World Wide Web (2022) 25:1085–1102
https://doi.org/10.1007/s11280-021-00944-w

EMERGING BLOCKCHAIN APPLICATIONS AND TECHNOLOGY

ELM-based data distribution model in ElasticChain

Dayu Jia[1] · Junchang Xin[1,2] · Zhiqiong Wang[3,4] · Han Lei[5] · Guoren Wang[6]

Received: 18 May 2020 / Revised: 17 May 2021 / Accepted: 30 August 2021 /
Published online: 25 February 2022

Abstract
Blockchain technology is becoming familiar to the public, along with the widespread use of cryptocurrency. The blockchain protocol requires that full nodes need to save the complete blockchain data, which limits the joining of resource-constrained nodes. A small number of full nodes will reduce the decentralize and security of system. Elasticchain was proposed in 2018 to solve this problem by saving fragments of the entire blockchain in reliable nodes. However, Elasticchain does not give an effective method to evaluate the reliability of nodes. If the fragmented data is stored in unreliable nodes, such as malicious tampering, are often not online or the latency is too high, the security of blockchain system will be seriously impacted. Therefore, in this paper, we propose an ELM-based method to comprehensively evaluate node reliability, and the blockchain system distributes the fragmented data to reliable nodes for storage. In the new method, ELM is used as a classifier to select reliable nodes because the ELM has a higher performance of training and classification compared to other machine models. Moreover, in ELM classifier five novel evaluation features are considered: the security, the trustworthiness, the activeness, the stability and the communication costs. Finally, the experimental results on synthetic data demonstrate the accuracy and efficiency of the optimized data distribution model.

Keywords Blockchain · Node reliability · Classification · ElasticChain · Extreme learning machine

1 Introduction

The first blockchain was conceptualized by Satoshi Nakamoto in 2008 [17]. The goal of Blockchain technology is to create a decentralized open environment to store information, execute transactions and perform functions [26]. With the increasing popularity of digital

This article belongs to the Topical Collection: *Special Issue on Emerging Blockchain Applications and Technology*
Guest Editors: Rui Zhang, C. Mohan, and Ermyas Abebe

✉ Junchang Xin
xinjunchang@mail.neu.edu.cn

Extended author information available on the last page of the article.

encryption currency such as Bitcoin [7], Ethereum [3] and DCEP (Digital Currency Electronic Payment) proposed by the People's Bank of China, blockchain technology is gaining more and more attention and development.

In blockchain systems, there are two kinds of nodes [27]. One is a full node, which contains a complete copy of the blockchain, and the other is a light node, which is just a client to access the data from full nodes. One of the bottlenecks in the current blockchain is that many nodes who want to participate in the blockchain system do not have enough storage space to be full nodes. When there are a small number of full nodes in the system, the decentralization and security of the system are insufficient.

ElasticChain [12] was proposed to solve the bottleneck. In [12], the entire blockchain data is fragmented and stored in relatively stable nodes. However, the node reliability evaluation method is very simple in [12]. It only detects the integrity of the data in storage nodes and the number of responses of the nodes. These two features cannot fully describe the reliability of nodes. Many other factors affect the reliability of a node, such as the performance of nodes, the length of time the nodes stay in the system, and the size of the data saved by the nodes, etc.

Challenges: Some methods have been proposed to improve the node reliability evaluation in ElasticChain, such as [11]. However, there are two shortcomings in the work of [11]. One shortcoming is that the calculation process of some evaluation features is relatively simple and only a small number of factors are considered. As a result, the current calculation results of these features are unrepresentative. The other is that the features of evaluating node reliability are still incomplete. Some important features are not taken into account. These two shortcomings greatly reduce the accuracy of classifying reliable nodes. If the blockchain data is stored in some unreliable nodes, which may exist problems such as node downtime, excessive latency, etc., the security of data will be seriously impacted.

Our contributions: This paper first proposes an optimized data distribution model for the ElasticChain. The Extreme Learning Machine (ELM) method [8] is used to be the classifier in this model. ELM [8, 28] is one of the machine learning models. We can produce reliable, repeatable decisions and uncover hidden insights through learning from historical relationships and trends in the data [18] by using machine learning methods. We use the ELM method instead of other machine learning methods because the ELM classifiers have a good performance in training and classification [9] (the details are in Section 3.3).

Second, we design a comprehensive evaluation method of node reliability. In the method, we define five evaluation indicators: the security, the trustworthiness, activeness, stability and communication costs of storage nodes. According to this evaluation standard, we can accurately classify truly reliable nodes and save blockchain data in them.

Finally, we conduct extensive experiments to demonstrate the efficiency and effectiveness of the optimized data distribution model based on the synthetic data.

This paper extends a preliminary work [11] in the following aspects. First, we analyze the basic theory and the advantage of ELM method in detail, and then propose the optimized data distribution model to further improve the accuracy of reliable node classification. Second, we add new features of node reliability evaluation (such as the communication costs between nodes) and redefine the incomplete features (such as the security of nodes, node activeness, etc.) in the new optimized model. Moreover, compared with [11], we add two sets of experiments to verify the efficiency of the new model and double the experiment to prove the effectiveness of this model.

Paper organization The remainder of the paper is organized as follows. Section 2 reviews the related work on the technique and application background of blockchain. Section 3 introduces the ElasticChain model and ELM. Section 4 introduces the architecture of the optimized data distribution model and the strategies of feature selection. Section 5 reports experimental evaluation. Finally, conclusions are presented in Section 6.

2 Related work

In this section, we review some related work on the technique and application background of blockchain.

In [21], the research directions in blockchain data management and analytics are detailed. Four topics were mentioned: leverage existing capabilities of mature data and information systems, enhance data security and privacy assurances, enable analytics services on blockchain as well as across off-chain data, and make blockchain-based systems active-oriented and intelligent. In [4], BlockBench is proposed, which is the first evaluation framework for analyzing private blockchains. It serves as a fair means of comparison for different platforms and enables a deeper understanding of different system design choices. The results on Ethereum, Parity and Hyperledger Fabric demonstrate that these systems are still far from displacing current database systems in traditional data processing workloads.

Wang et al. [22] and Li et al. [15] and many models improve the query speed of data in the blockchain. Wang et al. [22] presents ForkBase, a storage engine specifically designed to provide efficient support for blockchain and forkable applications. By integrating the core application properties into the storage, ForkBase not only delivers high performance but also reduces development effort. Li et al. [15] proposes an effective model for analyzing Ethereum data, called EtherQL. EtherQL provides highly efficient query primitives, such as range queries and top-k queries.

Xu et al. [25] and Kokoris-Kogias et al. [13] and many other systems increase the scalability of the blockchain. Xu et al. [25] describes a consensus unit-based storage scheme for blockchain systems, called CUB. CUB organizes some nodes into a unit, and a unit stores at least one copy of blockchain data. It addresses the high storage requirement in the wide usage of blockchain on various devices such as mobile phones or low-end PCs. Kokoris-Kogias et al. [13] presents OmniLedger, a novel scale-out distributed ledger that preserves long-term security under permissionless operation. It ensures security and correctness by using a bias-resistant public-randomness protocol for choosing large, statistically representative shards that process transactions.

