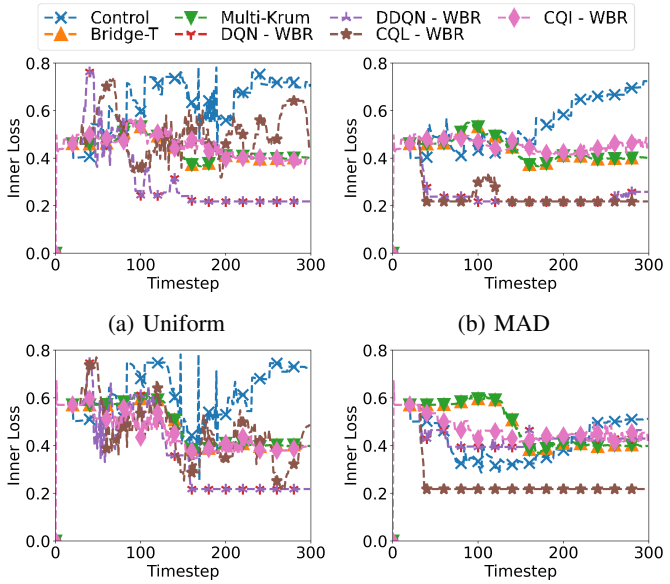


Fig. 3: Control execution.

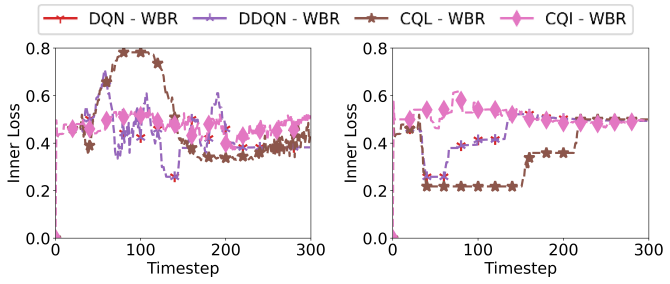
with varying amounts of intelligence. Results for the attack configurations with the pure and mixed strategy are shown in Fig. 4 and 5 respectively. Fig. 4 also shows the pure strategy used in conjunction with a Sybil attack where adversary nodes with pure strategies outnumber the majority with f percentage doubled ($f = 0.6$). Fig. 5 only shows the methods that use the robust reputation system since adversary mixed actions are dictated by reputation values.



(a) Uniform (b) MAD

(c) Uniform + Sybil (d) MAD + Sybil

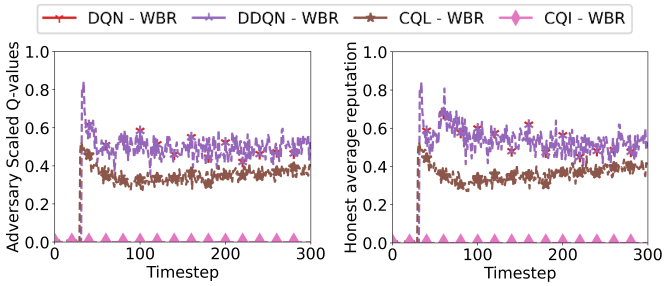
Fig. 4: Idealistic pure strategy results.



(a) Uniform (b) MAD

Fig. 5: Idealistic mixed strategy results.

Overall, the WBR approach with the different q-learning schemes performs better than the control, Bridge, and Multi-krum techniques for reducing loss to the inner classifier. DQN and DDQN both perform well in 3/4 pure strategies, while CQL performs well only for the MAD+Sybil attack. In the mixed scenario, each reputation approach seems to perform similarly to each other. The uniform mixed attack seems to cause CQL to increase loss before recovering, while the opposite occurs for the MAD attack. Compared to the control case in the pure strategies, the mixed reputation system does outperform, though it becomes harder to determine optimal nodes to chose from in the mixed scenario.



(a) Pure MAD (b) Mixed MAD

Fig. 6: Idealistic malicious node Q-values.

