Computers and Electrical Engineering 104 (2022) 108421

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

# Intelligent contracts: Making smart contracts smart for blockchain intelligence☆

Liwei Ouyang [a,b], Wenwen Zhang [c,*], Fei-Yue Wang [b,d]

[a] School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China
[b] Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
[c] School of Computer Science, Shaanxi Normal University, Xi'an 710119, China
[d] Institute of Systems Engineering, Macau University of Science and Technology, 999078, Macau

ARTICLE INFO

Dataset link: https://archive.ics.uci.edu/ml/datasets/iris, http://yann.lecun.com/exdb/mnist/, https://image-net.org/challenges/LSVRC/2012/

*Keywords:*
Blockchain
Smart contracts
Blockchain intelligence
Decentralized artificial intelligence

ABSTRACT

With the deep integration of blockchain and Artificial Intelligence (AI), more and more blockchain-based AI tasks are accomplished using Smart Contracts (SCs) and create win-win solutions. That is, blockchain provides a trustworthy and decentralized data infrastructure for AI, and AI helps blockchain perform tasks requiring intelligence. Since these special SCs designed for blockchain-based AI tasks have different characteristics from the widely studied SCs designed for business logic, we name them Intelligent Contracts (ICs) for a focused study. In this paper, we systematically analyze ICs and propose a constructive framework for their construction and application. Specifically, we first formulate two construction modes of current ICs, including encoding AI models and scheduling AI collaboration. Then, we compare the characteristics of these two modes theoretically and experimentally as a reference for future mode selection. Finally, to extend the application of ICs and encourage AI-driven blockchain intelligence, we propose a technical route that helps blockchain autonomously respond to AI tasks through the dynamic and optimal configuration of ICs. Using typical AI tasks of classifying IRIS, MNIST, and ImageNet data sets as examples, we implement and thoroughly evaluate two modes of ICs on Ethereum. Based on the constructed ICs, we illustrate their optimal configuration and automatic response process. Experimental results demonstrate the effectiveness and feasibility of the proposed framework.

## 1. Introduction

Blockchain intelligence [1], the convergence of two disruptive technologies, blockchain and artificial intelligence, has attracted widespread attention recently as blockchain and AI can complement each other to revolutionize the coming digital transformation [2]. Intuitively, blockchain has the potential to provide AI with a trustworthy and decentralized data infrastructure, while AI can help blockchain extract valuable information and perform tasks that require intelligence, including learning, reasoning, planning, and problem solving. Compared to simply applying AI in the context of blockchain (e.g., data analysis on blockchain data), accomplishing AI tasks based on blockchain is considered a more profound integration of blockchain and AI, as well as an essential approach to facilitate decentralized AI and AI-driven blockchain intelligence [3]. AI-driven blockchain intelligence and AI-aided blockchain intelligence are two relative concepts, where AI-aided blockchain intelligence concentrates on optimizing blockchain through data analysis. And AI-driven blockchain intelligence expects blockchain to achieve the same intelligence as AI [4].

☆ This paper is for regular issues of CAEE. Reviews processed and recommended for publication to the Editor-in-Chief by Huimin Lu.

* Corresponding author.
*E-mail address:* 2021136@snnu.edu.cn (W. Zhang).

https://doi.org/10.1016/j.compeleceng.2022.108421
Received 16 August 2022; Received in revised form 22 September 2022; Accepted 12 October 2022
Available online 12 November 2022

Currently, blockchain-based AI tasks have led to many exciting advances. For example, blockchain-based AI marketplaces not only help share data and models while retaining ownership and privacy [5,6], but also help outsource or crowdsource learning tasks with fair incentives [7]. And blockchain-based distributed AI computing transforms traditional centralized AI into a semi-decentralized or decentralized architecture and helps eliminate single points of failure, incentivize trustless distributed collaboration, and innovate edge intelligence [8,9]. Moreover, since Smart Contracts (SCs) are the key technology enabling the programmable blockchain and developing SCs is much easier than building a dedicated blockchain from scratch [10], a growing number of proposals for blockchain-based AI tasks are implemented with the help of flexible and portable SCs. Examples include specifying collaboration protocols with SCs in blockchain-based AI marketplaces [11] and aggregating model weights with SCs in blockchain-based distributed AI computing [12]. SCs are gradually becoming the preferred intermediaries to accomplish blockchain-based AI tasks. And accordingly, SCs designed for blockchain-based AI tasks are becoming vital modules to facilitate decentralized AI and AI-driven blockchain intelligence.

However, statistical studies indicate that the existing widely studied SCs are mainly designed for business logic. He et al. [13] have analyzed millions of SCs deployed on Ethereum, the most prominent SCs platform, and found that most of these Ethereum SCs are highly homogeneous and mainly implement simple and similar control logic for tokens, games, Initial Coin Offering (ICO), and so on. It is clear that SCs designed for blockchain-based AI tasks have different functionalities than conventional SCs designed for business logic, which may lead to other characteristics and require a separate study. To make a distinction, this paper names this special type of SCs designed for blockchain-based AI tasks as Intelligent Contracts (ICs). Although ICs show a bright future, their current attempts are independent and fragmented, lacking systematic analysis. As a result, it remains unclear how to construct ICs from SCs, what their characteristics are, and how their applications contribute to AI-driven blockchain intelligence.

To fill this critical gap, we propose a constructive framework to guide the construction and application of ICs and conduct a systematic analysis. Specifically, we first analyze the current progress of ICs to formulate their construction modes. Then, we comprehensively compare different modes of ICs to show their characteristics. Finally, we elaborate on a novel technical route to encourage AI-driven blockchain intelligence through the configuration of the constructed ICs. In summary, our contributions are as follows:

(1) We formulate two modes to construct ICs: Mode 1 for encoding AI models and Mode 2 for scheduling AI collaboration. We explain their core ideas with representative research.

(2) We analyze the advantages and challenges of these two construction modes from theoretical and experimental perspectives as a reference for future mode selection. In particular, using typical AI tasks of classifying IRIS, MNIST, and ImageNet data sets as examples, we implement two modes of ICs on representative Ethereum blockchain, respectively, and evaluate their performance thoroughly. The experimental results demonstrate the effectiveness of the proposed framework.

(3) We propose maintaining an interaction and verification records-based trusted database in the form of a blockchain-based decentralized market for modular and reusable ICs and associated AI collaboration modules. Then, with the help of their dynamic and optimal configuration, blockchain can respond to different AI tasks, thus meeting the expectations of AI-driven blockchain intelligence. Based on the ICs constructed in experiments, we illustrate their optimal configuration and automatic response process to show our feasibility.

The rest of this paper is organized as follows: Section 2 reviews related work; Section 3 proposes the constructive framework and compares SCs and two modes of ICs; Section 4 demonstrates the effectiveness of the proposed framework; Section 5 concludes the paper.

## 2. Related work

In this section, we briefly review the basic concepts and recent advances of blockchain and SCs, distributed AI, and the crossover research on blockchain and AI.