In terms of node reliability, there is no relevant research to fully describe the reliability of blockchain nodes currently. Therefore, we give an evaluation method combined with machine learning in this paper.

3 Preliminaries

In this section, we introduce preliminary knowledge of ElasticChain and extreme learning machine. Furthermore, we propose the problem definition.

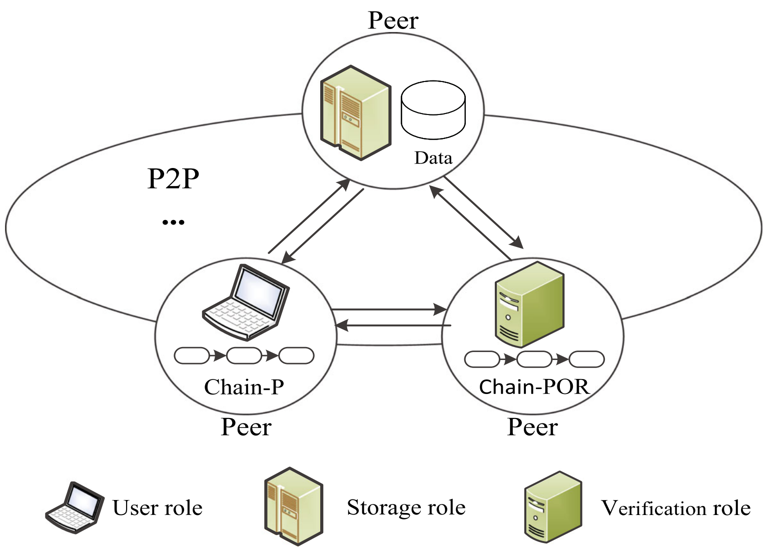


Figure 1 The nodes in ElasticChain

3.1 Storage node reliability verification in ElasticChain

According to the node reliability verification method [12], nodes in ElasticChain include three roles: the user node, the storage node, and the verification node, as shown in Figure 1. A node in a network would have one, two or three roles at the same time. User nodes are participants in the blockchain system. Blockchain operations, such as transactions, are completed between user nodes. And the fragmented blockchain data is stored in storage nodes. The role of the verification node is to provide reliable storage nodes for the user nodes.

The process of storage node reliability verification [12] is shown in Figure 2. Firstly, ElasticChain sets the same reliability values to each storage node. Then, the verification

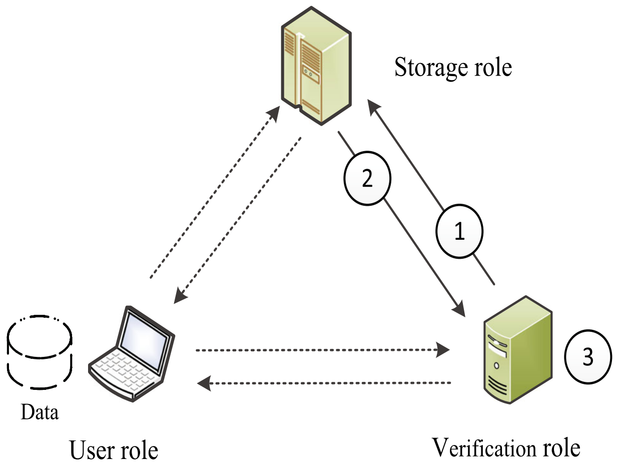


Figure 2 Node reliability verification

nodes check the reliability of data in storage nodes at every same period time. If the data in the storage nodes is complete, the reliability value remains unchanged. If the storage node data is modified or lost, the verification nodes will reduce its reliability value and store it in the POR (Proofs of Reliability) chain. The ElasticChain uses the reliability values of each storage node in the POR chain as a standard to select the highly reliable storage nodes. When the user nodes apply for storing data, the verification nodes provide the latest reliability value of storage node for the user nodes. Then, user nodes can select the most stable storage nodes to store the block data.

3.2 Problem definition

In ElasticChain, the reliability verification method for storage nodes U (U = $\{u_1$, u_2, ..., $u_i\}$) is relatively simple. It only considers the integrity of the data in the storage node and the number of responses. These two features cannot fully describe the reliability of nodes.

Threat model Data in storage nodes can be tampered with, and the malicious node may transmit false information to user nodes. Furthermore, the unreliable storage nodes are often offline or they drop frequently. In addition, the network where the storage node is located has a huge delay and cannot deliver data to users in time.

Therefore, these features checked in ElasticChain are not sufficient. In this paper, we propose an optimized data distribution model, which can comprehensively describe node reliability characteristics, and accurately classify nodes based on their reliability.

3.3 Extreme Learning Machine(ELM)

ELM has been originally developed for SLFNs and then extended to the "generalized" SLFNs where the hidden layer need not be neuron alike [5, 14]. Firstly, ELM randomly assigns the input weights and the hidden layer biases, and then analytically determines the output weights of SLFNs. It can achieve better generalization performance than other conventional learning algorithms at an extremely fast learning speed. Besides, ELM is less sensitive to user-specified parameters and can be deployed faster and more conveniently [9]. In recent years, a lot of research has been done on ELM. In [23], in order to effectively use the information from multiple attributes, an upper integral network is considered as a classification system by using multiple upper integral classifiers with a single layer neural network, and the learning mechanism of ELM is used to train the single-layer neural network. In [24], a novel Distributed Extreme Learning Machine (ELM*) based on a distributed MapReduce framework is proposed, which can learn massive data efficiently in parallel. In [2], an image classification method was proposed based on extreme k-means and EELM, and the method has superior performances on classification rate compared with other traditional methods based on experimental results. Compared with the naive implementation, the ELM-based implementation achieves much better performance.

The basic steps of ELM, as introduced in [8], are as follows. For n arbitrary distinct samples ($\mathbf{x}_j$, $\mathbf{t}_j$), where $\mathbf{x}_j = [x_{j1}, x_{j2}, ..., x_{jn}]^T \in \mathbf{R}^n$,and $\mathbf{t}_j = [t_{j1}, t_{j2}, ..., t_{jm}]^T \in \mathbf{R}^m$, standard SLFNs with L hidden nodes and activation function $g(x)$ are mathematically modeled as

$$\sum_{i=1}^{L} \beta_i g_i(\mathbf{x}_j) = \sum_{i=1}^{L} \beta_i g(\mathbf{w}_i \cdot \mathbf{x}_j + b_i) = \mathbf{o}_j \qquad (j = 1, 2, ..., N) \tag{1}$$

where $\mathbf{w}_i = [w_{i1}, w_{i2}, ..., w_{in}]^T$ is the weight vector connecting the ith hidden node and the input nodes, $\beta_i = [\beta_{i1}, \beta_{i2}, ..., \beta_{im}]^T$ is the weight vector connecting the ith hidden node and the output nodes, b_j is the threshold of the ith hidden node, and $\mathbf{o}_j = [o_{j1}, o_{j2}, ..., o_{jm}]^T$ is the jth output vector of the SLFNs.

