Journal of Network and Systems Management (2020) 28:478–501
https://doi.org/10.1007/s10922-020-09532-1

Toward a Machine Learning and Software Defined Network Approaches to Manage Miners' Reputation in Blockchain

Abdellah Kaci[1] · **Abderrezak Rachedi**[2]

Received: 31 July 2019 / Revised: 25 March 2020 / Accepted: 9 April 2020 / Published online: 23 April 2020

Abstract

In blockchain, transactions between parties are regrouped into blocks, in order to be added to the blockchain's distributed ledger. Miners are nodes of the network that generate new blocks according to the consensus protocol. The miner that adds a valid block to the distributed ledger is rewarded. However, to find a valid block, the miner needs to solve a computationally difficult problem, which makes it difficult to a single miner to gain rewards. Therefore, miners join mining pools, where the powers' of miners are federated to ensure stable revenues. In public blockchains, access to mining pools is not restricted, which makes mining pools vulnerable to considerable threats such as: block withholding (BWH) attacks and distributed denial of service (DDoS) attacks. In the present work, we propose a new reputation based blockchain named PoolCoin based on a distributed trust model for a mining pools. The trust model used by PoolCoin is inspired from the job market signaling model. The proposed PoolChain blockchain allows pool managers the selection of trusted miners in their mining pools, while miners are able to evaluate them. Furthermore, to detect malicious miners that claim bigger computing capacity, we also provided a machine learning module to estimate the real miners' capacities. The efficiency of the proposed trust model is studied and the obtained simulation results are presented and discussed. Thus, the model parameters' are optimized in order to detect and exclude misbehaving miners, while honest miners are maintained in the mining pool.

Keywords Blockchain · Distributed trust model · Miner reputation · Signaling games · Machine learning · Software defined networks

✉ Abdellah Kaci
ab_kaci@esi.dz

Abderrezak Rachedi
rachedi@u-pem.fr

[1] Ecole Nationale Superieure de Technologie (ENST), Algiers, Algeria

[2] Universite Paris-Est, Champs sur Marne, France

1 Introduction

The blockchain was introduced for the first time by the creator(s) of Bitcoin in [1]. Due to the success of Bitcoin, many others blockchain emerged. A blockchain uses a data structure called distributed ledger, that consists in an ordered list of transactions, regrouped into blocks. The same copy of the distributed ledger should be shared between all the nodes of the system. A blockchain uses a consensus protocol, in order to ensure that all nodes agree replicated updates on the distributed ledger [2].

Different consensus protocols are used in blockchains, such as Proof-Of-Work (PoW), Proof-Of-Stake (PoS), Practical Byzantine Fault Tolerance (PBFT), etc. The Proof-Of-Work (PoW) is used by common blockchains, such as BitCoin and Ethereum [3]. In order to validate transactions in Proof Of Work (PoW), miners should compute a value that when hashed, the hash begins with a number of zero bits. The average work required is exponential with the number of zero bits required. However, the verification requires executing only a single hash [1].

Nodes that verify transactions and participate in the generation of new blocks are called miners. From a single miner's point of view, the interval between mining events exhibits high variance. By consequence, miners organize themselves into mining pools, in order to make the mining revenues more predictable. All members of a pool work together to mine blocks, and share their revenues when one of them successfully mines a block [4].

In the mining pool, miners submit to the pool manager, partial proofs of work (also known as *shares*), having a difficulty level less than the difficulty level of the proof of work acceptable by the network. The pool manager, checks the submitted shares, in order to find among them a full proof of work valid according to the consensus protocol. This full proof will be used to claim the mining reward [4].

Mining pools use reward systems based on shares submitted by miners. For instance, in Proportional Payout Schema (PPS), a miner is rewarded proportionally to the number of its shares [5]. One of the commonly used reward mechanism is Pay-Per-Last-N-Shares (PPLNS), which distributes the reward among the miners that reported the N last shares [6].

Mining pools are competing between them to find valid blocks. Therefore, a mining pool may be confronted to a distributed denial-of-service (DDoS) attack from other mining pools [7]. Mining pools are also faced to the block withholding (BWH) attack, where malicious miners submit only partial proofs and withdraw the full proof of work [4]. The BWH attack uses the fact that miners are rewarded according to their submitted shares.

Haddadou et al. in [8] proposed a distributed trust model named DTM^2, based on the economical job market signaling model. DTM^2 allocates credits to nodes and securely manage them using Trusted Platform Modules (TPMs). DTM^2 applies costs on data reception and sending. However, DTM^2 rewards honest miners that broadcast information considered as useful by other nodes.

In order to incite miners to behave honestly, Tang et al. proposed in [9] a reputation based mechanism for mining pools, where the Pool Manager is selected

randomly at each round, such that the reputation of each miner is given by the Pool Manager. The reputation system proposed in [9] is vulnerable, due to the fact that the Pool Manager is randomly selected. In fact, if a malicious miner is selected as the Pool Manager, the malicious miner will be able to drive attacks on the mining pool. In addition, in [9], pool managers are able to evaluate miners, while miners are not able to evaluate pool managers.

In this paper, we propose to introduce the distributed trust model, in the miner selection problem. Therefore, we propose the PoolCoin blockchain that manages miners' reputations and their complaints against Pool Managers. The reputation of a miner is stored in the blockchain as a *trust score*. According to the miners' trust scores, the access to mining pools will be restricted to honest miners. The main contributions of this research work are:

- The proposition of PoolCoin, a reputation based blockchain that manages reputation of miners and pool managers.
- The proposition of a trust model inspired from the job market signaling model.
- Optimization of model's parameters, in order to detect malicious miners and differentiate them from honest miners.
- Providing a machine learning module that allows the Pool Manager to predict the real computing capacity of miners.

2 Related Works

Rewards provided to miners incite dishonest nodes of the blockchain to attack miners. For instance, in selfish mining attack, a strong attacker increases its revenue at the expense of others nodes [10]. In fact, in the selfish mining strategy a group of miners keeps its discovered blocks private, to intentionally generate forks in the chain. Thus, the group develops a longer branch in the public chain, kept private. The selfish miners reveal their private blockchain, only when the length of public branch approaches the public branch's length [11].

Individual miners tend to join mining pools in order to secure stable benefits [12]. Thus, mining pools represents a valuable target for attackers. In addition to existing attacks on blockchains, mining pools are faced to special attacks such as the distributed denial-of-service (DDoS) attack and the block withholding (BWH) attack.

Johnson et al. [7] studied the motivation behind the DDoS attack against Bitcoin mining pools. Their study showed that, there is a greater incentive to attack a large mining pool than a small one. In addition, they observed that mining pools larger than a threshold are subject to economically motivated attacks, while pools smaller than the threshold are not. The authors extended existing game theory models, by including defense cost. In their study they concluded that launching DDoS attacks depends on the failure probability of DDoS attacks. In addition, they stated that the main goal of a mining pool is to increase its computing power [7].

In the BWH attack, malicious miners don't submit full proof of work used by the pool manager to add a valid block and claim its reward. Bag and Sakurai proposed

in [4] a solution to the BWH attack, by introducing a special reward for the miner that shared the full proof of work.

The selection of miners is crucial for the security of the blockchain. Yahiatene and Rachedi proposed in [13] a blockchain based framework to secure vehicular social networks and a distributed algorithm named Distributed Miners Connected Dominating Set (DM-CDS), where the selection of miners is based on a trust metric and network parameters.

In a recent research work [9], Tang et al. proposed a reputation based mechanism that incites miners to behave honestly. For a blockchain composed of J miners and I mining Pool, a pool manager M_i is selected randomly. The latter will choose a reputation access threshold v_0, such that only miners with reputation value greater than v_0 can join the mining pool. The reputation evaluation of a miner j is based on the Pool Manager's satisfaction.