### *2.1. Blockchain and smart contracts*

As the underlying technology of Bitcoin [14], blockchain is an append-only distributed ledger with chained data blocks maintained and shared by all nodes in a decentralized system [15]. The concept of Smart Contracts (SCs) is first proposed and defined by Nick Szabo as "a set of promises, specified in digital form, including protocols within which the parties perform on these promises" [16]. After Ethereum with a built-in Turing-complete programming language introduces the concept, SCs running on the blockchain become self-enforce and self-verify computer programs with preset conditional-response rules, that can accurately encapsulate and execute complex behaviors of distributed nodes. Benefiting from the decentralization, openness, transparency, and tamper-resistance inherited from blockchain, SCs help participants without mutual trust to trade and collaborate without any trusted third-party authority or intermediary, which significantly expands the application of blockchain.

As an emerging technology, SCs still have many key issues that restrict their developments, including security issues [17], performance issues [18], privacy issues [19] and legal issues [20]. In particular, the ideal SCs are expected to be on-chain smart agents for distributed participants. Whereas the current SCs are mainly designed as a series of control logic that can only be executed passively [21]. This intelligence gap restricts their widespread use, so there is an urgent need for crossover research on blockchain and AI.

### 2.2. Distributed artificial intelligence

The recent advances of AI in machine learning [22], deep learning [23] and reinforcement learning [24] have paved the way for many exciting applications in the field of robotics, computer vision and natural language processing. Most of these impressive results adopt centralized architectures that require large-scale collected data and high-performance computing resources. And once these data and resources are hacked or limited, they will face great decision-making risks. Thus, distributed AI with better parallelism, fault tolerance, and openness is increasingly attractive.

Traditionally, distributed AI has two research branches: distributed problem solving and multi-agent systems [25]. Distributed problem solving focuses on dividing a specific problem solving task among multiple cooperative and knowledge-sharing modules or nodes [26], while multi-agent systems focus on the knowledge, goals, skills, and planning of multiple agents, as well as their collaboration and competitive mechanisms [27]. Recently, with the popularity of the Internet of Things (IoT) and mobile computing, more and more distributed AI computing architectures have been proposed to leverage edge intelligence. Federated Learning (FL) [28] and edge computing [29] are two of the most representative ones. Their core idea is to transfer computing and communication resources from the centralized cloud or server to the edge nodes, thus providing edge users with faster service response, better data privacy, and fewer communication costs. Their advances highlight the advantages of distributed AI.

### 2.3. Crossover research on blockchain and artificial intelligence

The convergence of blockchain and AI can be mutually beneficial. Currently, their crossover research has two directions: blockchain-optimized AI and AI-optimized blockchain. In terms of blockchain-optimized AI, blockchain and SCs can serve as digital infrastructures to provide AI with credible and efficient resource sharing [30], data traceability [31] and collaboration incentive mechanisms [32]. On this basis, various distributed AI computing architectures can be architected for multiple AI tasks, thereby driving the transformation from centralized AI to distributed and decentralized AI. There have been some attempts at blockchain-based FL, including BlockFL [33], FLchain [34], and CREAT [35]. For AI-optimized blockchain, the existing work mainly uses AI to analyze blockchain and SCs data and extract valuable information for their benign developments, including the analysis of transaction patterns [36], contract codes [37] and cryptocurrencies [38]. Since these data analytics do not improve the intelligence of blockchain and SCs, some studies refer to them as AI-aided blockchain intelligence and point to the need for AI-driven blockchain intelligence. That is, with the help of AI, blockchain and SCs can have the capacity for learning, reasoning, planning, and problem solving in an autonomous manner [4]. Accordingly, this paper names such a blockchain that can autonomously respond to requests for accomplishing different AI tasks as an AI-driven blockchain.

To the best of our knowledge, while many researchers have reviewed the state-of-the-art crossover research on blockchain and AI [39], there is a lack of research specifically focused on ICs. For example, Pandl et al. [3] divide blockchain-based distributed AI computing into three categories: the computation within the blockchain, the computation in SCs, and the computation outside the blockchain, where the latter two categories can roughly correspond to our Mode 1's ICs featuring on-chain computing and Mode 2's ICs featuring off-chain computing. However, they only enumerate related work without in-depth analysis and comparison of the two modes. At the same time, how to achieve AI-driven blockchain intelligence is still an open problem with little progress. Thus, our constructive framework aims to comprehensively analyze ICs and provide new ideas for encouraging AI-driven blockchain intelligence.

## 3. Intelligent contracts

In this section, we explain the motivation of ICs, propose the constructive framework and compare SCs and two modes of ICs theoretically.

### 3.1. Why do we need intelligent contracts?

To answer this question, first, we examine the main functionalities of the current SCs from the available statistical studies. In the literature, SCs are mainly classified from the perspective of design patterns and application domains. Table 1 summarizes some representative taxonomies from these two perspectives. As can be seen, the taxonomies of design patterns aim to abstract the essential development structure of current SCs, while the taxonomies of application domains aim to reflect their high-level usage. All detailed categories listed in Table 1 show that the current SCs are mainly designed to implement domain-specific business logic. Oliva et al. [40] and He et al. [13] provide some more intuitive data. Oliva et al. [40] analyze 1.9 million Ethereum SCs deployed from July 2015 (inception of Ethereum) to September 2018, and find that 72.9% of the high-activity SCs are token SCs, and 5 of the top-10 SCs with the highest activity are currency exchanges. He et al. [13] analyze nearly 10 million Ethereum SCs deployed from July 2015 to December 2018, and find that over 96% of SCs have duplicates, and the most popular clusters of SCs are for tokens, ICOs, and Games. All these statistical studies indicate that the main functionalities of SCs at this time are tied to simple control logic.

However, a growing body of research has demonstrated the advantages of accomplishing AI tasks based on blockchain. For instance, Zhang et al. [44] propose a blockchain-based FL framework for device failure detection in industrial IoT to ensure client data privacy. Zhao et al. [7] propose a blockchain-based AI task outsourcing framework with integrity assurances and fair payments. And Sarpatwar et al. [45] propose a blockchain-based AI marketplace to trade data and models with guaranteed privacy and fairness. More importantly, because it is not easy to build a dedicated blockchain for each personalized AI task from scratch, more researchers

**Table 1**
The representative taxonomies of the current SCs.

| Research | Taxonomy | Perspective |
|---|---|---|
| Bartoletti and Pompianu [41] | Financial, Notary, Game, Wallet, Library | Application domain |
| | Token, Authorization, Oracle, Randomness, Poll, Termination, Time constraint, Math, Fork check | Design pattern |
| Wöhrer and Zdun [42] | Action and control, Authorization, Maintenance, Lifecycle, Security | Design pattern |
| Oliva et al. [40] | Games, Exchanges, Gambling, Finance, Property, Wallet, Governance, Media, Storage, Development, Identity, Social | Application domain |
| Hu et al. [43] | Game, Gambling, Exchange, Finance, High-risk, Social | Application domain |