Learning Q-values for the best-case and worst-case attack scenarios are shown in Fig 6, being the MAD attack with pure and mixed strategies. For both attack scenarios, the DQN and DDQN learners both give the adversaries a quick increase in reputation as they begin receiving data, but slowly decrease reward over time as they observe noise in the adversaries' strategies. The CQL learner initially performs well for both cases, but stabilizes around 0.5 for the uniform attack for the remainder of the test. CQI generates extremely conservative reputation values for the adversarial nodes, which is likely the case it outperformed the other q-learning schemes in the MAD scenarios. CQI has Q-values that are very small (though non-zero), which is likely due to the limits of the tree-generation from the approach and can be seen in the results pure and mixed attacks.

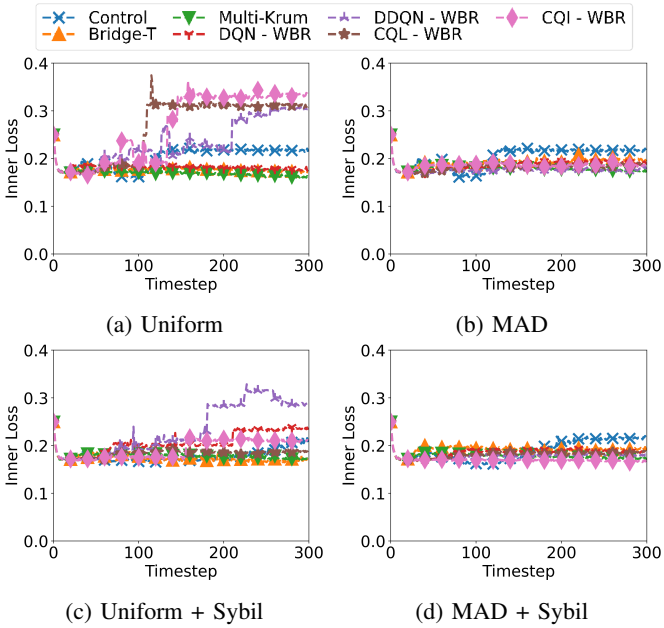
B. Realistic Simulation Results

Using the realistic reputation setup, the same attack scenarios for both the Uniform and MAD attack were simulated using the same setup shown in Table I for the pure, mixed and pure+sybil attack combinations. Fig. 7 and 8 show the pure and mixed strategies respectively. Note the difference in y-axis scale for the realistic setting.

The results for the realistic simulation differ drastically for the idealistic case. The only theoretical difference between the idealistic and realistic simulations is that instead of uniformly pinging each node for data updates, each node gossips weight updates through GoShimmer to be received by each node. There may be cases where network delays or transmission errors lead to some updates being dropped, which has shown to have a critical impact on overall performance. The MAD attack for the pure and mixed strategies was basically ineffective

TABLE II: Single-step complexity comparison [30], [31]

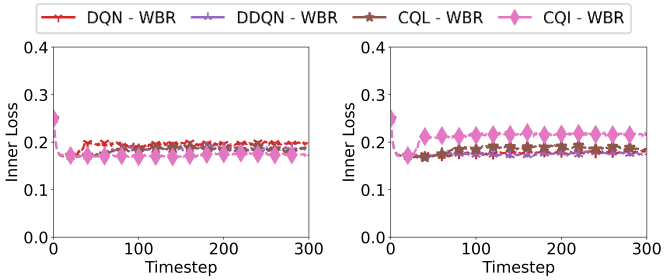
Algorithm	Time	Space	Communication
Multikrum	$\mathcal{O}(mk^2d)$	$\mathcal{O}(md)$	$\mathcal{O}(kd)$
Bridge-T	$\mathcal{O}(bkd)$	$\mathcal{O}(kd)$	$\mathcal{O}(kd)$
WBR	$\mathcal{O}(kQ) \rightarrow \mathcal{O}(k(dN + (L-1)N^2))$	$\mathcal{O}(nd + Q) \rightarrow \mathcal{O}(nd + Nd + (L-1)N^2)$	$\mathcal{O}(kd)$



(a) Uniform (b) MAD

(c) Uniform + Sybil (d) MAD + Sybil

Fig. 7: Realistic pure strategy results.