Recently Yu et al. in [14] proposed a new blockchain named RepuCoin that uses a new consensus protocol named Proof-of-Reputation, where the validation of new blocks doesn't rely only on the miner's instantaneous power. Indeed, the validation relies on the miner's reputation, computed based on two metrics: the miner's contribution and the miner's work regularity. RepuCoin is designed to counter miners with high computational power. Furthermore, when a miner misbehaves, RepuCoin mark that as a negative contribution, which involves a decrease in the integrated power (reputation) of the miner [14]. The consensus protocol Proof-of-Reputation, is also used by the GoChain blockchain [15].

In blockchain, Sharding separates validators into groups, in order to parallelize transactions processing. Recently, Huang et al. [16] proposed RepChain, a reputation based blockchain, where the reputation is introduced for Sharding system, in order to select shard leaders, enhance system's security, and establish a reputation based reward scheme.

Nojoumian et al. in [17], proposed a trust model based on the miner's lifetime in order to avoid Re-Entry attack, where a miner previously excluded from a mining pool comes-back with a new identity. The proposed trust model relies on the seniority of miners and includes the lifetime of miners in the miners' reputation.

The Distributed Trust Model (DTM^2) proposed by Haddadou et al. [8], provides a reputation based trust model, that incite network's nodes to cooperate honestly. The trust model of DTM^2 is inspired from the job market signaling model. Indeed, to send a message, a node should pay a cost that will be covered by the reward, if the message is considered correct. Furthermore, to receive messages, receivers also pay a reception cost. By consequence, nodes are incited to send correct messages on the network.

3 Background

3.1 Software Defined Network (SDN)

Software defined network (SDN) embeds network policies on the top of the network. The SDN network is composed of two layers: the *control layer* and the *data plane*.

The policies and control mechanisms are handled by the control plane, whereas the data forwarding is performed at the data plane [18]. The concept of programmable networks raised in mid-1990s, in order to allow modifications in the control logic of network devices. Two of the most significant early ideas were OpenSig (Open Signaling) working group and Active Networking initiative. *OpenSig* focused on Asynchronous Transfer Mode (ATM) networks, where the control plane was separated from the data plane. Signaling between the planes was performed through an open interface. *Active Networking* exposes resources of network nodes through a network API, allowing operators to execute arbitrary code [19].

As described in Fig. 1, an SDN network can be divided into Control Plane and Data Plane. The **Data Plane** contains programmable network devices and is responsible of forwarding packets according to assigned policies. The *Control Plane* can be formed of one or more SDN controllers, manages network devices from the Data Plane through south-bound APIs (such as OpenFlow). In addition, the Control Plane communicates with client applications through northbound interfaces [20]. Existing SDN architectures can be divided into three classes: *centralized single controllers*, *flat model distributed controllers*, *Hierarchical distributed controllers*, and *Hybrid distributed controllers* [18].

A centralized single controller (CSC) architecture is composed of a unique controller that maintains the global network state and controls the entire data plane of the network [18]. In CSC architecture, the entire network is solely managed by a unique SDN controller [20]. The flat model distributed controllers (FMC) architecture contains a number of controllers, where each controller manages only a sub-network [18]. The controllers coordinate between them through East/Westbound APIs, in order to maintain a global view network [21].

The hierarchical distributed controllers (HDC) architecture is composed of a root controller and local controllers. Each local controller manages a local network, while the root controller manages the global network and communicates with local controllers to manage the whole network [18]. The hybrid distributed controllers

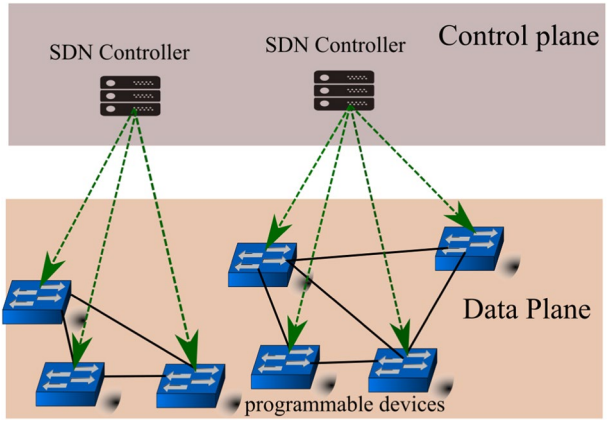

Fig. 1 Architecture of a software defined network

architecture combines between both flat and hierarchical architectures. In fact, at the top level, the network is divided into a number of root controllers, connected to each other through East/Westbound APIs, in order to maintain a global view of the network, in the same way that it is done with FMC architecture. In addition, like in HDC architecture, each root controller orchestrates the management of its sub-network, through its local controllers, in the same way that a root controller in HDC architecture acts as the leader of its local controllers [18].

3.2 Machine Learning

Machine learning is a field of computer science and a subset of artificial intelligence compromised of algorithms, able to be trained to perform a given task based on provided data. Different learning techniques exist: supervised, unsupervised, semi-supervised, and reinforcement learning [22]. The machine learning methods can also be divided into online and offline learning methods [23].

Supervised learning is a machine learning technique that takes a labeled dataset as input, in order to build a regression or classification model. Classification models predict discrete responses, while regression models are used to predict continuous responses. **Unsupervised learning** consists to find regularities in the dataset, such that certain patterns occurs more often than others and to learn to analyse what generally happens and what does not. In unsupervised learning, unlabeled datasets are used, such that no labels are given to the learning algorithm. **Semi-supervised learning** combines supervised and unsupervised learning, in order to generate a function mapping from inputs of both labeled and unlabeled data. The semi-supervised aims to classify some of the unlabeled data using labeled data set. **Reinforcing learning** is used for an autonomous agent that sens and acts in its environment. With reinforcement learning, the agent can learn to choose optimal actions to achieve its goals [24].

The term regression was introduced the first time by Francis Galton, where they studied the relationship between the tall of adult children and their parents, in their article entitled “Regression towards Mediocrity in Hereditary Stature”. Since that time, the term “‘regression” has been used to denote the relationship between two or more variables. If the variable X causes the variable Y, then X is called an independent variable and Y is called the dependent variable. In a simple linear regression model, the Y variable is linearly related to a single variable X [25]. Linear regression also generalizes to multiple independent variables x_1, x_2, … as described in Eq. 1, such that the dependent is predicted as the intercept C plus the of all independent variables, each multiplied by their corresponding coefficients plus noise [26]:

$$Y = C + \beta_1 \cdot x_1 + \cdots + \beta_2 \cdot x_2 + \epsilon \tag{1}$$

Another supervised machine learning regression technique is support vector regression (SVR), that represents an extension to the Support Vector Machine (SVM). In fact, in SVM classification, given a training labeled dataset a hyperplane is computed to correctly classify as many training samples as possible. New samples are then classified based on which side of the hyperplane they fall in the multidimensional

feature space. To find a good hyperplane, optimization is performed by maximizing the margin between the support vectors (i.e., the data points nearest to the hyperplane). For data regression, instead of finding a hyperplane that separates the training samples, SVR introduces an ϵ-insensitive loss function to compute a hyperplane such that the predicted response values of the training samples have at most an ϵ deviation from their observed ϵ (actual) response values hyperplane plus. The hyperplane plus ϵ defines an ϵ-insensitive band for computing generalization bounds for regression. Optimization is performed by minimizing the ϵ-insensitive tube to be as flat as possible while containing most of the training sample. The hyperplane is represented in terms of a few support vectors training samples that lie outside the boundary of the ϵ-insensitive tube. To allow SVR to handle nonlinear data, a kernel function transforms the original input data to a higher-dimensional space, referred to as a kernel space. There are several popular functions, such as linear kernels, polynomial kernels, radial basis function (RBF) kernels, and ANOVA RB kernels [27].