The standard SLFNs with L hidden nodes and activation function $g(x)$ can approximate these N samples with zero error. It means $\sum_{j=1}^{L} ||\mathbf{o}_j - \mathbf{t}_j|| = 0$ and there exist β_i, $\mathbf{w}_i$ and b_i such that

$$\sum_{i=1}^{L} \beta_i g(\mathbf{w}_i \cdot \mathbf{x}_j + b_i) = \mathbf{t}_j \qquad (j = 1, 2, ..., N) \tag{2}$$

The equation above can be expressed compactly as follows.

$$\mathbf{H}\beta = \mathbf{T} \tag{3}$$

where $\mathbf{H}(\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_L, b_1, b_2, ..., b_L, \mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_L)$

$$= [h_{ij}] = \begin{bmatrix} g(\mathbf{w}_1 \cdot \mathbf{x}_1 + b_1) & g(\mathbf{w}_2 \cdot \mathbf{x}_1 + b_2) & ... & g(\mathbf{w}_L \cdot \mathbf{x}_1 + b_L) \\ g(\mathbf{w}_1 \cdot \mathbf{x}_2 + b_1) & g(\mathbf{w}_2 \cdot \mathbf{x}_2 + b_2) & ... & g(\mathbf{w}_L \cdot \mathbf{x}_2 + b_L) \\ \vdots & \vdots & \vdots & \vdots \\ g(\mathbf{w}_1 \cdot \mathbf{x}_N + b_1) & g(\mathbf{w}_2 \cdot \mathbf{x}_N + b_2) & ... & g(\mathbf{w}_L \cdot \mathbf{x}_N + b_L) \end{bmatrix}_{N \times L} \tag{4}$$

$$\beta = \begin{bmatrix} \beta_{11} & \beta_{12} & \cdots & \beta_{1m} \\ \beta_{21} & \beta_{22} & \cdots & \beta_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ \beta_{L1} & \beta_{L2} & \cdots & \beta_{Lm} \end{bmatrix}_{L \times m} \quad and \quad \mathbf{T} = \begin{bmatrix} t_{11} & t_{12} & \cdots & t_{1m} \\ t_{21} & t_{22} & \cdots & t_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ t_{N1} & t_{N2} & \cdots & t_{Nm} \end{bmatrix}_{N \times m} \tag{5}$$

H is called the hidden layer output matrix of the neural network and the ith column of **H** is the ith hidden node output with respect to inputs $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N$. The smallest norm least-squares solution of the above linear system is computed by

$$\beta = \mathbf{H}^{\dagger}\mathbf{T} \tag{6}$$

where $\mathbf{H}^{\dagger}$ is the Moore-Penrose generalized the inverse of matrix **H**. Then the output function of ELM can be modeled as follows.

$$f(\mathbf{x}) = \mathbf{h}(\mathbf{x})\beta = \mathbf{h}(\mathbf{x})\mathbf{H}^{\dagger}\mathbf{T} \tag{7}$$

Given a training set $\mathcal{N} = \{(\mathbf{x}_j, \mathbf{t}_j) \mid \mathbf{x}_j \in \mathbf{R}^n, \mathbf{t}_j \in \mathbf{R}^m, j = 1, 2, ..., N\}$, activation function $g(\mathbf{w}_i, b_i, \mathbf{x}_j)$ and hidden node number L, the pseudo code of ELM [9] is given in Algorithm 1.

Algorithm 1 ELM.

```
1  for $i = 1$ to $L$ do
2      Randomly assign input weight $\mathbf{w}_i$
3      Randomly assign input bias $b_i$
4  Calculate $\mathbf{H}$
5  Calculate the output weight $\beta = \mathbf{H}^{\dagger}\mathbf{T}$
```

We use the ELM method instead of other machine learning methods in this paper. The reason is that compared to other machine learning methods, the ELM classifiers have higher performance of training and classification [9]. For example [6], the testing accuracy of ELM is 99.14% in MNIST OCR dataset, which is 0.27%, 0.09%, 0.54% and 0.42% higher than Deep Belief Networks (DBN), Deep Boltzmann Machines (DBM), Stacked Auto Encoders (SAE) and Stacked Denoising Auto Encoders (SDAE), respectively. The training time of ELM is 281.37s, while the training time of the other methods is more than 17 hours. Based on the 3D Shape Classification dataset, the testing accuracy of ELM is 81.39%, which is 4.07% higher than the Convolutional Deep Belief Network (CBDN) method. The training time of ELM is 306.4s, while the CBDN method needs more than two days.

Moreover, different from Deep Learning which requires intensive tuning in multi hidden layers and hidden neurons, ELM theories show that hidden neurons are important but need not be turned (for both single hidden-layer feedforward neural networks (SLFNs) and multi-hidden-layer of networks) [20]. Therefore, the learning in ELM can simply be made without iteratively tuning hidden neurons, and this is one of the reasons ELM is efficient.

Compared to other traditional efficient models, online sequential learning can be achieved in ELM [16]. When new data is generated, traditional models need to put the new data and old data together and retrain. ELM can retain previous training experience and train new data based on current experience. Fast training can ensure that the training data of the model is complete and real-time. Therefore, the predicted result by ELM can be more accurate.

In ElasticChain, the fast classification will reduce the time for nodes to distribute duplicates when each block is generated and reduces the impact on blockchain system throughput. Thus, ELM is chosen as the classifier in our method.

4 The optimized data distribution model

In this section, we first describe the architecture of the optimized data distribution model. Then we introduce the features used to classify reliable nodes. After that, we propose an algorithm to describe the data distribution process.

4.1 Architecture

Figure 3 shows the architecture of the optimized model. It consists of three modules: the ElasticChain system module, the node feature extraction module and the ELM classifier module.

In the ElasticChain module, the verification nodes check the reliability of the storage node at the same time interval. There are 6 inspection results in total, and the results are saved in verification nodes. The verification nodes check the data volume and data integrity in the storage nodes, the number of disconnections and the online time of the storage nodes, the number of times the storage nodes have completed the verification work and the network environment of the system.

In the feature extraction module, we get five main features of the storage node: the security, the trustworthiness, the activeness, the stability and the communication costs by calculating the inspection results. These features can describe the reliability of a node completely.

Finally, in the classifier module, the reliable storage nodes are classified based on these five important features by using ELM. Then, user nodes save blockchain data in these reliable storage nodes. In ELM classifier, some of the nodes are sampled as training data. The sample nodes are input of the classifier module. They consist of two kinds of nodes, the reliable storage nodes, and the unreliable storage nodes. The way to create and sample training data are introduced in Section 4.3.