(a) Uniform (b) MAD

Fig. 8: Realistic mixed strategy results.

against drastically increasing loss, while the uniform attack was very effective for the reputation system. Mixed strategies also seemed ineffective regardless of attack.

C. Complexity Comparison

The asymptotic complexity of the WBR reputation system was compared to other robustness technique approaches for distributed learning. This was done for the time, space and communication complexity in Table II. Q notes the complexity of the q-learning algorithm used in WBR, with DQN used as a replacement for each entry in WBR. For each approach, k is the number of nodes in communication in the connectivity graph of the network and d is the number of weights shared between nodes. For Multikrum, m is the number of weight vectors chosen for comparison. BRIDGE-T is a fairly lightweight algorithm with a parameter for removing the number of outliers in weight distribution b that is fixed for algorithm execution. For the DQN approach used to replace the q-learning scheme, L is the number of layer in the MLP network and N is the number of neurons, where it is assumed that beyond the presentation layer of D input features to the network, the other weight matrices are square with N neurons. The calculation of the reward in (8) is assumed to be asymptotically overshadowed by the weight calculations of the Q-network.

For time complexity, the base calculations compared to other robustness techniques for WBR are not drastically dissimilar beyond the quadratic complexity of Multi-krum. Space complexity is increased for the WBR approach since each nodes' weights are store for analysis by the q-learning solvers. Each approach has fairly minimal communication complexity, determined mostly by the connectivity of the network with k nodes available.

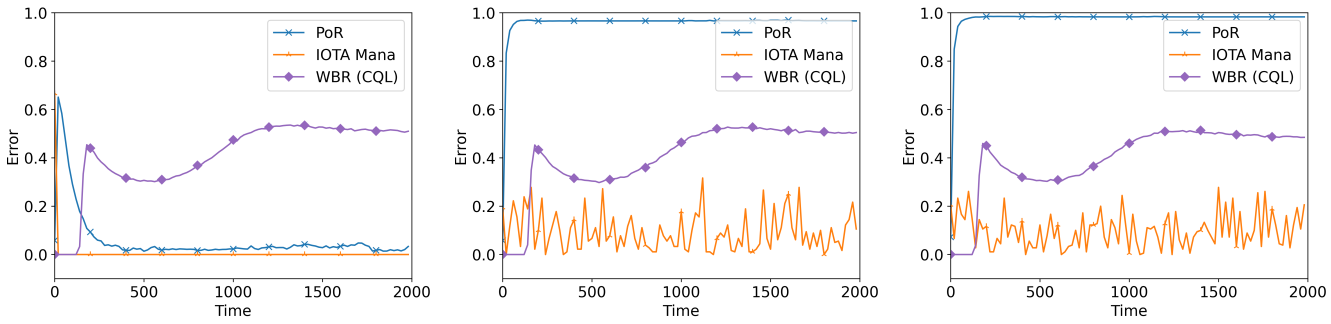
D. Computational Performance Results

A comparison study between WBR and standard PoW in IOTA was used to show resource consumption differences in simulation. Using the same parameters in Table I, built-in Docker command-line tools recorded hardware resources during tests. Ten node containers were limited to 20 CPU threads with 3 GB of RAM, mimicking a resource-constrained machine-learning IoT device such as the Nvidia Jetson AGX Xavier[5]. These consisted of processing 1000 *pseudo*-transactions through a custom dApp with the GoShimmer framework while simultaneously conducting 100 steps of WBR using the DQN approach. Pseudo-transactions process identically to regular ones minus bookkeeping to confirm account balances to allow for parsing of arbitrary account types for classification like account systems in the Ethereum Fraud dataset. The IOTA PoW used for comparison was set to difficulty level 21 [48] for pseudo-transactions and 2 otherwise. Averaged results over 5 trials are shown in Table III.