A neural network is formed of many connected neurons, arranged into multiple layers: an input layer, an output layer, and one or more hidden layers. The output y of the neuron having n inputs is calculated as shown by Eq. 2 and is activated if the signal strength reaches a certain threshold θ [28].

$$y = f(x), x = \sum_{i=1}^{n} w_i u_i \tag{2}$$

Deep learning models, can be classed into two categories: discriminative and generative models. Common discriminative models are multi-layer perceptron (MLP), Recurrent neural networks (RNN), and Convolutional neural networks (CNN). For generative models, we can cite: Restricted Boltzmann machine (RBM), Deep Boltzmann machines (DBM), and Deep Auto Encoders (DA) [29]

3.3 Evaluation of Machine Learning Algorithms

Different metrics are used to evaluate the performance of machine learning. Common classification metrics are: accuracy, confusion matrix, precision, and recall. In terms of regression, Root Mean Squared Error (RMSE) and quantiles of error are the major evaluation metrics [24].

3.3.1 Confusion Matrix

The confusion matrix is a table that records the number of instances in a dataset that fall in a particular category. The class label is a binary training set that can take two possible values: *positive* class and *negative* class. The number of positive and negative that a classifier predicts correctly are called true positives (TP) and true negatives (TN), respectively. The misclassified classes are known as false positives (FP) and false negatives (FN) [24]. Table 1 shows the confusion matrix.

Table 1 Confusion matrix

	Predicted	
	Positive	Negative
Actual		
True	TP	FN
False	FP	TN

3.3.2 Precision and Recall

The precision measure is the fraction of relevant instances among the retrieved instances [24]. In other terms, the precision is equal to the number of true positives divided by the total number of instances classified as positives (true positives and false positives):

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

The Recall measure (opposite of precision) is the ratio between the number of true positives and the sum of true positives and false negatives:

$$Recall = \frac{TP}{TP + FN} \tag{4}$$

3.3.3 F Measure

The F Measure (or F Score) is the harmonic average of precision and recall. F measures reaches its best value at 1 (perfect precision and recall) and worst at 0 [24]. The F Measure is calculated as follows:

$$F\ Measure = \frac{2}{\frac{1}{Recall} + \frac{1}{Precision}} \tag{5}$$

3.3.4 Root Mean Squared Error

For regression models, the Root Mean Squared Error (RMSE) is defined as the square root of the squared distance between the actual score and the predicted score. Let be y_i the real value of the ith data point and $haty_i$ its predicted value [24]. The RMSE metric is calculated as follows:

$$RMSE = \sqrt{\sum_i \frac{(y_i - \hat{y}_i)^2}{n}} \tag{6}$$

3.3.5 Quantiles of Errors

The RMSE metric takes mean of all the data points. Assuming the dataset contains an outlier, it will have a major impact on the average value. The effect of large outliers during the evaluation can be reduced by using robust metric called *quantiles errors* [24]. It considers *Median Absolute Percentage Error*:

$$MAPE = median\left(\left|\frac{y_i - \hat{y}_i}{y_i}\right|\right) \tag{7}$$

4 The PoolCoin Blockchain

In this section, we will present the proposed blockchain called PoolCoin, used to manage the miners' reputation. Thus, we will begin with a system overview of PoolCoin, followed by the trust model used by PoolCoin for miners' reputation.

4.1 System Overview

The proposed PoolCoin system is composed of Pool Managers, Miners, Mining Pools, and the PoolCoin's distributed ledger, shared between miners (Fig. 2). A *Pool Manager* receives transactions from the blockchain, then it distributes workload

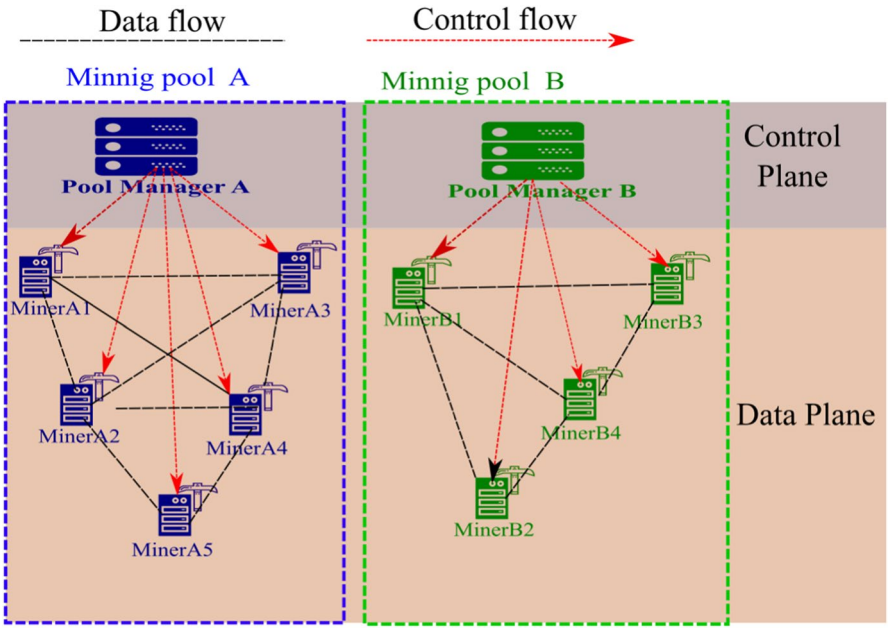

Fig. 2 System overview

between miners, in order to find a valid block. When a valid block is found, the Pool Manager distributes the corresponding reward among miners that participated in the mining process. The workload sent by the Pool Manager to miners represents the control flow PoolCoin. *The Miner* performs computations in order to find a valid block and send it back to the Pool Manager. The Miner sends shares (partial proofs of work) to the Pool Manager, in order to prove that it participated to the mining process. These shares will be used by the Pool Manager to determine the Miner's revenue. A *Mining Pool* is formed by the Pool Manager and its Miners, in order to ensure stable revenues. A Pool Manager uses the reputation of miners to accept them in its Mining Pool. The *PoolCoin's distributed ledger* is stored in a distributed manner among nodes of PoolCoin: Miners and Pool Managers. This distributed ledger stores information relative to miners' reputation, and is updated through data flows. The Pool Managers affect workload to miners that have sufficient trust score.

The proposed PoolCoin system uses a flat model distributed controller (FMC) SDN architecture. Each Pool Manager in PoolCoin plays the role of a SDN controller. Indeed, the Pool Manager sends control flows containing workloads that Miners should execute. In the Data Plane, the data flows exchanged between miners are used to maintain the distributed ledger of PoolCoin.

4.2 PoolCoin's Transactions

Compared to transactions of existing blockchain such as [30], transactions in the proposed Blockchain PoolCoin introduce two new fields: the reputation value RepValue and the miner's complaint. As shown in Fig. 3, a PoolCoin transaction is composed of six fields:

- From: represents the address of the sender.
- To: represents the address of the receiver.
- Value: The value of funds received by the miner indicated in the *To field* from the mining pool mentioned in the *From field*.
- RepValue: indicates the reputation value transferred from the sender to the receiver.
- Complaint: used to store complaints of miners against malicious pool manager
- ExtraInfo holds information such as: nonce, signatures, etc.

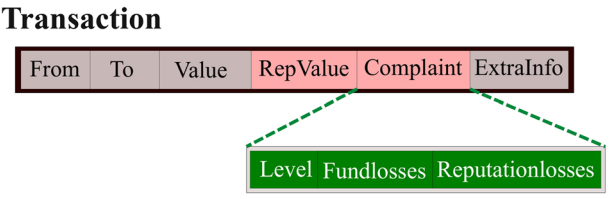

Fig. 3 Transactions of PoolCoin

The *RepValue* field is used for the management of the miners' reputation. In addition, the *Complaint* field allows miners to report misbehaving pool managers. The *Complaint* field is composed of: *Level*, *FundLosses*, and *ReputationLosses*. *Level* measures the critically of the complaint. *FundLosses* and *ReputationLosses* represent the losses caused by a pool manager. The pool manager is specified in the "*TO*" field of the transaction. Likewise, the miner that complained is specified in the "*FROM*" field of the la transaction.