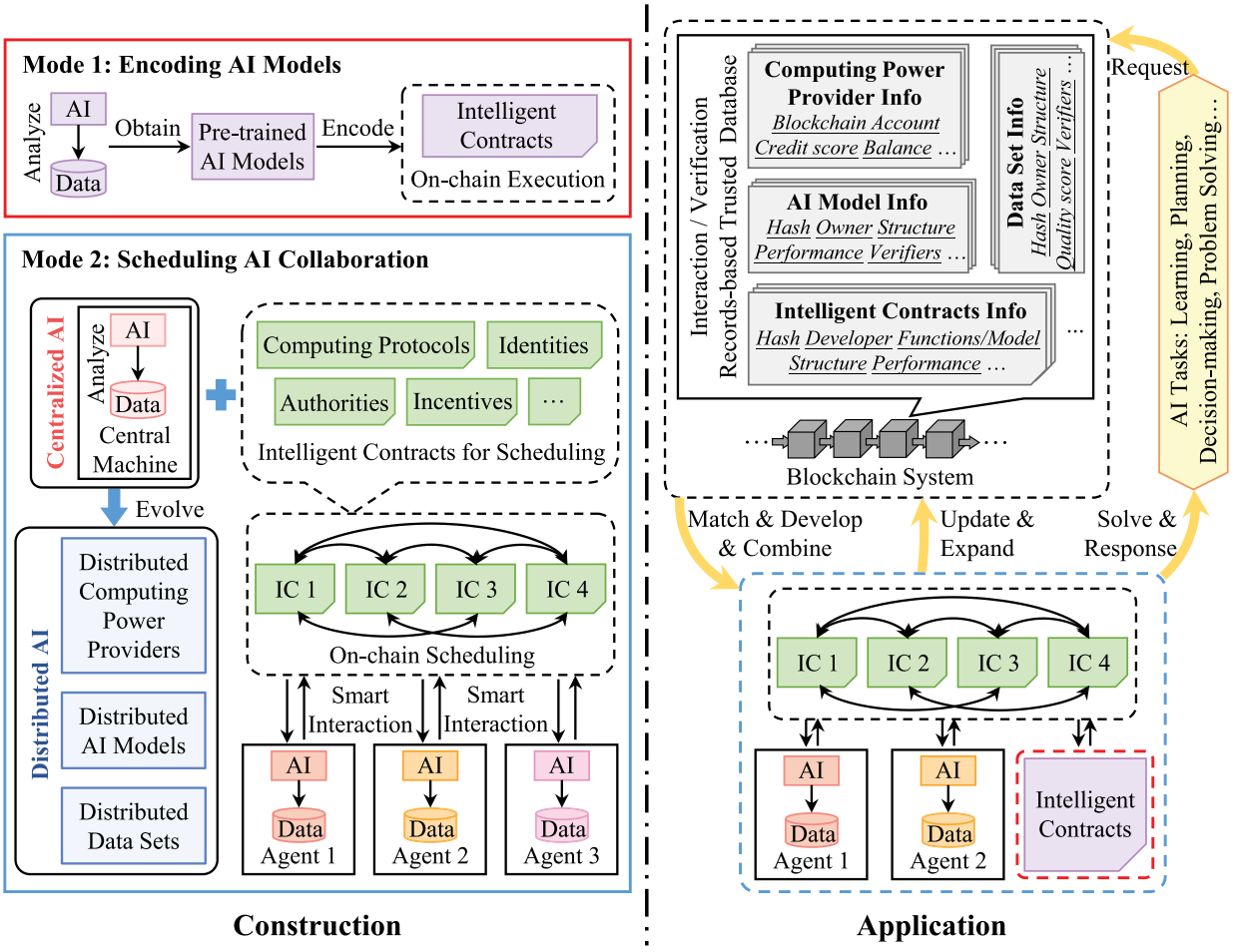



**Fig. 1.** The proposed framework. The left shows the construction of ICs, and the right illustrates the application of ICs. The red borders present Mode 1, the blue borders present Mode 2, and the dashed lines present that on-chain operations are included.

are choosing to implement their proposals with flexible and portable SCs. In other words, SCs designed for AI tasks will likely be the primary driver when the blockchain moves from processing automatic business logic to accomplishing autonomous AI tasks (AI-driven blockchain). Just as SCs designed for business logic have driven the blockchain from simply recording cryptocurrencies to automatically performing business logic. Obviously, such SCs designed for AI tasks cannot be easily classified into any of the categories listed in Table 1, and we need to study them separately. Therefore, as a distinction, we refer to them as Intelligent Contracts (ICs) in this paper and propose a constructive framework to help their systematic analysis and future expansion. As shown in Fig. 1, the proposed framework consists of two stages: the construction of ICs and the application of ICs.

### *3.2. The construction: two modes*

Considering the limited computing capabilities of current blockchain and SCs [46,47], our framework formulates two modes to construct ICs: encoding AI models and scheduling AI collaboration.

#### 3.2.1. Mode 1: encoding AI models

In this mode, ICs are constructed by directly encoding pre-trained AI models into SCs. AI models are computed and executed on the blockchain, so the constructed ICs obtain the corresponding perception, reasoning, learning, and decision-making capabilities. Taking the essential classification task in machine learning as an example, encoding pre-trained classifiers into SCs can form ICs with the same classification capabilities. When these ICs are further applied to finance, healthcare, IoT, and other fields, they can act as autonomous smart agents to classify participants' identity, behavior, and credit and thus complete multiple AI tasks. There are some attempts at this mode. For example, Harris and Waggoner [48] propose a framework of decentralized and collaborative AI on blockchain, which directly encodes three AI models supporting online learning, including naive Bayes, Nearest Centroid classifier (NC), and single-layer perceptron. Since these models encoded in ICs are publicly shared and freely available for inference, participants can collaboratively expand data sets and constantly update models.

#### 3.2.2. Mode 2: scheduling AI collaboration

In this mode, AI models computed and executed in the trusted off-chain environment indirectly supplement the functionalities of ICs. Complex AI computing is still completed off the blockchain based on technologies such as trusted computing, homomorphic encryption, and differential privacy. And ICs only verify, evaluate, fuse, and record the returned results on the blockchain, thereby providing the required intelligence as a combination. As on-chain computing may be inefficient, Mode 2 inspired by the layer 2 scaling solutions [49] that aim to offload complex computing from blockchain has become increasingly attractive. In Mode 2, the current SCs are not required for advanced computing and expression capabilities, but the combination of ICs and AI collaboration still obtains powerful computational intelligence. Meanwhile, Mode 2 provides new solutions for AI collaboration to effectively organize their modules, including distributed computing power providers, distributed AI models, and distributed data sets. And this helps to break the resource constraints of centralized AI and promote its evolution to distributed and decentralized AI.