TABLE III: Average Resource Consumption Comparison

Method	IOTA PoW	WBR (Train)	WBR (Test)
Memory [%]	16.4	22.03	**14.11**
CPU [%]	16.0	35.5	21.86
Power [W]	6.32E-6	17.3	8.43E-6
Blocks/Second [bps]	20.93	18.75	**29.3**
Block Delay [ms]	12.4	**0**	**0**

[5]Jetson AGX Xavier Series



(a) Control (b) Attack (c) Sybil and Attack

Fig. 9: Blockchain reputation systems average absolute agreement error comparison.

Resource consumption with online training using WBR is more computationally intense with the full implementation of PyTorch. Testing phases showed a much improved performance decrease while increasing throughput by 50%, while training phases showed a decrease in throughput due to exchange of weight data for processing WBR. Calcuations for the IOTA mana reputation system was removed from consideration in these tests since the execution is very lightweight compared to PoW. Since mana also utilizes a separate consensus scheme outside of computation, it wasn't considered a fair comparison. Future work will provide a direct comparison to WBR with alterations for consensus.

E. Comparison to Blockchain Reputation Systems

The WBR approach was also compared to other existing blockchain reputation-based consensus mechanisms: the IOTA mana reputation system [34] and Proof-of-Reputation (PoR) [49]. The mana system was the only reputation system from surveyed literature that had strong mathematically design justification for active security. While the mana system is technically considered a reputation system, it is actually used in conjugation with IOTA's Fast Probabilistic Consensus (FPC) for making decisions based on cellular automata where greater reputation values are arbitrarily assigned to nodes with good behavior for making and confirming transactions. The Mana and FPC system was simulated using the official FPC simulator[6]. The PoR approach was selected as a baseline to compare against systems with stronger theoretical guarantees, while the original publication does not provide strong justification for selection of some of its parameters.

Each approach was simulated to generate a rough estimate for how they converge in agreement error for a control case, a respective effective attack and a network Sybil attack combined with the effective attack where adversaries make up more than 50% of the network. Simulation parameters comparing each approach are shown in Table IV. PoR has an adjustable parameter ϕ for reputation values and λ for transaction generation rate, while a moving average with window size W was added to smooth values. Mana has a Binomial distribution affecting node opinions $\mathcal{B}(n = 2000, p)$ making k queries with threshold τ, with q proportion of adversarial nodes affected by parameter β.

Each reputation system was compared to show roughly how reputation accrues for suspicious nodes in a control environment and to Sybil-based attacks which are defined in their respective works [49], [50] or in Section II-B. This does not serve as a comprehensive security comparison between each system, but does show how each system typically behaves under different attacks. Fig. 9 shows the control and two attack scenarios with error scaled to $[0, 1]$ for comparison. For PoR and WBR, agreement error is calculated based on the reputation for the malicious nodes, while the mana system error is calculated based on the difference in the final value agreed upon by nodes in the network and not raw reputation.

TABLE IV: Blockchain reputation system parameters

WBR	PoR [49]	IOTA Mana	WBR (CQL)
Nodes	100	100	100
Time	2000	2000	2000
Attack %	30	30	30
Sybil %	60	60	60
Attack	On-off [49]	Berserk [50] (Maximize Uncertainty)	Uniform (Sec II-B)
Param.	$\phi = 0.1$, $\lambda = 10$, $W = 30$	$p = 0.95$, $\tau = 0.66$, $\beta = 0.3$, $k = 10$, $q = 0.3$	See Table I

Each approach in the control case does converge in agreement over time for each approach. In the attack scenarios, PoR completely fails to reduce the reputation for the adversary nodes. The IOTA system also experiences some error that does not converge, though the results show relatively low error under 0.5 compared to PoR and can be reduced with parameter variation. The Sybil approach shows similar results to the vanilla attack case for both PoR and IOTA, with IOTA's being slightly reduced. The WBR approach does not change drastically with attacks compared to the control case, so attacks were not as effective at altering reputation compared to other blockchain reputation systems. The reputation values for WBR also seem to be more stable for the $n = 100$ case compared to the other simulation results.

[6]IOTA FPC Simulator

To summarize, reputation systems without mathematical justification like PoR fail even under minor attacks from the proposed simulation. IOTA and mana perform the best in terms of agreement error, but noise is consistently present between nodes since values are not remembered for a great period of time. WBR values do converge to 0.5 but don't reduce to zero due to lack of consensus between nodes. PoR should not be implemented for secure blockchain scenarios, mana can be used but lacks memory, and WBR lacks strong consensus.