4.3 The Mining Process in PoolCoin

In PoolCoin, the Pool Manager receives transactions from the Blockchain. Then, once it selects transactions to be included in the block, it broadcast the block's template to the miners usbscribed to its mining pool. To participate in the mining process, miners should pay a mining cost $C_{minining}$, using their trust score. Only miners that paid their mining cost are able to participate in the mining pool. By consequence, miners that their trust score is low, cannot participate to the mining pool.

The allowed miners will receive workload according to the paid mining cost. To proof that miners are honestly participating to the mining process, miners send Proof-Of-Works (PoWs) to the Pool Manager. We distinct two kinds of PoWs: Full PoWs and Partial PoWs. Full PoWs allows to the Pool Manager to claim the block reward, while the Partial PoWs, allows the Pool Manager to check that

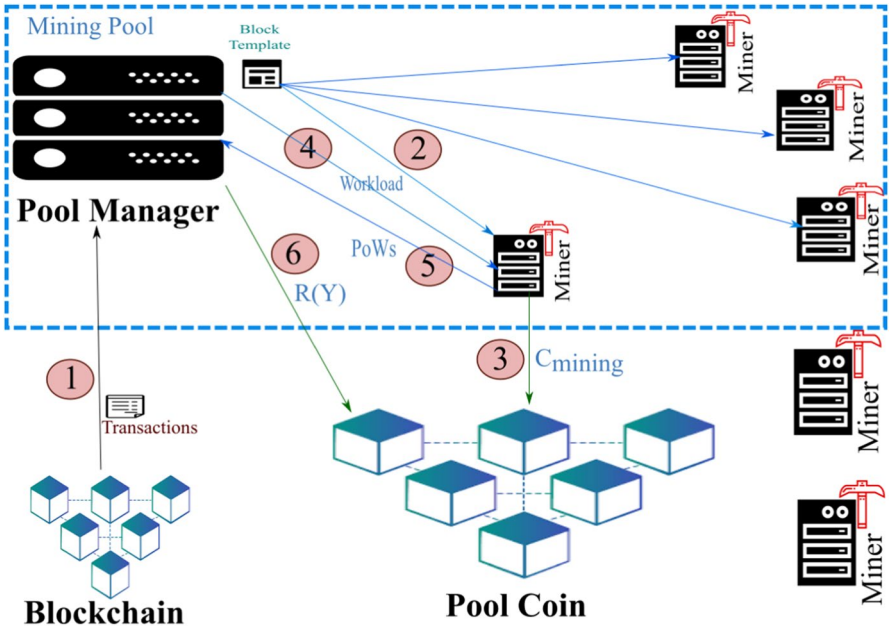

Fig. 4 Trust model of PoolCoin

miners are honestly participating to the mining process. The number of partial PoWs should be proportional to the received workload. At each Round, the Pool managers provides to miners a reward W(Y) according to their participation in the mining process. The Pool manager provides a financial reward R according to its reward system. The financial reward R can be transferred to any blockchain using the pegged sidechains technology [31]. If the miner is not satisfied by the obtained rewards, PoolCoin allow it to store a Complaint in the blockchain, using the data field of the transactions. Thus, PoolCoins provides protection mechanisms against malicious Pool Managers. Figure 4 shows the mining process using PoolCoin.

4.4 The Trust Model of PoolCoin

In this section, we will present the trust model used by PoolCoin to manage the reputation of miners. When a Miner wants to join a mining pool, he should be able to pay the mining cost C_{mining} from its trust score. According to his hash rate, the miner use a signal Y corresponding to its computing capacity (in terms of gigahash per second). The mining cost C_{mining} to be paid to the Pool Manager is proportional to the signal Y. However, the mining cost C_{mining} decreases when the miner's trust score θ increases. Indeed, the C_{mining} is equal to the used signal Y divided by the miner's trust score θ, powered to the inverse of the addition of the signal Y and the model's parameter α. Therefore, the mining cost C_{mining} is calculated as follows:

$$C_{mining} = \frac{Y}{\theta^{\frac{1}{Y+\alpha}}} \tag{8}$$

The Pool Manager affects workload to a miner according to the used signal Y. If the miner doesn't provide any share to the Pool Manager, the latter will not update its trust score θ. However, if the miner behaves honestly, the Pool Manager will reward it according to the number of shares and the used signal Y. More precisely, the number of partial shares (npPoWs) is divided by the signal Y to incite the miner to use its real capacity in the signal Y. In addition, to reward miners those find and provide full shares. Thus, the pool manager will provide to the miner the reward W(Y) calculated as described in the formula below. The parameter β is used to incite to share the solution with the Pool Manager when found.

$$W(Y) = \frac{npPoWs^{\frac{1}{\delta}}}{Y} + \beta \cdot nfPoWs \tag{9}$$

The trust score θ of a miner determines the reputation of the mining among different Mining Pool. This will allow a miner to transfer his trust score when it moves from one mining pool to another. When a miner joins the system, it obtains an initial score θ_{init}. The value of the initial score should as small as possible; in order to limit dishonest miners that uses several identities in mining.

4.5 Mining Pools Formation

To reward miners, Pool Managers should have sufficient reputation score. In fact, unlike [9] where Pool Managers are randomly selected among miners, Pool Coins requires that only miners with sufficient reputation are able to become Pool Managers. In fact, in PoolCoin Pool Managers should have sufficient funds to reward the miners. By consequence, miners before joining a mining pool, they should check that its Pool Manager has a reputation score sufficient to reward its miners.

The miners joins a mining pool based on the pool's rank, calculated based on the reputation of its pool manager and the complaints stored in the PoolCoin's transactions. To rank mining pools, for each transaction of PoolCoin containing complaints, the ranking algorithm (Algorithm 1) calculates the complaint score *cs*, based on the complaint's information. The algorithm updates the ranking of the Pool Manager based on θ_m (the trust level of the miner) and cs (the calculated complaint score). As described in Eq. 10, he complaint score (cs) is calculated based on the complaint's information: the level of the complaint (lv), the amount of fund losses (fl), and the amount of reputation losses (rl). In fact, the complaint score is equal total losses (funds losses and reputation losses) powered to the complaint's level.

$$cs = (fl + rl)^{lv} \tag{10}$$

The trust score of the miner θ is computed from PoolCoin's transactions containing where the miner appears in the field *From* or the field *To*. Indeed, θ is calculated from the distributed ledger of PoolCoin, by summing the reputation values of transactions where the miner is paying trust cost (transactions where the miner address appears in the *From* field) minus the sum of reputation values of transactions where the miner receives rewards (transactions where the miner address appears in the *To* field).

4.6 Signal Prediction

The pool manager distributes the workload among its miners, according to their claimed computing power. Indeed, when a miner joins a mining pool, it uses a signal Y corresponding to its computing capacity. The Pool Manager will affect workload based on the miner's signal Y. Malicious miners, can claim bigger computing capacity as a part of DDoS attack, in order to keep the mining pool from finding the proof of works. To protect the mining pool from attackers that pretends wrong computing capacity, the Pool Manager should check the signal value before affecting workload to miners. In order to achieve that, we provide a machine learning module, that can be used by the pool manager to estimate the real computing capacities of miners from its mining pool. The proposed machine learning module, estimates the real computing capacity (realY), according to a regression model, based on miner's information accessible to the pool manager, such as:

Algorithm 1 RankingPoolManagers

```
for each b in PoolCoin.getBlocks() do
  for each tx in b.getTransactions() do
    lv = tx.complaint.Level
    fl = tx.complaint.FundLosses
    rl = tx.complaint.ReputationLosses
    cs = complainteScore(lv, fl, rl)
    adrPM = tx.getTo()
    adrMiner = tx.getFrom()
    θ_m = getTrustScore(adrMiner)
    updateRanking(adrPM, θ_miner, cs)
  end for
end for
```

- PlAge: The age of the miner in the mining pool.
- MiAge: The age of the miner in PoolCoin.
- pPoWs: The number of partial PoWs submitted by the miner.
- fPoWs: The number of full PoWs submitted by the miner.
- City: The city of the miner.
- ISP: The internet service provider of the miner.
- trustScore: The trust score of the miner.
- claimedY: The claimed computing capacity.