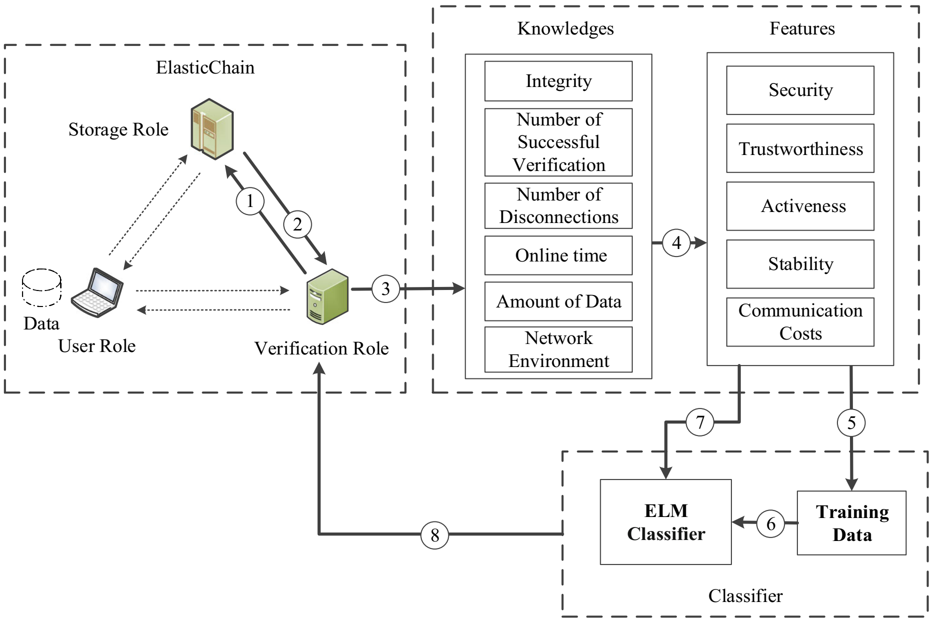


Figure 3 Architecture of the optimized data distribution model

Next, we describe the feature selection process in detail.

4.2 Feature selection

In ElasticChain system, multiple features will affect the reliability of storage nodes, and we choose five important features among them in this paper. The chosen features are the security of storage nodes (S), the trustworthiness of storage nodes (TR), node activeness (NA), node stability (NS) and communication costs (CC). The system can make a correct evaluation on the reliability of nodes in most cases by using these five features. Admittedly, other features will also have an impact on node reliability in some special environments. This problem is easy to solve because new features that have an impact on the environment can be added to the evaluation criteria without changing the structure of the model.

For the five features, each of them is calculated from two to four related features. The work of summarizing several related features into one feature can reduce the dimension of the classifier. The dimension reduction can reduce the calculation of classifiers and increase the speed of classification.

The storage nodes U ($U = \{u_1, u_2, ..., u_i\}$) will be detected at a fixed interval. The five features are updated after each detection. When there are i storage nodes in the system, i sets of feature data will be generated. Each set has five features, so the data sets (DS) of the storage node u_i can be expressed as a $5 \times i$ matrix:

$$DS = \begin{bmatrix} S_1 & TR_1 & NA_1 & NS_1 & CC_1 \\ S_2 & TR_2 & NA_2 & NS_2 & CC_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ S_i & TR_i & NA_i & NS_i & CC_i \end{bmatrix}_{5\times i} \tag{8}$$

Then, we define the five node reliability features.

Definition 1 (Security of storage nodes (S)) For the storage nodes $U = \{u_1, u_2, ..., u_i\}$, where i is the number of storage nodes in the blockchain system, they keep the duplicates of blockchain. When some of the storage nodes attempt to modify the data in the blockchain, these malicious nodes will tamper with the local data stored. It will have a great impact on the security of blockchain data. There is another situation where the blockchain data in the storage nodes is lost. This will cause great difficulties in data recovery. Therefore, the verification nodes $V = \{v_1, v_2, ..., v_j\}$, where j is the number of verification nodes in the blockchain system, check the integrity of blockchain data stored in the storage nodes at every same period time. The inspection results are the security of storage nodes, which are represented by S. It can be expressed as follows:

$$S = \sum_{k=1}^{K}(\omega_k - \mu_k a^{-(N-n)}) \tag{9}$$

where k is the kth inspection, and K is the total number of inspections. ω is the weight when data is complete, while μ is the weight when data is modified. If the blockchain data is integrated in the kth inspection, $\mu_k = 0$ and $\omega_k = \omega$. If the data is modified, $\mu_k = \mu$ and $\omega_k = 0$. Security is one of the most important evaluation indicators of the blockchain, and we do not tolerate the emergence of malicious nodes. Therefore the value of μ is much greater than ω.

Next, we explain the coefficient of μ_k. We assume a complete blockchain $B = \{b_1, b2, ..., b_N\}$, where N is the total number of blocks in the system. n ($1 \leq n \leq N$) is the number of blocks in which the modified data resides. Then, we know that the security of a block shows an exponential relation with its distance from the latest block [17]. Therefore, we add this coefficient ($a^{-(N-n)}$) for μ_k (here, $a > 1$). When the location of this modified data is near by the latest block, this data is easier to modify. Thus, if the newer data is modified in one storage node, we will give more weight to μ_k in order to the punishment and significantly reduce the security of this node.

Definition 2 (Trustworthiness of storage nodes (TR)) In ElasticChain, the user nodes just like light nodes that do not store the blockchain data. Operations of the user node, such as querying and validating data, must be completed by visiting the storage node. Each time a user node generates a transaction, the transaction needs to be verified. The querying process is as follows. First, the user node sends the transaction information to the storage nodes. Then, the result and the validation path of this transaction obtained by the query between storage nodes will be returned to the user node. Since the user node saves all block header data, the user node can verify the authenticity of this result by performing a Merkle check on the path information. The validation results can be regarded as the trustworthiness of storage nodes, which are represented by TR. It is stored in the verification nodes, and it can be expressed as follows:

$$TR = \sum_{p=1}^{P}(\psi_p - \phi_p) \tag{10}$$

where P is the total number of times the storage node completes the verification request initiated by the user nodes. ψ is the weight when the user node verifies the returned result successfully, while ϕ is the weight when the result is verified to be false. If the verification is successful in pth verification, $\phi_p = 0$ and $\psi_p = \psi$. If validation fails, $\psi_p = 0$ and $\phi_p = \phi$.

The security (S) and trustworthiness (TR) are two different evaluation indicators. The security (S) evaluates whether the data stored locally by a storage node is safe. The trustworthiness (TR) evaluates whether the storage node can honestly return the correct data to the user nodes.

Definition 3 (Node activeness (NA)) A storage node u_i cannot perform their job well in this case that it stays a few times in the network and only a few data stored in it. The system tends to choose storage nodes that can store large amounts of data and work online for a long time. Therefore, we have added this indicator, node activeness (NA), which can be expressed as follows:

$$NA_i = \xi_i A_i \times T_i \tag{11}$$

where i is the ith storage node, and ξ_i is the activeness weight of the ith node, and we set this weight to 1 in the paper. A_i is the amount of blockchain data in the ith storage node. T_i is the online time that the ith storage node stays in the network. The value of $A_i \times T_i$ shows the total amount of data that can be retrieved traceable in the network for the ith storage node.