F. Summary of Results

WBR outperforms the Multi-krum and Bridge robustness schemes in the explored attack scenarios. The experimentally best q-learning algorithm for WBR is dependant on the type of attack performed. For standard pure attacks, CQL performs well, while CQI outperforms with Sybil cases. For mixed strategies, DQN and DDQN outperform the more conservative cases with faster action-taking. With these results, DQN is currently declared the best algorithm. The q-learning approaches were able to improve over other robustness techniques roughly 50% in the pure attack settings.

WBR adds minimal computational demand beyond the deep network calculation dependant on the q-learning scheme chosen. WBR also is able to produce more stable reputation values compared to other blockchain reputation schemes that is robust to adversarial perturbation. A definitive comparison between the blockchain systems cannot be made yet as the WBR as consensus guarantees have not been defined. WBR does converge well between nodes and remembers tasks, but lacks consensus to finalize decisions.

V. Conclusions and Future Work

WBR was implemented and scrutinized under many attack scenarios. Compared to other state-of-the-art protocols for robust learning amidst Sybil-capable adversaries, this effort outperformed and repelled attacks in all test cases. Inner training loss was improved up to 50% with the DQN solver, performing well in scenarios with pure and mixed attack strategies. WBR was also found to be more robust to adversarial perturbations than other blockchain reputation systems, even with lack of consensus.

As part of future work in enhancing WBR is to develop a policy between the honest nodes for forming a consensus and determining what transactions are considered valid. The current version of WBR could be considered to have a weak form of averaging consensus, but this does not show fast agreement from the simulation results. Another direction open for future research is expanding the WBR technique to other distributed algorithms that extend SGD, such as SGD with momentum or Adam.

Acknowledgements

This work was supported by the Graduate Assistance in Areas of National Need (GAANN) national fellowship program. In addition, the authors thank the Missouri S&T HPC Center, which is supported in part by the National Science Foundation under Grant No. OAC-1919789, for providing the computing hardware for executing the simulations. The authors also thank Cheng-Yuan Wang and Payton Redemer for the implementation of the GoShimmer simulation combined with the intelligent prediction.

References

[1] A. A. Sadawi, M. S. Hassan, and M. Ndiaye, "A Survey on the Integration of Blockchain With IoT to Enhance Performance and Eliminate Challenges," *IEEE Access*, vol. 9, pp. 54 478–54 497, 2021. DOI: 10.1109/ACCESS.2021.3070555.

[2] S. He, Q. Tang, C. Q. Wu, and X. Shen, "Decentralizing IoT Management Systems Using Blockchain for Censorship Resistance," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 1, pp. 715–727, Jan. 2020. DOI: 10.1109/TII.2019.2939797.

[3] Popov, Serguei, *The Tangle*, Apr. 2018. [Online]. Available: https://tinyurl.com/tanglewhitepaper.

[4] Nakamoto, Satoshi, *Bitcoin: A Peer-to-Peer Electronic Cash System*, 2009. [Online]. Available: https://bitcoin.org/bitcoin.pdf.

[5] I. Eyal and E. G. Sirer, *Majority is not Enough: Bitcoin Mining is Vulnerable*, arXiv, Nov. 2013. DOI: 10.48550/ARXIV.1311.0243.

[6] C. Decker and R. Wattenhofer, "Bitcoin Transaction Malleability and MtGox," in vol. 8713, arXiv:1403.6676 [cs], 2014, pp. 313–326. DOI: 10.1007/978-3-319-11212-1_18. [Online]. Available: http://arxiv.org/abs/1403.6676.

[7] L. Herskind, P. Katsikouli, and N. Dragoni, "Privacy and Cryptocurrencies—A Systematic Literature Review," *IEEE Access*, vol. 8, pp. 54 044–54 059, 2020. DOI: 10.1109/ACCESS.2020.2980950.