5 Performance Evaluation

In this section we will study the performance of the proposed trust model. Thus, we considered several miners' behaviors and studied the performance of the proposed model. We also extracted optimal parameters of the model. In addition, we will study the performance of the signal prediction module, by considering different machine learning algorithms.

5.1 Experimental Setup

We developed a program in Python that simulates the PoolCoin blockchain, in order to evaluate the trust model. Thus, we considered the trust model presented Section 4.4. To study the reactivity of our system, we considered exclusion time of miners, expressed in terms of number of mining pool required to exclude the miner from the mining process. Likewise to optimize the model's parameters (Section 5.3), we considered the exclusion time of miners. In order to study the reliability of the proposed model (Section 5.5), we considered the evolution of the miners' trust score.

To study reactivity of our system, we observed the exclusion time according to the variation of the initial trust score θ_{init}. The model's parameter α is equal to 0,

the other parameters β and δ are assigned to 1. We considered different scenarios of Miners behaviors and for each scenario, we observed the evolution of miners' trust score during ten mining rounds. We observed the score of each miner after each mining Round R_i, while R_0 corresponds to the initial state before mining.

The first scenario **Honest Miner** corresponds to a honest miner. The second scenario **BWH Malicious Miner** corresponds to a malicious miner that didn't share the solution of the puzzle (the proof of work accepted by the network). In other terms, it corresponds to a malicious miner that performs a BWH attack. The third scenario **Absolute Malicious Miner** corresponds to a malicious miner that didn't contribute to solve the computational puzzle. The other scenarios **Alternate n% Malicious Miners** alternate between honest and malicious behaviours, where n represents the percentage of the malicious behavior. We considered three such scenarios: **Alternate 80% Malicious Miners**, **Alternate 50% Malicious Miners**, and **Alternate 20% Malicious Miners**

5.2 Results Analysis

In order to study the impact of the choice on the efficiency of the reputation system of PoolCoin, we varied the initial score in order to get optimized value for the initial score $\theta_{initial}$. The results shown in Fig. 5, shows the evolution of the exclusion time (in terms of number of rounds) according to the variation of the initial trust score.

As shown in Fig. 5, $\theta_{initial} < 1$, all miners are excluded in the first round. For other values of $\theta_{initial}$, Absolute Malicious Miners are rejected, while other miners are not detected. We observe that $\theta_{initial}$ values between 1 and 2, provide optimized exclusion time (2 mining rounds). In order to keep the initial trust score as small as possible, we choose 1 as optimal value of $\theta_{initial}$. However, the considered model

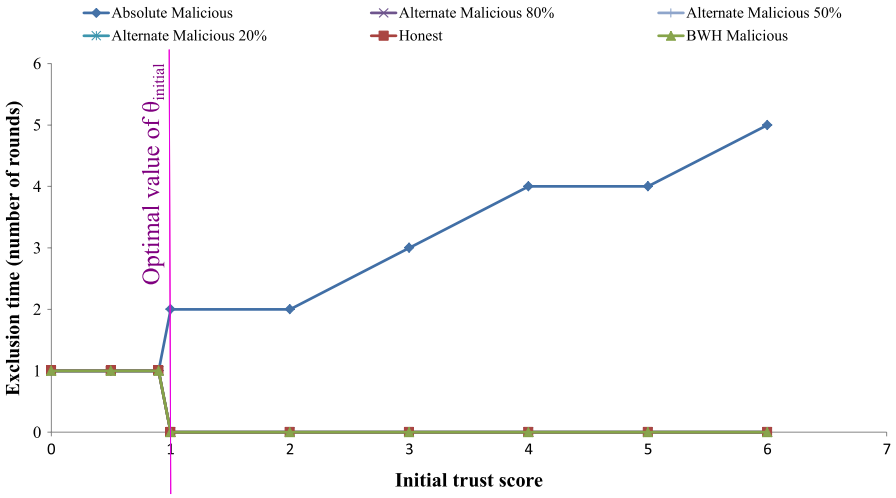

Fig. 5 Impact of initial trust score

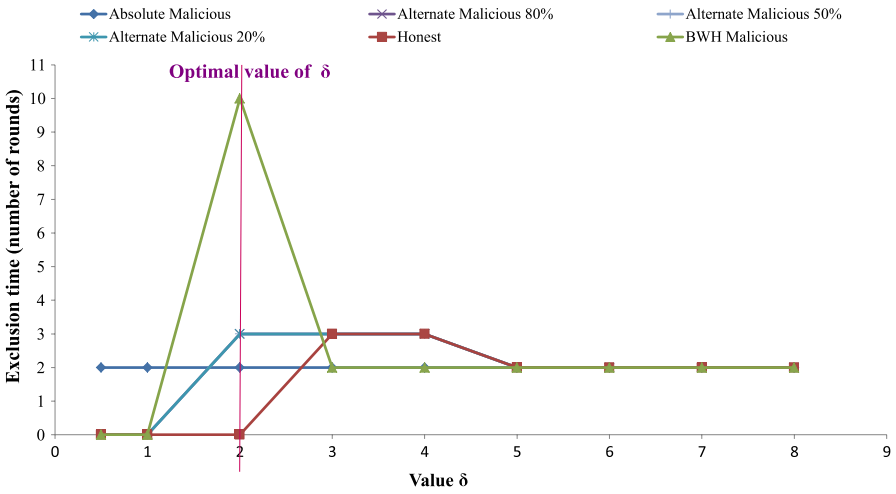

Fig. 6 Impact of parameter δ on the execution time

parameters should be optimized in order to detect: Alternate Malicious Miners and BWH Malicious Miner.

5.3 Parameters Optimization

In order to enhance the accuracy of our model, we will optimize the value of the model's parameters. By consequence, we will consider the optimal value of the initial trust (1), and extract optimal values of the model's parameters: α, β, and δ.

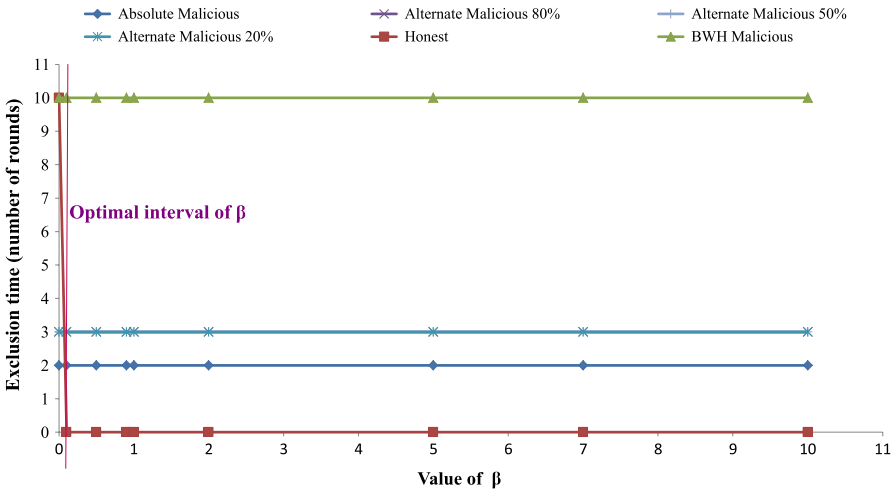

Fig. 7 Impact of parameter β on the execution time

5.3.1 Optimized Value of the Parameter δ

To extract the optimized value of the parameter δ, we considered the optimized value of the initial score (1). In addition, the other model's parameters α and β are assigned to 0 and 1, respectively. The evolution of the miner's exclusion time according to the value of the parameter δ, is shown in Fig. 6. We note that for values greater than 2, the Honest Miner is excluded from the mining pool. In addition, other values of the parameter δ, don't detect some dishonest miners, namely: BWH Malicious Miner and Alternate 20% Miner. By consequence, we conclude that the optimal value of the parameter δ is 2.