ICs schedule AI collaboration at two levels: distributed computing protocols and distributed management mechanisms. In terms of distributed computing protocols, ICs can be combined with existing distributed AI computing architectures, such as FL and edge computing, to encode various transmission, computing, and verification protocols for data and models. For example, LearningChain [12] adopts ICs to aggregate the global gradient for their proposed decentralized stochastic gradient descent. Chain FL [50] adopts ICs to aggregate FL models' updates. These ICs perform partially distributed computing, so their models can be collaboratively tuned on the blockchain. For distributed management mechanisms, ICs can encode diverse mechanisms for identity authentication, authority management, and economic incentive, so as to organize distributed participants, crowdsource or outsource learning tasks, and share trusted data and models. For instance, Kurtulmus and Daniel [11] encode a outsourcing protocol for machine learning tasks into ICs to support constrained organizers to match competent problem solvers via blockchain. Brune [51] and Surya et al. [52] adopt ICs to link Oracle interfaces or service providers and invoke pre-trained AI models encapsulated in trusted hardware or distributed storage systems.

### 3.3. The application: management and configuration

Our idea to encourage AI-driven blockchain intelligence through the application of ICs requires two important foundations: management and configuration of the constructed ICs.

#### 3.3.1. Management

After numerous ICs and associated AI collaboration modules are constructed, it is necessary to manage them efficiently for subsequent configuration. Inspired by OpenZeppelin [53], a library of modular, reusable, and secure SCs built to facilitate development and enhance security, we believe that these ICs and AI collaboration modules should also be modular and reusable. Thus, we propose maintaining an interaction and verification records-based trusted database recording key information of computing power providers, AI models, data sets, and modular ICs. Fig. 1 gives examples of the key information that the trusted database can contain.

In a decentralized and trustless blockchain environment, trustworthiness and liquidity are crucial to ensuring the reusability of ICs and AI collaboration modules. Therefore, it is more appropriate to implement the trusted database as a blockchain-based decentralized market than a traditional centralized database. In this way, capable contributors worldwide can participate in the construction, validation, evaluation, and transaction of ICs and AI collaboration modules regardless of geographic locations and computing resources, which gives full play to swarm intelligence and ensures the necessary trustworthiness and liquidity. Our previous work, Learning Market (LM) [54], is the initial realization of this idea. LM consists of a collaboration market and a sharing market whose ICs implement both distributed computing protocols and distributed management mechanisms in Section 3.2.2. When distributed participants collaboratively train AI models in the collaboration market, ICs schedule their collaboration and automatically record key information of models and participants on the blockchain. Referring to these trusted on-chain collaboration records, distributed participants can further encapsulate, price, and trade their verified models and ICs in the sharing market.

### 3.3.2. Configuration

Based on the quantitative parameters of ICs and AI collaboration modules recorded in the trusted database, their dynamic and optimal configuration problems can be transformed into combinatorial optimization problems, in which the optimal solution is found from a set of finite feasible solutions [55]. As an essential branch of optimization problems in operations research, combinatorial optimization problems have a solid theoretical and algorithmic foundation. Hence, some classical algorithms, such as genetic and heuristic algorithms, can be applied to the configuration. For more clarity, we illustrate a feasible configuration process based on the constructed ICs of Mode 1 in Section 4.2

Fig. 1 presents the complete process of applying ICs for AI-driven blockchain intelligence: once we receive a request to accomplish a blockchain-based AI task, we match, develop, and combine ICs and associated AI collaboration modules based on their key information in a distributed problem solving or multi-agent systems approach. After these ICs complete the AI tasks via on-chain computing or on-chain scheduling, they respond to the requester, and update and expand the trusted database. When the response process of deployed ICs and associated AI collaboration modules are executed automatically, in the view of the requester, ICs autonomously generate and execute intelligent decisions, and this meets the expectations of AI-driven blockchain intelligence. We also illustrate a feasible auto-response process based on the constructed ICs of Mode 2 in Section 4.3.

## 3.4. The comparisons: smart contracts and two modes of intelligent contracts

Due to the fact that most current blockchain systems and SCs do not support computationally intensive tasks, we should analyze the costs and limitations of the evolution from SCs to ICs to guide the trade-offs between different modes. We still take Ethereum as an example, not only because it is the most mature and representative platform for SCs development, but also because its $gas$ mechanism can quantitatively characterize on-chain computational resources consumed to execute SCs and ICs.

(1) Expensive on-chain computational resources limit the complexity of on-chain control logic and on-chain computing, which affects SCs and two modes of ICs equally.

According to Ethereum yellow paper [56], all programmable computation in Ethereum is subject to fees, and any execution of given segments triggers a payment specified in units of $gas$. A transaction $T$ pays intrinsic $gas$ $g_0$ defined as Formula (1) prior to execution, and pays $C(\sigma, \mu, A, I)$ defined as Formula (2) for every given instruction. As they are defined in detail in yellow paper, we briefly explain them here. $T_i$ and $T_d$ mean the series of bytes of $T$'s associated data and initialization code, $T_t$ means $T$'s receiver and $T_t = \phi$ for a contract-creation transaction. $T_A$ is optional and means the list of access entries to warm up, and all $G$s are fixed values. $C(\sigma, \mu, A, I)$ is the general $gas$ cost function determined by full system state $\sigma$, machine state $\mu$, accrued substate $A$ and execution environment $I$. $C_{mem}(\mu_i') - C_{mem}(\mu_i)$ and $C_w$ mean the $gas$ consumed by executing the current operation $w$ due to the increase of memory usage and the computation of $w$, respectively. All $C_w$s are fully defined. Generally, $w$ that creates and modifies on-chain data consumes more $gas$ than $w$ that only reads. As shown in Formula (1) and (2), $T$'s $gas$ costs increase with the size of data and code it inputs, the number of instructions it contains, the memory it occupies, and the complexity of operation it executes. In other words, complex on-chain control logic and on-chain computing with these characteristics will easily lead to expensive execution $gas$ costs, and this equally limits the complexity of SCs and two modes of ICs.

$$g_0 \equiv \sum_{i \in T_i, T_d} \begin{cases} G_{txdatazero} & i = 0 \\ G_{txdatanonzero} & otherwise \end{cases} + \begin{cases} G_{txcreate} & T_t = \phi \\ 0 & otherwise \end{cases} + G_{transaction} + \sum_{j=0}^{\|T_A\|-1} \left(G_{accesslistaddress} + \|T_A[j]_s\| \, G_{accessliststorage}\right) \tag{1}$$

$$C(\sigma, \mu, A, I) \equiv C_{mem}(\mu_i') - C_{mem}(\mu_i) + C_w \tag{2}$$

(2) Mode 1 with on-chain computing has better transparency and security, but will be the first to reach the ceiling. Currently, Mode 1 can hardly encode popular deep learning models on Ethereum.