[8] M. Saad, J. Spaulding, L. Njilla, *et al.*, "Exploring the Attack Surface of Blockchain: A Comprehensive Survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 1977–2008, 2020. DOI: 10.1109/COMST.2020.2975999.

[9] A. Alzubaidi, K. Mitra, and E. Solaiman, "Smart Contract Design Considerations for SLA Compliance Assessment in the Context of IoT," in *2021 IEEE International Conference on Smart Internet of Things (SmartIoT)*, Aug. 2021, pp. 74–81. DOI: 10.1109/SmartIoT52359.2021.00021.

[10] J. Li, Y. Shao, K. Wei, *et al.*, "Blockchain Assisted Decentralized Federated Learning (BLADE-FL): Performance Analysis and Resource Allocation," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 10, pp. 2401–2415, Oct. 2022. DOI: 10.48550/ARXIV.2101.06905.

[11] S. M. H. Bamakan, A. Motavali, and A. Babaei Bondarti, "A survey of blockchain consensus algorithms performance evaluation criteria," *Expert Systems with Applications*, vol. 154, p. 113 385, Sep. 2020. DOI: https://doi.org/10.1016/j.eswa.2020.113385.

[12] K. Regan, P. Poupart, and R. Cohen, "Bayesian Reputation Modeling in E-Marketplaces Sensitive to Subjectivity, Deception and Change," in *Proceedings of the National Conference on Artificial Intelligence*, vol. 2, 2006. DOI: 10.5555/1597348.1597380.

[13] G. Rens, A. Nayak, and T. Meyer, *Maximizing Expected Impact in an Agent Reputation Network – Technical Report*, arXiv:1805.05230 [cs], May 2018. DOI: 10.48550/arXiv.1805.05230.

[14] M. Debe, K. Salah, M. H. U. Rehman, and D. Svetinovic, "IoT Public Fog Nodes Reputation System: A Decentralized Solution Using Ethereum Blockchain," *IEEE Access*, vol. 7, pp. 178 082–178 093, 2019. DOI: 10.1109/ACCESS.2019.2958355.

[15] M. T. d. Oliveira, L. H. A. Reis, D. S. V. Medeiros, R. C. Carrano, S. D. Olabarriaga, and D. M. F. Mattos, "Blockchain reputation-based consensus: A scalable and resilient mechanism for distributed mistrusting applications," en, *Computer Networks*, vol. 179, p. 107 367, Oct. 2020, ISSN: 1389-1286. DOI: 10.1016/j.comnet.2020.107367.

[16] R. Saha, G. Kumar, A. Brighente, and M. Conti, "Towards An Enhanced Reputation System for IOTA's Coordicide," in *2021 Third International Conference on Blockchain Computing and Applications (BCCA)*, Nov. 2021, pp. 26–33. DOI: 10.1109/BCCA53669.2021.9656975.

[17] C. Huang, Z. Wang, H. Chen, *et al.*, "RepChain: A Reputation-Based Secure, Fast, and High Incentive Blockchain System via Sharding," *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4291–4304, Mar. 2021. DOI: 10.1109/JIOT.2020.3028449.

[18] F. Li, Y. Yang, and J. Wu, "Attack and Flee: Game-Theory-Based Analysis on Interactions Among Nodes in MANETs," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 40, no. 3, pp. 612–622, Jun. 2010. DOI: 10.1109/TSMCB.2009.2035929.

[19] P. Christiano, "Provably manipulation-resistant reputation systems," in *Conference on Learning Theory*, PMLR, Jun. 2016, pp. 670–697. [Online]. Available: https://proceedings.mlr.press/v49/christiano16.html.

[20] S. H. Silva and P. Najafirad, *Opportunities and Challenges in Deep Learning Adversarial Robustness: A Survey*, Jul. 2020. DOI: 10.48550/ARXIV.2007.00753.

[21] H. Zhang, H. Chen, C. Xiao, *et al.*, "Robust Deep Reinforcement Learning against Adversarial Perturbations on State Observations," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 21 024–21 037. DOI: 10.5555/3495724.3497489.