5.3.2 Optimized Value of the Parameter β

Using the optimal values of $\theta_{initial} = 1$ and the parameter $\delta = 2$, we will extract the optimal value of the parameter β. Figure 7 shows the behavior of the model according to the variation of the parameter β. We remark that, for values of the parameter β less than 0.1, the Honest Miner is excluded from the mining pool. However, for values of the parameter β greater than 0.1, the exclusion time is constant. Thus, we choose the value 0.1 as the optimal value for the parameter β.

5.3.3 Optimized Value of the Parameter α

In order to optimize α, we considered: $\theta_{initial} = 1$, $\delta = 2$, and $\beta = 0.1$. As shown in Fig. 8, the values of the parameter α greater than 0.02 exclude the Honest Miner. In addition, for values less than 0.02, the exclusion times of misbehaving miners are constants. By consequence, any value of the parameter α less than 0.02 is optimal. Thus, we consider the value 0.02 as the optimal value of the parameter α.

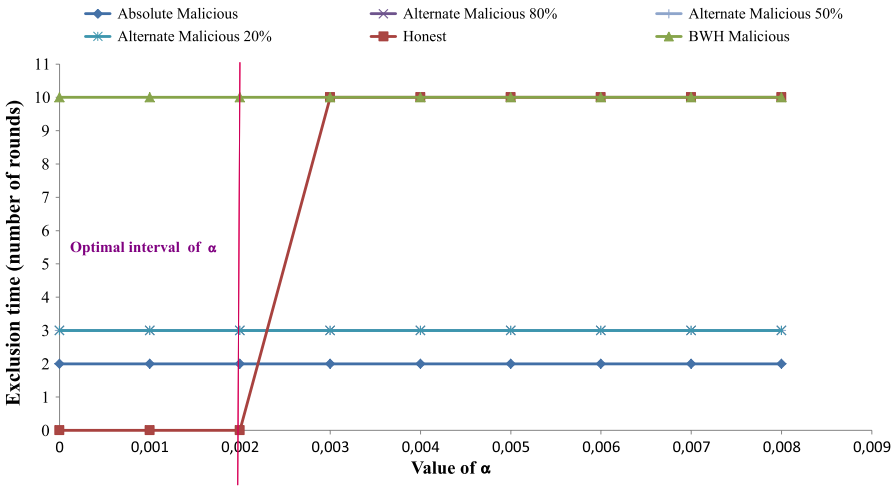

Fig. 8 Impact of parameter α on the execution time

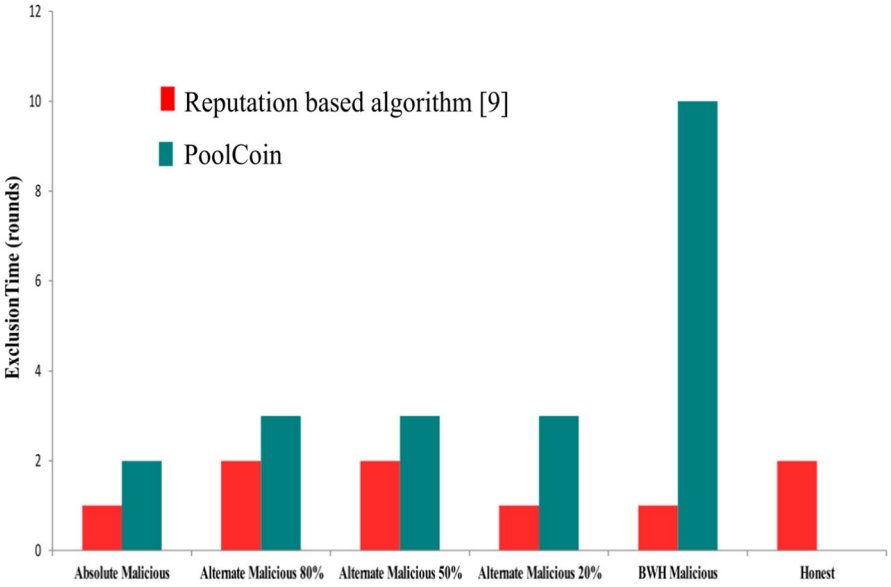

Fig. 9 Comparative study: PoolCoin vs. trust model of [9]

5.4 Comparative Study

We compared the proposed trust model of PoolCoin to the trust model [9]. For the trust model of PoolCoin, the optimal values of model parameters' was used($\alpha = 0.02$, $\beta = 0.1$, $\delta = 2$, and $\theta_{initial} = 1$). For the trust model of [9], we considered the reputation threshold $v_0 = 0.2$ and the fluctuation threshold $\sigma_0 = 0.1$. Thus, we compared the exclusion times (expressed in terms of number of rounds) between the PoolCoin and the trust model of [9], according to several miners' behaviors, as shown in Fig. 9.

As shown in Fig. 9, the trust model of PoolCoin requires more time to exclude malicious miners, than the trust model of [9]. Furthermore, for BWH Attacks, PoolCoin requires ten rounds to exclude the BWH Malicious miner, while the latter is excluded from the mining pool after only one round, when the pool manager uses the trust model of [9]. However, the trust model of [9] exclude honest miners after two rounds, which involves the reduction of the mining pool's revenues. Unlike the trust model of [9], the trust model of the proposed PoolCoin, doesn't exclude honest miners, while detecting malicious ones.

The RepuCoin proposed by Yu et al. in [14], is based on a new consensus protocol, named Proof-of-Reputation. Unlike RepuCoin, the proposed PoolCoin is consensus protocol agnostic and can be integrated in existing blockchains such as Bitcoin, without any changes on their consensus protocol. Due to the economic impact of the modification of the consensus protocol of existing crypto-currencies, such as Bitcoin, the used of PoolCoin is more practical. Likewise, the reputation based blockchain RepChain proposed by Huang et al. in [16] is based on new consensus

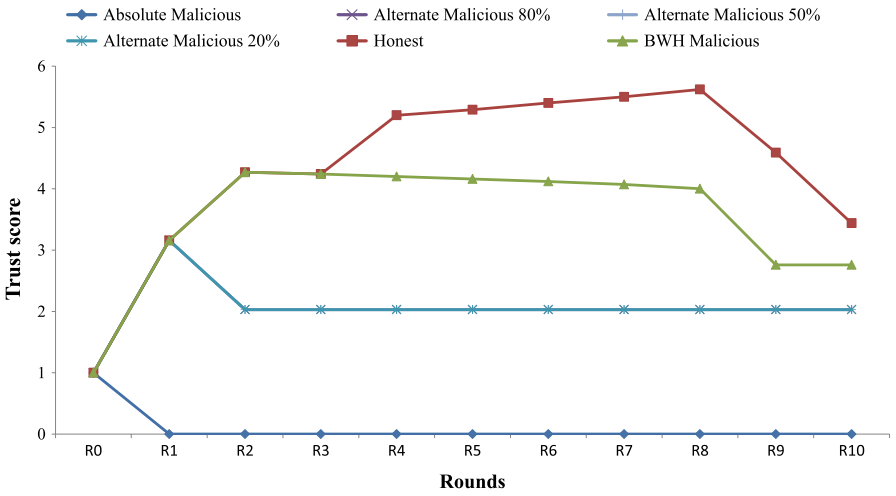

Fig. 10 The trust score evolution according to different rounds

protocols. In addition, RepChain is designed for Sharding systems and doesn't meets the mining pools context.

5.5 Reliability of the Model

To study the reliability of our proposed model we studied the evolution of the trust score of the different scenarios through the mining rounds. We considered the optimal values of model parameters': $\alpha = 0.02$, $\beta = 0.1$, $\delta = 2$, and $\theta_{initial} = 1$. Figure 10 shows the evolution of the miners' trust score during the ten mining rounds.

As shown by Fig. 10, the trust model of PoolCoin allows the Honest Miner to maintain a trust score allowing it to stay in the mining pool as long as it behaves honestly. When a miner alternate between honest and dishonest behaviors (Alternate Malicious n% Scenario), the trust score decreases in order to achieve a trust score that excludes the miner from the mining pool. The exclusion time is three mining rounds regardless of the percentage of the misbehaving. In addition, the trust model of PoolCoin is able to detect the BWH attacks at the 10th mining round (R_{10}), when the trust score of the BWH Malicious Miner achieves the score of 2.76.