Definition 4 (Node stability (NS)) In ElasticChain, each storage node u_i can not be online at all times. If the storage nodes are always offline, the user nodes cannot get the blockchain data in time, and the security of blockchain system will be reduced. Moreover, although an active node has a long online time, it is incompetent to finish its job when the online time is intermittent. Therefore, the number of times a node disconnects to the network affects the stability of this node. Here, the stability (NS) for a storage node u_i is given as follows:

$$Ns = \sum_{q=1}^{Q} \lambda_q f(t_q) \tag{12}$$

where q is the qth disconnection to the network, and Q is the total number of disconnections. λ_q is the weight of the qth disconnection. $f(t_q)$ is the time weight of the qth disconnection. $f(t_q) = \dfrac{k}{t_q}$, where $k > 0$. t_q is the time from the qth disconnection occurred to now. The function $f(t_q)$ decreases with the increase of t_q, which means that when the time of disconnection is far from now, this disconnection will have less effect on the current system.

Definition 5 (Communication costs (CC_i)) The blockchain technology is based on the P2P network, and the communication costs are one of the important indicators for selecting the node for communication. The fewer communication costs, the less network load will be generated. Many factors that affect communication costs, and we considered three important factors: the distance between nodes, link capacity and network conditions. The communication costs(CC_i) for the ith storage node can be expressed as follows:

$$CC_i = \sum_{r=1}^{R} \rho_r \frac{L_r}{R} \times C_r \times \sigma_r \tag{13}$$

where R means the storage node u_i connects R user nodes, and ρ_r is the weight of the rth link, and we set this weight to 1 in the paper. L_r is the distance between u_i and the rth user node, and the cost increases with distance. C is the weight of the network link capacity. C can be evaluated by the Shannon formula [19]. A small amount of cost requires sufficient capacity. σ_r is the evaluation index for network conditions. Due to the existence of a large number of evaluation indicators, the evaluation process is not described in detail in this paper. In practice, σ_r can be given by professional organizations.

4.3 Training ELM

After the feature extraction, ELM is selected as the classifier to learn security feature, trustworthiness feature, node activeness feature, node stability feature and communication costs feature. Each storage node responds to different user node requests, and the array of features is generated. This array is used as inputs to train his ELM. In the ELM-based classifier, each storage node can be classified into "reliable" class or "unreliable" class.

4.4 The optimized data distribution model

The pseudo-code of the main program of the optimized data distribution model for ElasticChain based on ELM is shown in Algorithm 2.

Firstly, the weight values(ω, μ, ψ, ϕ, ξ, λ, ρ) will be given according to system requirements. Secondly, the verification nodes V ($V = \{v_1, v_2, ..., v_d\}$) visit storage nodes $\{u_1, u_2, ..., u_i\}$ at every same period time and record its evaluation data (each ω_k and μ_k; each ψ_p and ϕ_p; each A_i and T_i; each t_q; each L_r, C_r and σ_r). Then, the optimized model calculates the feature values of these storage nodes according to the evaluation data. The features include the security (S), the trustworthiness (TR), the activeness (NA), the stability (NS), and the communication costs (CC). Next, these feature values are input into the trained ELM classifier and output the classification of node reliability. Verification nodes update and record the classification results (reliable node or unreliable node) in the ledger of the POR chain. Finally, when user nodes H ($H = \{h_1, h_2, ..., h_g\}$) generates new data, the verification nodes will provide reliable storage nodes for user nodes to store the new data.

Algorithm 2 The optimized data distribution model.

```
Input:  weight values($\omega$, $\mu$, $\psi$, $\phi$, $\xi$, $\lambda$, $\rho$)
Output: blockchain data distribution scheme
1  $V$ visit $\{u_1, u_2, ..., u_i\}$ at every same period of time;
2  $V$ record the data ($\omega_k$, $\mu_k$, $\psi_p$, $\phi_p$, $A_i$, $T_i$, $t_q$, $L_r$, $C_r$, $\sigma_r$) of $\{u_1, u_2, ..., u_i\}$;
3  $V$ calculates the feature values ($S$, $TR$, $NA$, $NS$, $CC$) of $\{u_1, u_2, ..., u_i\}$ according to
   the recorded data and weight values;
4  Enter feature values into the ELM classifier, and get the classification result of node
   reliability;
5  $V$ update and record the classification results in the ledger of the POR chain;
6  $V$ provide reliable storage nodes for user node $H$, when $H$ generates new data;
7  $H$ stores their data in these reliable storage nodes;
```

In the optimized data distribution model, the process of the user nodes reading the reliable storage nodes from the verification nodes and the process of the verification nodes classifying the reliable storage nodes through the ELM method are asynchronous. In other words, the verification nodes will evaluate the reliability of the storage nodes after a fixed period of time and then save the evaluation result locally. When a user node initiates a storage request, the verification nodes will provide the user nodes with reliable storage nodes in the evaluation result within this period. In this way, although the node feature extraction module and the ELM classifier module take some time, these two modules will not affect the working efficiency of the ElasticChain system.

For the efficiency of the node feature extraction module, the time complexity is proportional to the number of storage nodes. When a new storage node is added, six knowledges of this node need to be extracted. Therefore, the time complexity of the node feature extraction module can be expressed as $O(n)$. Here, n is the number of storage nodes.

For the efficiency of the ELM classifier module, the training complexity of the model is the same as the ELM model, the main computational cost comes from calculating the Lagrange multipliers [8]. ELM can get the calculation result based on Equation (37) in [8], where $\mathbf{H}^{\dagger}\mathbf{H}$ (size: $L \times L$) is used. The number of hidden nodes L can be much smaller than the number of training samples.

5 Evaluation

In this section, a series of experiments are implemented to verify the accuracy and efficiency of the optimized data distribution model in synthetic data. Experiments are carried out on the machine of Microsoft Windows 7, Intel Core i5 CPU, 3.20 GHz, and 16GB memory in java JDK 1.6. In the following, first, experiment settings are described in Section 5.1. Then we present and discuss the experimental results of the evaluation in Section 5.2.

5.1 Experiment settings

In our experiments, all experimental nodes are created using VMware Workstation 12.5.2. Each node has an ubuntu16.04 system with 300MB of memory and 1GB of hard disk space. We built ElasticChain and POR chain by use of the open-source Hyperledge fabric v0.6. The experiment established 10, 20, 30, 40 and 50 nodes, respectively. All nodes are storage nodes, user nodes and verification nodes. So, the feature values for a storage node is 9, 19, 29, 39 and 49 groups, and each group has five features.

We assign a value to each weight: $\omega_k = 1$, $\mu_k = 10$, $a = 1$, ψ_p =1, $\phi_p = 10$, $\xi_i = 1$, $\lambda_q = 1$, ρ_r =1. And we set multiple groups of parameters for the storage nodes through the control variable method. The parameters includes ω_k, μ_k, ψ_p, ϕ_p, A_i, T_i, t_q, L_r, C_r and σ_r. Therefore, we calculate the feature values (S, TR, NA, NS, CC) of this storage node by (9), (10), (11), (12) and (13), respectively.