As on-chain computing is typically much more complex than on-chain control logic, Mode 1 will be the first to reach the ceiling. We analyze Mode 1's boundaries on the current Ethereum, including model size and calculations. For model size, Ethereum specifies that the resultant byte sequence from the execution of the initialization code is less than 24,576 bytes, i.e., 24 kB. And for calculations, theoretically, the maximum $gas$ consumed by $T$'s instruction set can be calculated as Formula (3), where $T_g$ denotes $T$'s gasLimit, and its maximum value can be taken as block's gasLimit $B_{Hl}$. At present, $B_{Hlmax}$ = 30,000,000 $gas$ [57], when $T$ with no associated data or code, $g_{0min} = G_{transaction}$ = 21,000 $gas$. Then, if we assume that all $w$ do not increase memory usage, and $C_w = 4$, which is the average $gas$ consumption for operations of ADD, SUB, MUL, and DIV, we can obtain the maximum calculations $N_{max}$ = 7,494,750. Because $C_{mem}(\mu_i') - C_{mem}(\mu_i) = 0$ is a too ideal assumption, the actual $N$ will not reach 7,494,750.

$$\sum^{N} C(\sigma, \mu, A, I)_{max} = max\ T_g - min\ g_0 = B_{Hl} - \left(\sum_{i \in T_i, T_d} \begin{cases} G_{txdatazero} & if\ i = 0 \\ G_{txdatanonzero} & otherwise \end{cases} + G_{transaction}\right) \tag{3}$$

Table 2 summarizes the sizes and FLoating-point OPerations (FLOPs) of some popular deep learning models that are pre-trained for ImageNet data set and packaged in Keras [58], including VGG16 [59], ResNet50 [60], InceptionV3 [61] and MobileNet [62]. It can be found that their sizes and FLOPs far exceed the limits of 24 kB and 7,494,750 times, which means Mode 1 can hardly encode deep learning models on current Ethereum. Moreover, in practice, Ethereum miners and clients will reject oversized transactions

**Table 2**
The size and FLOPs of popular pre-trained deep learning models packaged in Keras.

| Model | VGG16 | ResNet50 | InceptionV3 | MobileNet |
|---|---|---|---|---|
| Size | 527 MB | 98.2 MB | 91.8 MB | 16.4 MB |
| FLOPs | 30.9 G | 7.7 G | 11.6 G | 1.1 G |

to prevent denial of service attacks, leading to a more restricted application of Mode 1. Fortunately, several proposals are under development to improve the computing capabilities of blockchain and SCs, including sharding protocols, computationally cheap consensus algorithms, and layer 2 scaling solutions [63]. Their full implementations will help Mode 1 to encode more complex AI models.

(3) Mode 2 with off-chain computing is more efficient and scalable, but requires additional security and communication costs due to collaboration.

Mode 2 that offloads partial or all of the AI computing from blockchain can have comparable execution $gas$ costs to conventional SCs, making it more efficient and scalable to support complex AI models. However, Mode 2 requires additional consideration of security and communication costs for off-chain computing, on-chain interactions, and off-chain interactions. We formulate these general costs of SCs and ICs as Formula (4). $Cost$ can represent payment, time, etc. $Cost_{on-cl}$, $Cost_{on-cmp}$ and $Cost_{off-cmp}$ represent the costs of on-chain control logic, on-chain computing and off-chain computing, respectively. $Cost_{off-int}$ represents the cost of interacting with off-chain systems that guarantee the security of computing, storage, and communication, such as distributed storage systems and cryptographic tools. Because Mode 2 involves distributed collaboration, $Cost_{IC2}$ is in the form of a summation. It can be seen that when Mode 2 requires frequent on-chain control and off-chain interactions for large-scale collaboration, $Cost_{IC2}$ easily increases sharply, and once $Cost_{IC2} > Cost_{IC1}$, Mode 1 will be a better choice. Hence, there is no one-size-fits-all mode, and we need to make trade-offs based on the preferences of specific AI tasks.

$$\begin{cases} Cost_{SC} = Cost_{on-cl} \\ Cost_{IC1} = Cost_{on-cmp} \\ Cost_{IC2} = \sum^{M} Cost_{on-cl} + \sum^{P} Cost_{off-cmp} + \sum^{K} Cost_{off-int} \end{cases} \tag{4}$$

In a word, our comparisons provide theoretical evidence for the intuitive features of SCs and two modes of ICs. As we can see, ICs are not cost-free while offering many exciting benefits. Even if we execute SCs and ICs on the blockchain without transaction fees, such as Hyperledger Fabric with docker containers, these execution costs characterize their overall computational burden and cannot be ignored.

## 4. Experiments and analysis

Considering that classification is a typical and foundational task in pattern recognition, machine learning and AI [64], we use it as an example to demonstrate and evaluate the proposed framework in this section. Specifically, for the classification tasks of IRIS, MNIST and ImageNet data sets, we construct ICs on Ethereum in each of the two modes described in Section 3.2. Then, we thoroughly evaluate their performance to analyze the advantages and challenges of both modes from an experimental perspective. Also, we illustrate an optimal configuration process of Mode 1's ICs, and an automatic response process of Mode 2's ICs.

### *4.1. Experimental platform*

In this paper, all experiments are conducted on a laptop computer with Windows 10, Intel Core i7–7700HQ CPU @2.80 GHz, and 16 GB RAM. We deploy popular development platforms of SCs and AI models for experiments. Specifically, we simulate a personal Ethereum blockchain locally based on Ganache (v7.0.3) [65], and develop ICs with Truffle (v5.1.27) framework [66] using Solidity (v0.8.7) language [67]. Then, all AI models are trained and tested using Python (v3.6.10) language and Keras (v2.3.1) framework, and all private data files that need to be shared are pre-stored in a distributed storage system, Inter-Planetary File Systems (IPFS, v0.12.2) [68]. At last, we use Node.js (v12.13.0) and Web3.js (v1.7.3) [69] to interact with ICs, off-chain AI models, and IPFS for performance testing.

### *4.2. Experiments for Mode 1*

This section demonstrates the construction of Mode 1's ICs for IRIS flower classification task and MNIST handwritten digits recognition task. We thoroughly test their classification accuracy and execution costs including $gas$, time, and payment, and illustrate an optimal configuration process of Mode 1' ICs.