5.6 Performance of the Signal Prediction Models

To evaluate the performance of the signal prediction machine learning module of Section 4.6, we generated a dataset composed of 20,000 mining history information, containing for each mining operation, the following mining information:

– PlAge: The age of the miner in the mining pool.
– MiAge: The age of the miner in PoolCoin.

– pPoWs: The number of partial PoWs submitted by the miner.
– fPoWs: The number of full PoWs submitted by the miner.
– City: The city of the miner.
– ISP: The internet service provider of the miner.
– trustScore: The trust score of the miner.
– claimedY: The claimed computing capacity.
– RealY: The real computing capacity.

We evaluated the performance of the signal prediction models, by comparing the accuracy of common regression models: linear regression, support vector regression (SVR), and multi-layer perceptron (MLP). For SVR model, we considered the radial basis function (RBF) kernel. For MLP model, we consider the MLP2 model composed of two hidden layers with eight neurons in each of them. In addition, we considered another MLP model, namely MLP3, composed of three hidden layers, with eight neurons on each. We applied these machine learning regression models to predict the miner's mining capacity (RealY), considering combinations of the most pertinent mining information: the miner's age (MiAger), miner's reputation (trustScore), and the claimed mining capacity (claimedY). To evaluate the performance of each model, we considered the model's training time and the Median Absolute Percentage Error (MAPE) accuracy metric. Training time's results are shown in Fig. 12, while accuracy results are shown in Fig. 11.

As shown by Fig. 11, the accuracy of signal prediction depends on the considered mining information. In fact, for instance, when considering the claimed mining

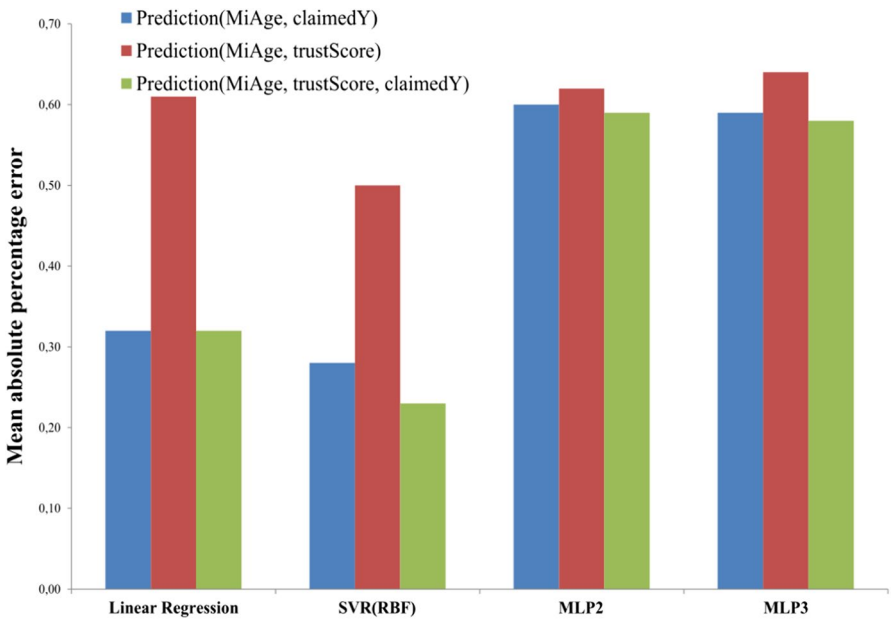

Fig. 11 Accuracy of signal prediction models

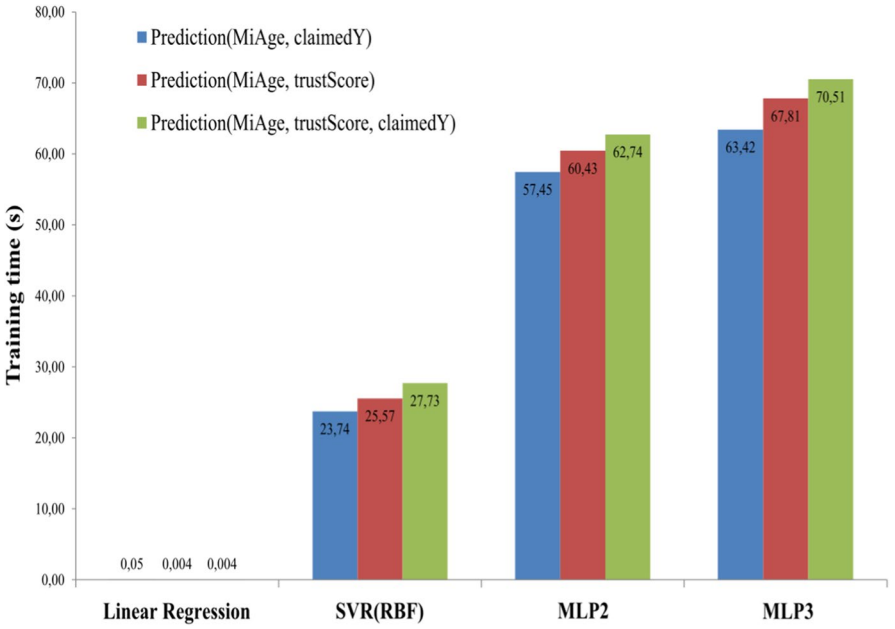

Fig. 12 Training time of signal prediction models

capacity (claimedY) with the miner's age (MiAger), all the models provides more accuracy and less error. In addition, Fig. 11 shows that SVR model is more accurate than other models. However, according to training times the linear regression model is the most efficient. Since, the mining time is crutial for mining pools and since the accuracy of the linear regression model is good (MAPE = 0.32), the linear regression model should be adopted by pool managers for signal prediction, considering the miner's age and the claimed mining capacity.

6 Security Analysis

The proposed trust model of PoolCoin detects five profiles of misbehaving miners: BWH Malicous, Absolute Malicous, Alternate Malicious 80%, Alternate Malicious 50%, and Alternate Malicious 50%.

The proposed trust model of PoolCoin protects Pool Managers against malicious miners that try to perform a DDoS attacks. In fact, if all the miners that coordinate their efforts in order to perform a DDoS attack, behave as Absolute Malicious Miner, the trust model of PoolCoin will detect them and exclude them from the mining pool. Likewise, if the miners alternate between honest and dishonest behaviours, they will be also detected and excluded from the mining pool. Furthermore, the detected miners will be denied from all the mining pools using PoolCoin. In addition, the Pool Manager is able to predict the real miner's computing capacity, to detect malicious miners that claim bigger computing capacity.

The trust model of PoolCoin detect and exclude from the mining pools, miners that fellow the BWH Malicious Miner's behavior. By consequence, mining pools are protected against the well know BWH attack. Compared to the solution proposed in [4], the proposed solution doesn't require the affectation of a special financial reward to the miner that share the full PoW. In addition, as the contrary to the solution of [4], the trust model is able to detect the miner's that performed the BWH attack and exclude it from the mining pool.

The proposed trust model of PoolCoin is able to detect malicious miners, in order to exclude them from the mining pool. Furthermore, miners that perform malicious attacks are excluded from all the mining pools of the blockchain. By consequence, miners are dissuaded from attacking mining pools. However, the trust model of PoolCoin requires that miners uses Trust Platform Modules (TPMs), in order to avoid Re-Entry attacks, where malicious miner use a new identity to be accepted in the system.

As described in Section 4.5, in PoolCoin only miners with high reputation are able to become Pool Managers. Thereby, unlike [9], a malicious miner is not able to become a Pool Manager. Furthermore, miners are able to report misbehaving Pool Managers, thanks to the complaint transactions stored in the distributed ledger of PoolCoin.