For the dataset used in the experiment, the real dataset is the best choice. However, there is no real dataset suitable for the experiment. For example, the data in popular public blockchain systems, such as Bitcoin and Ethereum, contains a large amount of block information and transaction information, but the reliability information about the nodes is hard to find (e.g. the number of disconnections for a node, the online time of a node, etc.). Therefore, the node features (S, TR, NA, NS, CC) cannot be calculated by these real datasets, and synthetic data is used in this experiment. In the future, if we can get the node features in the public blockchain systems, the optimization model is suitable for most blockchain systems.

The synthetic data is obtained by multiple testing the running results of the storage nodes when we set different groups of parameters in the ElasticChain. In the fabric v0.6, the system will continuously initiate PBFT-based consensus [1] requests. The running results of our test are whether the storage nodes obtain the correct verification information and broadcast the information on time in each round of consensus. The broadcast information includes pre-prepare message, prepare message and commit message [1]. If the storage nodes broadcast the three messages authentically and promptly, this node is considered to be reliable in this round of consensus. In the data set, after multiple rounds of consensus, when a storage node is a reliable node in more than 90% of the rounds, we define this node as a reliable node. In other applications, a higher ratio (maybe 99%, 99.9%, or more) can be used to define reliable nodes according to requirements.

Next, we divide the data set into four groups. Each group of dataset has contains 50 sets of data and each has its unique characteristics. 60% of nodes are reliable in Dataset1 and Dataset2. In Dataset3, there are more reliable nodes (80%), and more unreliable nodes appear in Dataset4 (40% reliable nodes). Then, we use the Dataset1 to train the ELM classifier, and use Dataset2, Dataset3 and Dataset4 to test the performance. We set the number of hidden layer nodes as 10 when using the ELM classifier.

Furthermore, an SVM-based classifier is added to compare performance with the ELM classifier. The SVM-based classifier replaces the ELM classifier in the classifier module of the optimized model, and distinguishes reliable storage nodes. We choose a sigmoidal kernel function and set the penalty parameter as 10 for the SVM-based classifier.

5.2 Experimental results

We experimented on the accuracy, precision, recall and F1-measure of node reliability evaluation in different datasets(Dataset2, Dataset3 and Dataset4) by using the optimized data distribution model(ELM-OM), SVM-based optimized model(SVM-OM) and ElasticChain model when there are 10, 20, 30, 40 and 50 nodes in model.

The accuracy of a model can be expressed as follows:

$$Accuracy = (TP + TN)/(TP + FN + FP + TN) \tag{14}$$

where TP is True Positive, FP is False Positive, TN is True Negative, FN is False Negative. And the precision of a model can be expressed as follows:

$$Precision = TP/(TP + FP) \tag{15}$$

The recall of a model can be expressed as follows:

$$Recall = TP/(TP + FN) \tag{16}$$

The F1-measure of a model can be expressed as follows:

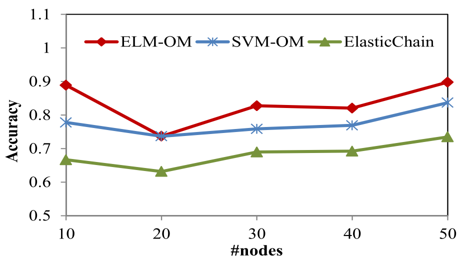
$$F1 - measure = 2 \times \frac{Precision \cdot Recall}{Precision + Recall} \tag{17}$$

Experimental results on Dataset2, Dataset3 and Dataset4 are shown in Figures 4, 5 and 6, respectively. We can get the following conclusions from these figures.

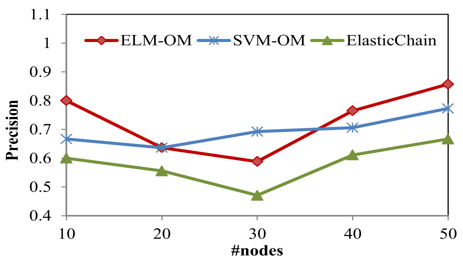
(1) Overall, the first conclusion we can draw is that the accuracy, precision, recall and F1-measure of node reliability evaluation in ELM-OM are higher than that in the SVM-OM when 10, 20, 30, 40 and 50 nodes exist in models. The reason is that ELM method has a higher performance of training and classification than the SVM method.

The second conclusion is that the accuracy, precision, recall and F1-measure of the node reliability evaluation in ElasticChain model are the lowest, and far below ELM-OM and SVM-OM. Because the ELM-OM and SVM-OM use the new evaluation strategy. The five features (security, trustworthiness, activeness, stability and communication costs) in new evaluation strategy can describe the reliability characteristics of a storage node in a comprehensive way.

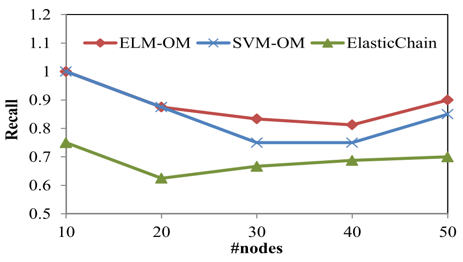
(2) Under the same dataset, except for 10 nodes, when the number of nodes is small, the values of the four evaluation indexes of the classification result are relatively low. The accuracy, precision, recall and F1-measure of node reliability evaluation in ELM-OM, SVM-OM and ElasticChain model become more and more higher with the increasing number of nodes



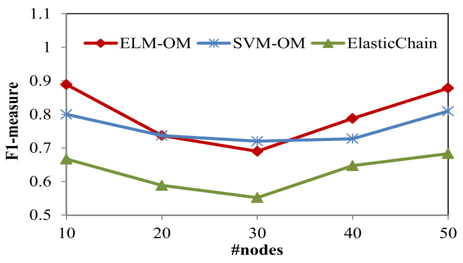
(a) Accuracy



(b) Precision



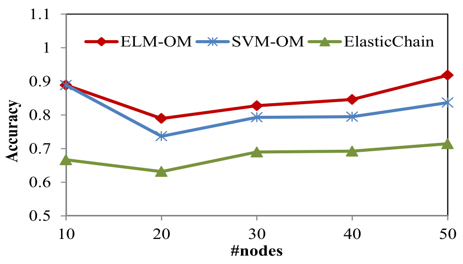
(c) Recall



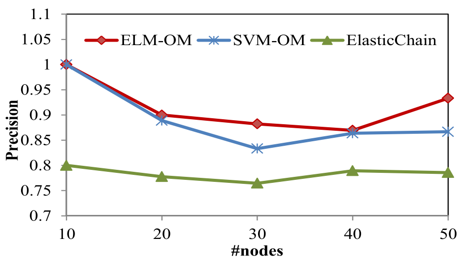
(d) F1-measure

Figure 4 Experimental results on Dataset2

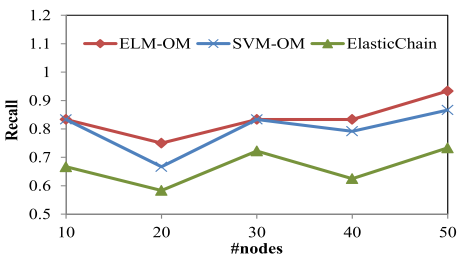
in models. It is because that there are few reliability evaluations for a node when the number of nodes is small, while the number of reliability evaluations increases as the number of nodes increases, then the classification of reliable nodes is more accurate.