**Table 3**
The total number of parameters loaded in different ICs.

| Classifiers | IRIS's parameters | MNIST's parameters |
|---|---|---|
| DT | 3 | – |
| NC | 12 | 7840 |
| SVM | 15 | 7850 |
| NN2 | 51 | 7960 |
| NN3 | 83 | 8196 |

**Table 4**
The classification accuracy of all pre-trained classifiers on IRIS data set and MNIST data set.

| Classifiers | IRIS | | MNIST | | |
|---|---|---|---|---|---|
| | Train set (120) | Test set ($IRIS_{30}$) | Train set (56 000) | Test set (14 000) | $MNIST_{30}$ |
| DT | 98.33% | 93.33% | – | – | – |
| NC | 92.50% | 93.33% | 80.83% | 81.12% | 83.33% |
| SVM | 98.33% | 90.00% | 92.70% | 91.64% | 93.33% |
| NN2 | 99.17% | 93.33% | 94.70% | 94.04% | 93.33% |
| NN3 | 98.33% | 96.67% | 95.81% | 94.76% | 96.67% |

#### 4.2.1. Experimental setup

(1) Data sets: IRIS flower data set [70] and MNIST handwritten digits data set [71] are two benchmark data sets that are widely used for pattern recognition and machine learning research [72,73]. IRIS consists of 50 samples from each of three species of IRIS flower. Each sample has four centimeters features: the length and width of sepals and petals. MNIST contains 60,000 training samples and 10,000 testing samples. Each sample is a 28 × 28 pixel handwritten digit between 0–9. For pre-trained AI models (classifiers), we randomly split train sets and test sets for IRIS and MNIST in a 4:1 ratio. Considering the memory constraints of the experimental machine, we take 30 samples from the MNIST test set as the small-scale test sets for ICs, denoted as $MNIST_{30}$. And for fairness of costs comparisons, we also take 30 samples from the IRIS test sets, denoted as $IRIS_{30}$.

(2) Classifiers: Without loss of generality, we pre-train four typical classifiers including Decision Tree (DT), NC, Support Vector Machine (SVM) and Neural Networks (NN) with scikit-learn [74], and encode them into SCs to construct corresponding ICs. Their characteristics and transformations are as follows.

- DT: a representative rule-based classifier, which is supervised and nonlinear. We encode its visualized decision paths with "If-Then" statements into a SC.
- NC: a representative clustering algorithm, which is supervised and nonlinear. We store its class centroids in a SC and compute the closest centroids for prediction.
- SVM (linear kernel): a representative classifier with linear decision functions, which is supervised and linear. We store its linear decision functions in a SC and compute the maximum function value for prediction.
- NN: a representative classifier with a hierarchical structure, which is supervised and nonlinear. We adopt a two-dense-layer NN (NN2) and a three-dense-layer NN (NN3). In SCs, we first encode dense layers and activation functions (softmax and relu), and then flexibly combine them into NNs with different structures.

Since the decision paths of MNIST's DT can hardly be manually encoded, we construct five ICs of DT, NC, SVM, NN2, and NN3 for IRIS, and four ICs of NC, SVM, NN2, and NN3 for MNIST. To minimize the impact of development structures, every IC is developed standardly, which contains two functions of **loadParameters** and **predict**. ICs that encode the same classifier differ only in the variable type size, and Table 3 shows the total number of parameters loaded in different ICs. In particular, for calculating floating-point numbers and exponential functions, we uniformly expand the data by $10^4$ times and call the ABDKMath64 × 64 library [75].

#### 4.2.2. Evaluations and analysis

(1) Classification accuracy

The classification accuracy of all pre-trained classifiers is shown in Table 4. We examine the classification accuracy of all ICs on $IRIS_{30}$ and $MNIST_{30}$, and the results show that they all have the same classification accuracy as the pre-trained classifiers they encode. In other words, the ICs we constructed effectively obtain the same classification capability as the pre-trained classifiers. After being deployed on the blockchain, they can act as on-chain smart agents to autonomously classify IRIS flowers or recognize handwritten digits. Besides, when the encoded classifiers support online learning, these smart agents can even learn and evolve by themselves on the blockchain. This on-chain autonomy is consistent with Mode 1's expectations in Section 3.2.1, thus validating its effectiveness.

(2) Execution costs

To evaluate the *gas* costs of all ICs, we deploy them on Ethereum, load their parameters twice, and use them to predict $IRIS_{30}$ and $MNIST_{30}$. All their deployment and execution *gas* costs are shown in Fig. 2. And for time costs, we run the experiment of predicting $IRIS_{30}$ and $MNIST_{30}$ ten times and compare the total prediction time of different ICs in Fig. 3.