[22] A. Kumar, A. Levine, and S. Feizi, *Policy Smoothing for Provably Robust Reinforcement Learning*, May 2022. DOI: 10.48550/arXiv.2106.11420.

[23] R. Yang, C. Bai, X. Ma, Z. Wang, C. Zhang, and L. Han, *RORL: Robust Offline Reinforcement Learning via Conservative Smoothing*, Oct. 2022. DOI: 10.48550/arXiv.2206.02829. [Online]. Available: http://arxiv.org/abs/2206.02829.

[24] B. G. Anderson and S. Sojoudi, "Certified Robustness via Locally Biased Randomized Smoothing," in *Proceedings of The 4th Annual Learning for Dynamics and Control Conference*, PMLR, May 2022, pp. 207–220. [Online]. Available: https://proceedings.mlr.press/v168/anderson22a.html.

[25] V. Mnih, K. Kavukcuoglu, D. Silver, *et al.*, "Playing Atari with Deep Reinforcement Learning," Dec. 2013. DOI: 10.48550/ARXIV.1312.5602.

[26] H. van Hasselt, A. Guez, and D. Silver, "Deep Reinforcement Learning with Double Q-learning," Sep. 2015. DOI: 10.48550/ARXIV.1509.06461.

[27] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative Q-Learning for Offline Reinforcement Learning," Jun. 2020. DOI: 10.48550/ARXIV.2006.04779.

[28] A. M. Roth, N. Topin, P. Jamshidi, and M. Veloso, "Conservative Q-Improvement: Reinforcement Learning for an Interpretable Decision-Tree Policy," *arXiv:1907.01180 [cs]*, Jul. 2019. DOI: 10.48550/ARXIV.1907.01180.

[29] Z. Wang, Y. Liu, Z. Ma, X. Liu, and J. Ma, "Lipsg: Lightweight privacy-preserving q-learning-based energy management for the iot-enabled smart grid," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 3935–3947, 2020. DOI: 10.1109/JIOT.2020.2968631.

[30] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent," in *Advances in Neural Information Processing Systems*, 2017. [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/f4b9ec30ad9f68f89b29639786cb62ef-Paper.pdf.

[31] C. Fang, Z. Yang, and W. U. Bajwa, "BRIDGE: Byzantine-Resilient Decentralized Gradient Descent," *IEEE Transactions on Signal and Information Processing over Networks*, pp. 1–16, 2022. DOI: 10.1109/TSIPN.2022.3188456.

[32] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of Computing*, Jul. 1996, pp. 212–219. DOI: 10.48550/ARXIV.QUANT-PH/9605043.

[33] I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable," *Communications of the ACM*, vol. 61, no. 7, pp. 95–102, Jun. 2018. DOI: 10.1145/3212998.

[34] IOTA Foundation, "The Coordicide," 2019. [Online]. Available: https://files.iota.org/papers/Coordicide_WP.pdf.

[35] A. Chakraborty, M. Alam, V. Dey, A. Chattopadhyay, and D. Mukhopadhyay, "Adversarial Attacks and Defences: A Survey," *arXiv:1810.00069 [cs, stat]*, Sep. 2018. DOI: 10.48550/ARXIV.1810.00069.

[36] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, *Can Decentralized Algorithms Outperform Centralized Algorithms? A Case Study for Decentralized Parallel Stochastic Gradient Descent*, Sep. 2017. DOI: 10.48550/ARXIV.1705.09056.

[37] J. Lee, L. Xiao, S. S. Schoenholz, *et al.*, "Wide Neural Networks of Any Depth Evolve as Linear Models Under Gradient Descent," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2020, no. 12, p. 124 002, Dec. 2020. DOI: 10.1088/1742-5468/abc62b. [Online]. Available: http://arxiv.org/abs/1902.06720.

[38] J. Lust and A. P. Condurache, *GraN: An Efficient Gradient-Norm Based Detector for Adversarial and Misclassified Examples*, Apr. 2020. DOI: 10.48550/arXiv.2004.09179.