The proposed PoolCoin protects the Pool Manager from malicious miners based trust scores. In fact, the trust model of PoolCoin, obliges miners to pay mining costs, in order to ensure that miners participating in the mining pool hold sufficient trust scores. Thereby, the PoolCoin's trust model is dissuasive for attackers, in dint to the payment of mining costs. In addition, miners are incited through rewards provided by Pool Managers to honest miners. Furthermore, since transactions between miners and Pool Managers are decentralised in PoolCoin, honest miners can complain against Pool Managers that don't pay their fees. By consequence, based on these information, miners can compute the reputation of Pool Managers, in order to protect themselves from Misbehaving Pool Managers.

7 Conclusion and Future Works

Mining pools are mainly faced to two kinds of attacks: BWH attacks and DDoS attacks. In the present paper, we proposed PoolCoin a blockchain that deals with reputations of miners. The PoolCoin blockchain provides a trust model that protects mining pool against misbehaving miners. The trust model of PoolCoin is inspired from the job market signaling model. The parameters of the trust model were optimized in order to exclude only misbehaving miners. The optimized parameters of the trust model efficiently protect mining pools from attacks such as DDoS and BWH attacks. Furthermore, PoolCoin allows also to evaluate the Pool Managers' reputation, based on the complaints stored by miners in the blockchain. As future work, we aim to integrate PoolCoin and its turst model in BitCoin Blockchain. To detect malicious miners that claim bigger computing capacity, we also provided a machine learning module that alows the Pool Managers to estimate the real miners' capacities based on its historical mining information.

In order to detect *Re-Entry* attacks, where malicious miner attack the mining pool using different identities, PoolCoin relies on Trusted Platform Modules (TPMs). As a future work, we will integrate the miner's lifetime in our trust model, in order to incite miners to use unique identity.

References

1. Nakamoto, S.: Bitcoin: a peer-to-peer electronic cash system (2008)
2. Dinh, T.T.A., Liu, R., Zhang, M., Chen, G., Ooi, B.C., Wang, J.: Untangling blockchain: a data processing view of blockchain systems. IEEE Trans. Knowl. Data Eng. **30**(7), 1366–1385 (2018)
3. Wang, X., et al.: Survey on blockchain for Internet of Things. Comput. Commun. **136**, 10–29 (2019)
4. Bag, S., Sakurai, K.: Yet another note on block withholding attack on bitcoin mining pools. In: Information Security, pp. 167–180 (2016)
5. Werner, S.M., Pritz, P.J., Zamyatin, A., Knottenbelt, W.J.: Uncle traps: harvesting rewards in a queue-based ethereum mining pool, 070 (2019)
6. Qin, R., Yuan, Y., Wang, F.-Y.: A novel hybrid share reporting strategy for blockchain miners in PPLNS pools. Decis. Support Syst. **118**, 91–101 (2019)
7. Johnson, B., Laszka, A., Grossklags, J., Vasek, M., Moore, T.: Game-theoretic analysis of DDoS attacks against bitcoin mining pools. In: Financial Cryptography and Data Security, pp. 72–86 (2014)
8. Haddadou, N., Rachedi, A., Ghamri-Doudane, Y.: A job market signaling scheme for incentive and trust management in vehicular ad hoc networks. IEEE Trans. Veh. Technol. **64**(8), 3657–3674 (2015)
9. Tang, C., Wu, L., Wen, G., Zheng, Z.: Incentivizing honest mining in blockchain networks: a reputation approach. IEEE Trans. Circuits Syst. II Express Briefs **67**(1), 117–121 (2019)
10. Sapirshtein, A., Sompolinsky, Y., Zohar, A.: Optimal selfish mining strategies in bitcoin. In: Financial Cryptography and Data Security, pp. 515–532 (2017)
11. Eyal, I., Sirer, E.G.: Majority is not enough: bitcoin mining is vulnerable. Commun. ACM **61**(7), 95–102 (2018)
12. Liu, X., Wang, W., Niyato, D., Zhao, N., Wang, P.: Evolutionary game for mining pool selection in blockchain networks. IEEE Wirel. Commun. Lett. **7**(5), 760–763 (2018)
13. Yahiatene, Y., Rachedi, A.: Towards a blockchain and software-defined vehicular networks approaches to secure vehicular social network. In: IEEE Conference on Standards for Communications and Networking (CSCN) **2018**, 1–7 (2018)
14. Yu, J., Kozhaya, D., Decouchant, J., Esteves-Verissimo, P.: Repucoin: your reputation is your power. IEEE Trans. Comput. **68**(8), 1225–1237 (2019)
15. Proof of reputation node Archives—GoChain GoChain. https://gochain.io/tag/proof-of-reputation-node/. Accessed 20 Mar 2020.
16. Huang, C., Wang, Z., Chen, H., Hu, Q., Zhang, Q., Wang, W., Guan, X.: Repchain: a reputation based secure, fast and high incentive blockchain system via sharding. arXiv preprint arXiv :1901.05741 (2019)
17. Nojoumian, M., Golchubian, A., Njilla, L., et al.: Incentivizing blockchain miners to avoid dishonest mining strategies by a reputation-based paradigm. In: Science and Information Conference. Springer, Cham, pp. 1118–1134 (2018)
18. Saraswat, S., Agarwal, V., Gupta, H.P., et al.: Challenges and solutions in Software Defined Networking: a survey. J Netw Comput Appl **141**, 23–58 (2019)
19. Foukas, X., Marina, M.K., Kontovasilis, K.: Software defined networking concepts, software defined mobile networks (SDMN): Beyond LTE Network Architecture, p. 21. (2015)
20. Rawat, Danda B., Reddy, Swetha R.: Software defined networking architecture, security and energy efficiency: a survey. IEEE Commun. Surveys Tutor. **19**(1), 325–346 (2016)
21. Oktian, Yustus Eko: Distributed SDN controller system: a survey on design choice. Comput. Netw. **111**, 100–111 (2017)
22. Levy, M.: 15-Machine learning at the edge. In: Oshana, R., Kraeling, M. (eds.) Software Engineering for Embedded Systems, 2nd edn, pp. 549–601. Newnes, London (2019)

23. Krestinskaya, O., James, A.P.: Learning algorithms and implementation. In: James, A.P. (ed.) Deep Learning Classifiers with Memristive Networks: Theory and Applications, pp. 91–102. Springer International Publishing, Cham (2020)
24. Shobha, G., Rangaswamy, S.: Chapter 8-Machine learning. In: Gudivada, V.N., Rao, C.R. (eds.) Handbook of Statistics, vol. 38, pp. 197–228. Elsevier, Amsterdam (2018)
25. Toğaçar, M., Ergen, B., Cömert, Z.: A deep feature learning model for pneumonia detection applying a combination of mRMR feature selection and machine learning models. IRBM (2019)
26. Hope, T.M.H.: Linear regression. In: Machine Learning. Academic Press, pp. 67–81 (2020)
27. Zhang, F., O'Donnell, L. J.: Support vector regression. In: Machine Learning, pp. 123–140. Academic Press (2020)
28. Yang, X.-S.: 8—Neural networks and deep learning. In: Yang, X.-S. (ed.) Introduction to Algorithms for Data Mining and Machine Learning, pp. 139–161. Academic Press, London (2019)
29. Ferrag, M.A., Maglaras, L., Moschoyiannis, S., Janicke, H.: Deep learning for cyber security intrusion detection: approaches, datasets, and comparative study. J Inf Secur Appl **50**, 102419 (2020)
30. Akira, S.: Ethereum—The Next Generation of Cryptocurrency: A Guide to the World of Ethereum. CreateSpace Independent Publishing Platform, Scotts Valley (2018)
31. Back, A., Corallo, M., Dashjr, L., Friedenbach, M., Maxwell, G., Miller, A., Poelstra, A., Timón, J., Wuille, P.: Enabling blockchain innovations with pegged sidechains, 72. http://www.opensciencereview.com/papers/123/enablingblockchain-innovations-with-pegged-sidechains (2014)