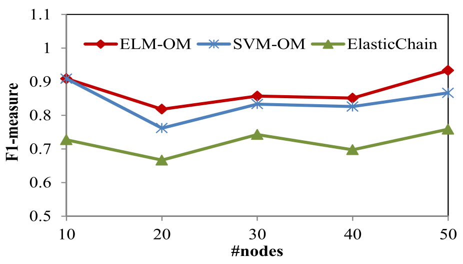
(a) Accuracy



(b) Precision

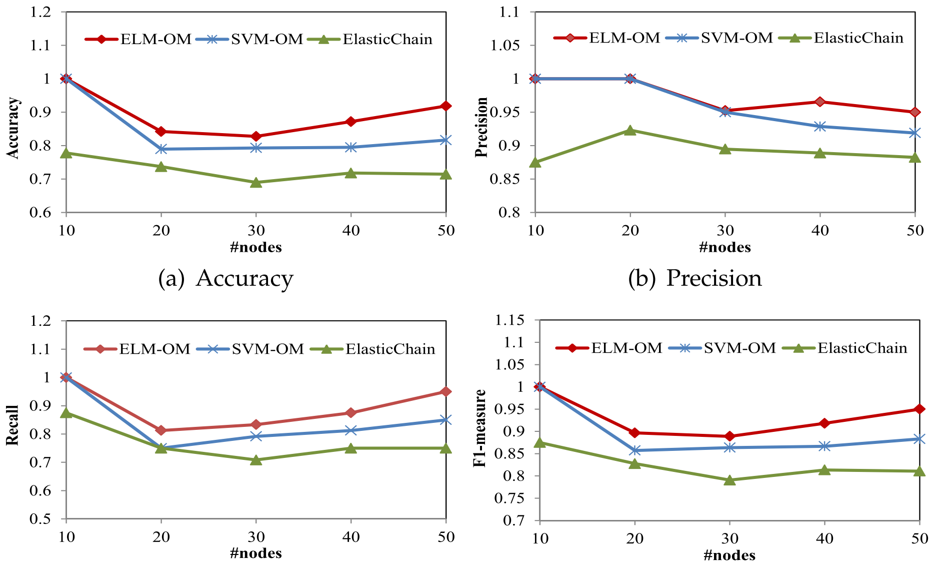


(c) Recall



(d) F1-measure

Figure 5 Experimental results on Dataset3



(a) Accuracy (b) Precision

(c) Recall (d) F1-measure

Figure 6 Experimental results on Dataset4

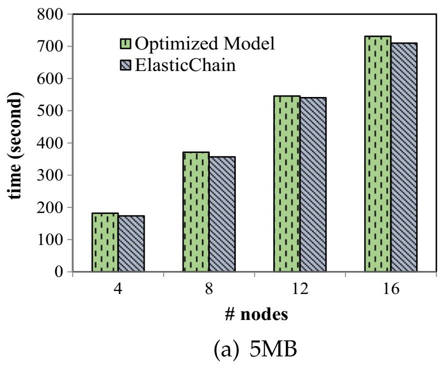
(3) When the number of nodes in the models is the same, the accuracy, precision, recall and F1-measure of the three models under Dataset4 are slightly higher than the values under Dataset3 and Dataset2. It is because that the proportion of reliable nodes is large in Dataset4, and the characteristics of nodes are obvious. Therefore, the classification results in Dataset4 are better.

Next, we tested the efficiency of the optimization model. We tested the processing time of the optimized data distribution model and ElasticChain when there are 4, 8, 12 and 16 nodes in systems. Both systems are implemented based on fabric v0.6, and the condition for fabric v0.6 to operate normally is that there are at most 16 nodes [4]. Therefore, we set the number of nodes as above.

Meanwhile, the nodes in both systems are set as reliable nodes. The reason why we did not choose unreliable nodes for testing is that the job of the optimized model is to select reliable nodes to store the fragmented data. Reliable nodes can respond to query requests in a timely manner and ensure that there are enough copies of data in the entire system.

The job of the optimized model will not affect the consensus process of ElasticChain. The reason is that when the optimized model and ElasticChain produce new blocks, the blocks will be confirmed based on the PBFT consensus. The condition for confirmation is that more than two-thirds of the participating nodes are honest nodes. The new block confirmation process and the long-term maintenance of the data by the storage node are two different stages.

In the experiment, we executed the $chaincode\_example02.go$ [10] transaction code 930 times and 1860 times, generating 5.00MB and 10.00MB data. 500KB of data will be stored as a shard, and each shard will be stored in at least 2 copies. The running times of the optimized model and ElasticChain are shown in Figure 7. The running is stopped recording after the new data is verified and written into the block.



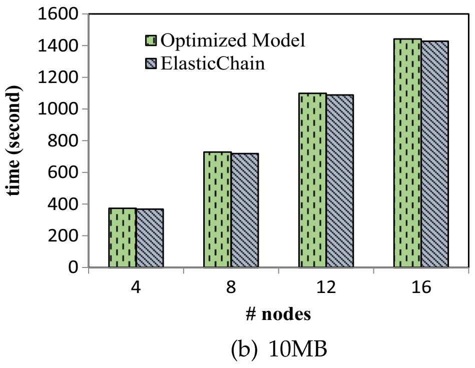


Figure 7 The running times of the optimized model and ElasticChain

From Figure 7, we can see that as the number of nodes increases, the running time of the two systems increases linearly. The reason is that a new block needs to be confirmed by more than 2/3 of the nodes in the PBFT consensus. The systems contain a large number of nodes and require a long confirmation time.

Meanwhile, the running times of the optimized model and ElasticChain are almost the same. The reason is that there is not much difference in the process of generating blocks between the two systems. However, in practical application, if the verification node frequently classifies the storage node and the verification node reaches the performance bottleneck, the block validation may be delayed. Therefore, we should set an appropriate interval time to classify storage nodes, rather than blindly set the interval time too small.

6 Conclusion

In our study, we present an optimized data distribution method for the ElasticChain, which combines the blockchain technology with the machine learning method. This method classifies the nodes according to their reliability by using the Extreme Learning Machine (ELM) and distributes the blockchain data in reliable nodes to increase data security. Moreover, we propose a new strategy to extract the node reliability in order to fully evaluate the reliability of the node. It includes five features, which are the security, trustworthiness, activeness, stability and communication costs of storage nodes. Finally, the experimental results on synthetic data demonstrate the accuracy and efficiency of the optimized data distribution model.