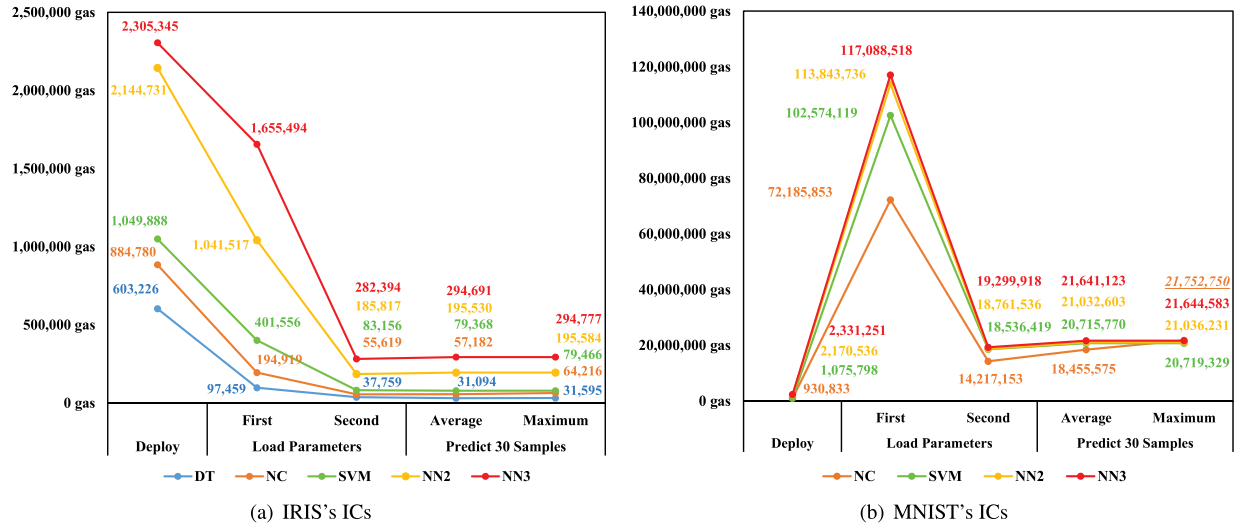



**Fig. 2.** The deployment and execution *gas* costs of different ICs.

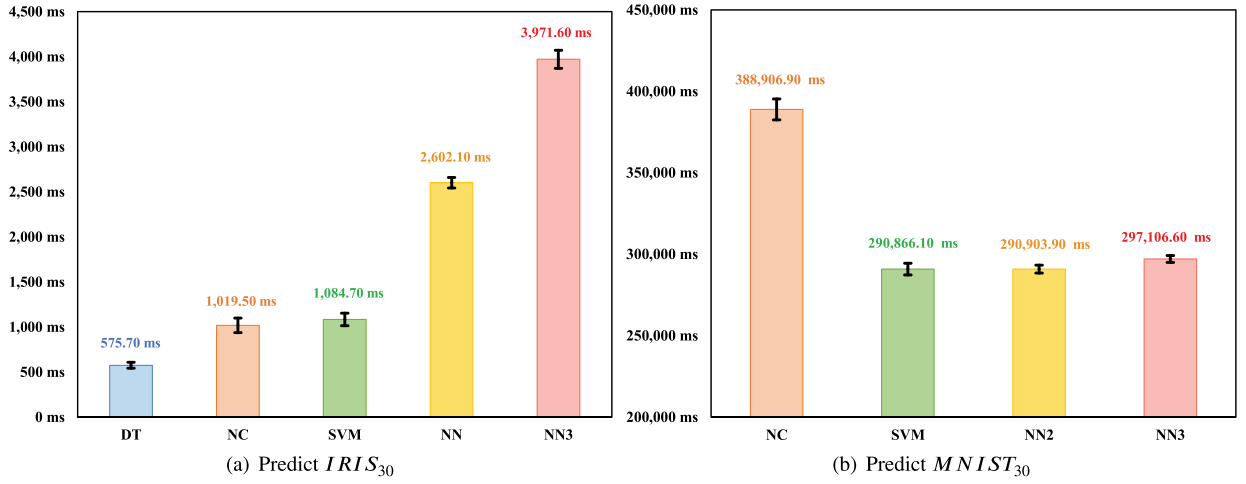



**Fig. 3.** The total prediction time of different ICs.

As shown in Fig. 2(a) and (b), ICs encoded with the same classifiers have similar deployment *gas* costs, which indicates that they are standardly developed as designed and their difference of *gas* and time costs have excluded the influence of development structures. Since Ethereum yellow paper sets a higher *gas* cost for creating new parameters than for modifying them, the *gas* costs to load parameters for the first time are much higher than for the second. As can be seen from Figs. 2(a) and 3(a), when ICs process simple IRIS data set, the complexity of classifiers is the main factor affecting *gas* and time costs. Classifiers with more parameters have higher *gas* costs for loading parameters and predicting new samples, as well as longer prediction time. As can be seen from Figs. 2(b) and 3(b), when ICs process more complex MNIST data set, the complexity of data set gradually becomes an influential factor. At this time, the *gas* costs of MNIST's ICs to load parameters and predict new samples far surpass that of deployments. And although the average prediction *gas* costs still satisfy $NC < SVM < NN2 < NN3$, there is a computationally complex sample for NC, which makes its maximum prediction *gas* costs and total prediction time increase sharply, and are even higher than others. These results suggest that both the complexity of AI models and data sets are constrained in Mode 1.

The execution payment in $USD$ and Ethereum's cryptocurrency $Eth$ is calculated as Formula (5). $EthPrice$ is the exchange rate between $Eth$ and $USD$, and $gasPrice$ is the exchange rate between $Eth$ and *gas*. $EthPrice$ is influenced by supply and demand, and transaction senders freely set $gasPrice$. The higher the $gasPrice$, the more likely the transaction will be packaged by miners. Because $gasPrice$ and $EthPrice$ have similar effects on $Cost_{USD}$, and $EthPrice$ fluctuates much more dramatically, we discuss $EthPrice$'s impact as an example. Taking $gasPrice = 2 \times 10^{10}$ *wei*, Fig. 4 shows the total prediction $Cost_{USD}$ of IRIS's ICs at 1 $Eth$ = \$4864.06 and 1 $Eth$ = \$1707.24, which are Eth's highest and lowest price in the past year ending May 20, 2022 [76]. It can be found that the fluctuation of $EthPrice$ greatly affects the $Cost_{USD}$ and determine whether an IC will be adopted in practical applications.

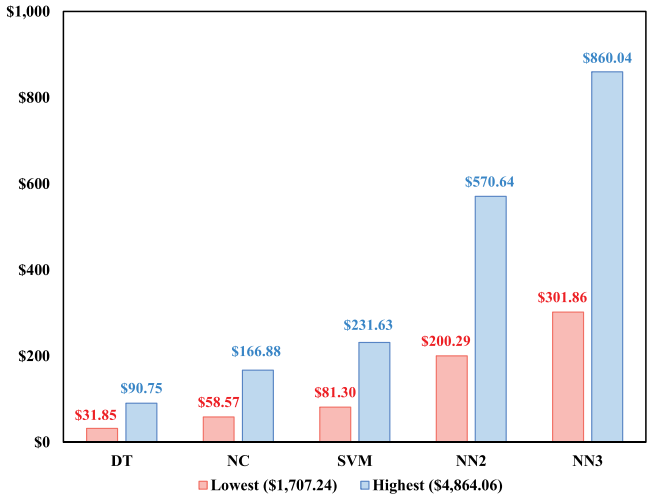



**Fig. 4.** The total prediction $Cost_{USD}$ of IRIS's ICs on $IRIS_{30}$ at $gasPrice = 2 \times 10^{10}$ $wei$, 1 $Eth$ = \$4864.06 and 1 $Eth$ = \$1707.24.

For instance, the total prediction $Cost_{USD}$ ranges from \$31.85 to \$860.04, and \$31.85 may be considered but \$860.04 is obviously unacceptable.

$$Cost_{USD} = EthPrice \times Cost_{Eth} = EthPrice \times gasPrice \times gas \tag{5}$$

Taking the obtained classification accuracy and execution costs as quantitative features of the constructed ICs, their optimal configuration problem can be modeled as a combinatorial optimization problem. For example, when an IC with the highest classification accuracy for MNIST is required, the problem is modeled as Formula (6), and then NN3 is matched. Similarly, different features and weights can be added to the objective function $F(x)$, and it can be solved with the existing algorithms.

$$min\ F(x) = \frac{1}{x},\ x \in \{Acc_{NC},\ Acc_{SVM},\ Acc_{NN2},\ Acc_{NN3}\}. \tag{6}$$

### 4.3. Experiments for Mode 2

This section demonstrates the construction of Mode 2's ICs for MNIST handwritten digits recognition task and ImageNet classification task. We design a basic AI collaboration scenario and thoroughly test their execution costs, including $gas$ and time. We also illustrate an automatic response process of Mode 2's ICs.

#### 4.3.1. Experimental setup

(1) Data sets and AI models: ImageNet data set is a large hand-annotated image database designed for visual object recognition research [77]. The Large Scale Visual Recognition Challenge 2012 (ILSVRC2012) is a subset of ImageNet containing 1,281,166 training samples and 50,000 validation samples depicting 1000 categories [78]. To ensure the fairness of comparison between Mode 1 and Mode 2, we use the same two test sets and NN3 classifier as in Section 4.2.1 for MNIST, denoting them as $MNIST_{30}$ and $MNIST_{14000}$. Accordingly, we randomly selected 30 and 150 samples from ILSVRC2012's validation set to form $ImageNet_{30}$ and $ImageNet_{150}$. The pre-trained ResNet50 packaged in Keras is adopted to classify ImageNet, which has 74.9% top-1 accuracy, 92.1% top-5 accuracy, and 25,636,712 parameters.