[39] J. Liu, X. Wang, S. Shen, G. Yue, S. Yu, and M. Li, "A Bayesian Q-Learning Game for Dependable Task Offloading Against DDoS Attacks in Sensor Edge Cloud," *IEEE Internet of Things Journal*, vol. 8, no. 9, pp. 7546–7561, May 2021. DOI: 10.1109/JIOT.2020.3038554.

[40] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press.

[41] I. Khan, X. Zhang, M. Rehman, and R. Ali, "A Literature Survey and Empirical Study of Meta-Learning for Classifier Selection," *IEEE Access*, vol. 8, pp. 10 262–10 281, 2020. DOI: 10.1109/ACCESS.2020.2964726.

[42] R. Fakoor, P. Chaudhari, S. Soatto, and A. J. Smola, *Meta-Q-Learning*, Apr. 2020. DOI: 10.48550/arXiv.1910.00125.

[43] C. Fung, C. J. M. Yoon, and I. Beschastnikh, "The Limitations of Federated Learning in Sybil Settings," 2020, pp. 301–316. [Online]. Available: https://www.cs.ubc.ca/~bestchai/papers/foolsgold-raid2020-slides.pdf.

[44] J. Regatti, H. Chen, and A. Gupta, "Byzantine Resilience With Reputation Scores," in *2022 58th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, Sep. 2022, pp. 1–8. DOI: 10.1109/Allerton49937.2022.9929385.

[45] I. A. Kash, M. Sullins, and K. Hofmann, *Combining No-regret and Q-learning*, Jan. 2022. DOI: 10.48550/arXiv.1910.03094.

[46] J. Fan, Z. Wang, Y. Xie, and Z. Yang, *A Theoretical Analysis of Deep Q-Learning*, Feb. 2020. DOI: 10.48550/arXiv.1901.00137.

[47] R. A. FISHER, "The use of multiple measurements in taxonomic problems," *Annals of Eugenics*, vol. 7, no. 2, pp. 179–188, 1936. DOI: https://doi.org/10.1111/j.1469-1809.1936.tb02137.x.

[48] L. Helmer and A. Penzkofer, "Report on the energy consumption of the IOTA 2.0 prototype network (GoShimmer 0.8.3) under different testing scenarios," 2022, Unpublished Report. [Online]. Available: https://files.iota.org/papers/Report_on_the_energy_consumption_of_the_GoShimmer_network.pdf.

[49] F. Gai, B. Wang, W. Deng, and W. Peng, "Proof of Reputation: A Reputation-Based Consensus Protocol for Peer-to-Peer Network," in *Database Systems for Advanced Applications*, J. Pei, Y. Manolopoulos, S. Sadiq, and J. Li, Eds., Cham: Springer International Publishing, 2018, pp. 666–681.

[50] S. Popov and W. J. Buchanan, "FPC-BI: Fast Probabilistic Consensus within Byzantine Infrastructures," *Journal of Parallel and Distributed Computing*, vol. 147, pp. 77–86, Jan. 2021. DOI: 10.1016/j.jpdc.2020.09.002.

Charles C. Rawlins is a PhD Computer Engineering candidate at Missouri University of Science and Technology. He received his B.S. in Electrical Engineering from Montana Technological University with an interest in Computer Science and high honors. His research interests involve Internet of Things devices and security. Some of his hobbies include scuba diving and learning about cybersecurity focusing on cryptocurrency.

S. Jagannathan is at the Missouri University of Science and Technology (former University of Missouri-Rolla) where he is a Rutledge-Emerson Distinguished Professor of Electrical and Computer Engineering. He served as a Site Director for the NSF Industry/University Cooperative Research Center on Intelligent Maintenance Systems. His research interests include machine learning and cyber-physical system security, neural network control, prognostics/bigdata analytics, and autonomous systems/robotics.

Venkata Sriram Siddhardh Nadendla is an Assistant Professor in the Department of Computer Science at Missouri University of Science and Technology. Prior to joining, he worked as a postdoctoral research associate at Coordinated Science Laboratory in University of Illinois at Urbana-Champaign for two years. He works on the design and analysis of cyber-socio-physical systems, particularly on issues such as security, human-system interaction, behavioral decisions and resource allocation.