In the future, we will focus on studying the optimized method to achieve the safe distribution of data and extracting the other more features for the storage nodes in ElasticChain to improve the comprehensiveness of node evaluation. Furthermore, we will experiment with multiple machine learning algorithms and nature-inspired algorithms as classifiers. By analyzing and comparing the classification results, we find a more accurate classification method and save the blockchain data in high-reliability nodes.

Acknowledgements This research was partially supported by the National Natural Science Foundation of China (No. 62072089), the Fundamental Research Funds for the Central Universities (Nos. N2116016, N2104001, N2019007, N180101028, N180408019 and N2024005-2), and the Open Program of Neusoft Corporation (No. NCBETOP2102).

References

1. Castro, M., Liskov, B., et al.: Practical Byzantine fault tolerance. In: OSDI, pp. 173–186 (1999)
2. Cao, F., Liu, B., Park, D.S.: Image classification based on effective extreme learning machine. Neurocomputing **102**(2), 90–97 (2013)
3. Chen, T., Teng, H.u., Chen, J., et al.: DataEther data exploration framework for ethereum. ICDCS **2019**, 1369–1380 (2019)
4. Dinh, T.T.A., Wang, J., Chen, G., Liu, R., Ooi, B.C., Tan, K.-L.: BLOCKBENCH: a framework for analyzing private blockchains. SIGMOD Conference **2017**, 1085–1100 (2017)
5. Cui, D., Huang, G.uang.-B.in., Liu, T.: ELM based smile detection using distance vector. Pattern Recognit (79)356–369 (2018)
6. Extreme learning machine. https://www.ntu.edu.sg/home/egbhuang/ [DB/OL] [2020-10-27]
7. Frey, D., Makkes, M.X., Roman, P.-L., Taïani, F., Voulgaris, S.: Dietcoin: Hardening Bitcoin Transaction Verification Process For Mobile Devices. Proc. VLDB Endow. **12**(12), 1946–1949 (2019)
8. Huang, G.-B., Siew, C.K.: Extreme learning machine: RBF network case, 8th International conference on control, automation, robotics and vision. ICARCV, pp 1029–1036 (2004)
9. Huang, G.-B., Zhu, Q.-Y., Siew, C.K.: Extreme learning machine: Theory and applications. Neurocomputing **70**(1-3), 489–501 (2006)
10. Hyperledger/fabric. https://github.com/hyperledger/fabric/blob/v0.6/examples/chaincode/go/chaincode_example02/chaincode_example02.go. [DB/OL] [2021-4-10]
11. Jia, D., Xin, J., Wang, Z., Guo, W., Wang, G.: An optimized data distribution model for elasticchain to support blockchain scalable storage. In: International conference on extreme learning machine (ELM), pp. 76–85 (2018)
12. Jia, D., Xin, J., Wang, Z., Guo, W., Wang, G.: ElasticChain: support very large blockchain by reducing data redundancy. APWeb-WAIM **2018**(2), 440–454 (2018)
13. Kokoris-Kogias, E., Jovanovic, P., et al.: OmniLedger: a secure, scale-out, decentralized ledger via sharding. IEEE Symposium on Security and Privacy **583-598**(2018), 2018 (2016)
14. Li, C., Deng, C., Zhou, S., Zhao, B., Huang, G.-B.: Conditional random mapping for effective ELM feature representation. Cogn. Comput. **10**(5), 827–847 (2018)
15. Li, Y., Zheng, K., Yan, Y., Liu, Q., Zhou, X.: EtherQL: a query layer for blockchain system, database systems for advanced applications-22nd international conference. DASFAA 2017, pp 556–567 (2017)
16. Liang, N.-Y., Huang, G.-B., Saratchandran, P., Sundararajan, N.: A fast and accurate online sequential learning algorithm for feedforward networks. IEEE Trans. Neural Networks **17**(6), 1411–1423 (2006)
17. Nakamoto S.: Bitcoin: A peer-to-peer electronic cash system, Consulted (2008)
18. Pedregosa, F., Varoquaux, G., et al.: Scikit-learn machine learning in python. J. Mach. Learn. Res. **12**(10), 2825–2830 (2011)
19. Rioul, O., Magossi, J.C.: On Shannon's formula and Hartley's rule: beyond the mathematical coincidence. Entropy **16**(9), 4892–4910 (2014)
20. Tang, J., Deng, C., Huang, G.-B.: Extreme Learning machine for multilayer perceptron. IEEE Trans. Neural Networks Learn. Syst. **27**(4), 809–821 (2016)
21. Vo, H.T., Kundu, A., Mohania, M.K.: Research directions in blockchain data management and analytics. Proceedings of the 21th international conference on extending database technology, EDBT 2018, pp. 445–448 (2018)
22. Wang, S., Dinh, T.T.A., Lin, Q., et al.: ForkBase: an efficient storage engine for blockchain and forkable applications. Proceedings of the VLDB Endowment **11**(10), 1137–1150 (2018)
23. Wang, X., Chen, A., Feng, H.-M.: Upper integral network with extreme learning mechanism. Neurocomputing **74**(16), 2520–2525 (2011)
24. Xin, J., Wang, Z., Chen, C., Ding, L., Wang, G., Zhao, Y.: ELM*: distributed extreme learning machine with MapReduce. World Wide Web **17**(5), 1189–1204 (2014)
25. Xu, Z., Han, S., Chen, L.: CUB, a consensus unit-based storage scheme for blockchain system. 2018 IEEE 34th International Conference on Data Engineering, ICDE 2018, pp 173–184 (2018)
26. Zhang, R., Xue, R., Liu, L.: Security and privacy on blockchain. ACM Comput. Surv. **52**(3), 1–34 (2019)
27. Zhang, C., Xu, C., Xu, J., Tang, Y., Choi, B.: GEM2-Tree: A gas-efficient structure for authenticated range queries in blockchain. ICDE **2019**, 842–853 (2019)
28. Zeng, Y., Li, Y., Chen, J., Jia, X., Huang, G.-B.: ELM embedded discriminative dictionary learning for image classification. Neural Networks (123)331–342 (2020)

Affiliations

Dayu Jia[1] · Junchang Xin[1,2] · Zhiqiong Wang[3,4] · Han Lei[5] · Guoren Wang[6]

Dayu Jia
jiadayu@stumail.neu.edu.cn

Zhiqiong Wang
wangzq@bmie.neu.edu.cn

Han Lei
leih0002@e.ntu.edu.sg

Guoren Wang
wanggrbit@126.com

1 School of Computer Science and Engineering, Northeastern University, Shenyang, China

2 Key Laboratory of Big Data Management and Analytics, Shenyang, Liaoning Province, China

3 College of Medicine and Biological Information Engineering, Northeastern University, Shenyang, China

4 Neusoft Corporation (Research Center of Liaoning Promotion for Blockchain Engineering Technology), Shenyang, China

5 Nanyang Technological University, Singapore, Singapore

6 School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China