(2) Collaboration scenario and IC design: we take the scenario of "AI model as an IC" to demonstrate a basic implementation of Mode 2 and analyze its general characteristics. Specifically, we assume that there is an approved requester with test sets and a trusted responder with AI models. After an AI model is verified and encapsulated, the responder deploys its dedicated IC on the blockchain and monitors IC's events. Once the IC receives a model request from the requester, it triggers the responder to complete the off-chain computing and return the results to the requester. As we encode the responder's monitoring and responding operations into one program and execute it automatically and continuously, it appears that "AI model as an IC" provides an automatic AI service. Accordingly, our dedicated ICs (Model-ICs) contain three functions of **setModel**, **requestModel** and **returnRes**, and each function triggers an corresponding event, namely, "modelSet", "modelRequest" and "resReturn".

To save on-chain storage and preserve data confidentiality, we store private files containing AI models, test sets, and result files in IPFS and only record and share their content identifiers (CIDs) on the blockchain. Thus, when requesting AI models and returning results, we first encrypt and sign $CID_{test}$ and $CID_{res}$ as Formula (7), and then invoke **requstModel** and **returnRes** to transmit the obtained $Msg_{from \to to}$. In Fig. 5, we take a unified modeling language sequence diagram to show the complete collaboration process.

$$Msg_{from \to to} = \{Enc\{CID_{file}\}_{pk_{to}}, sig\{Enc\{CID_{file}\}_{pk_{to}}\}_{sk_{from}}\} \tag{7}$$

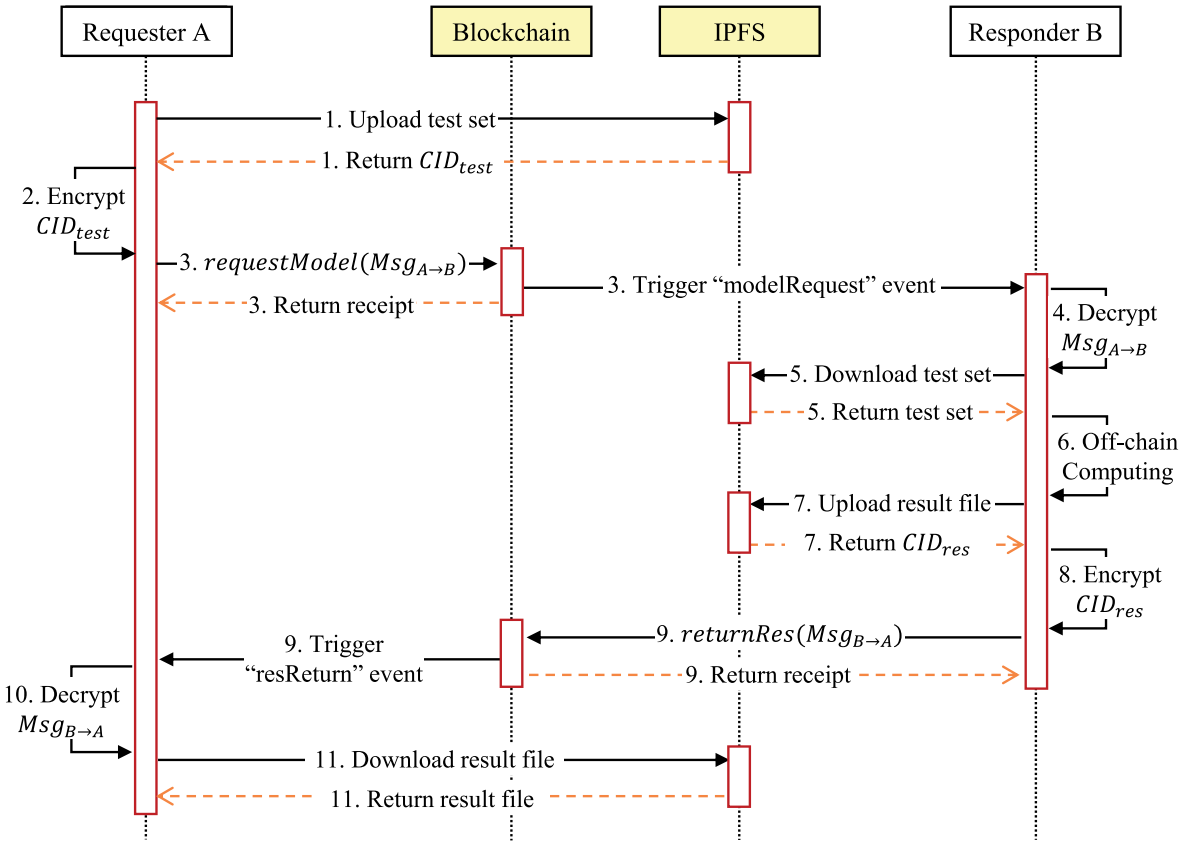



**Fig. 5.** The complete process of completing classification tasks by invoking Model-ICs in Mode 2.

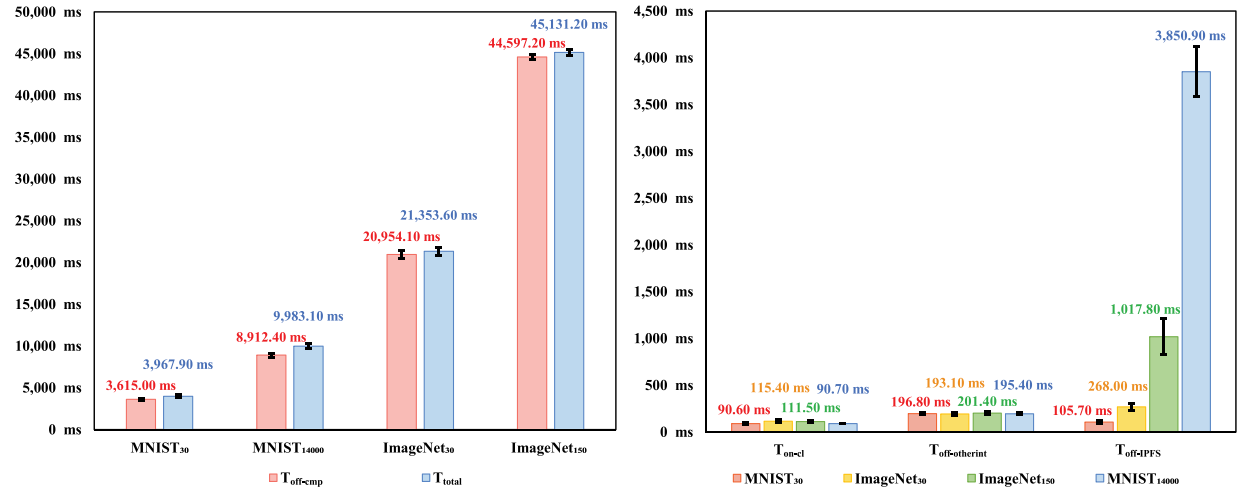
**Table 5**
The deployment and execution *gas* costs of Model-ICs for different data sets.

| Data sets | Deployment | setModel | requestModel | returnRes |
|---|---|---|---|---|
| $MNIST_{30}$ | 643,628 | 141,610 | 37,293 | 37,222 |
| $MNIST_{14000}$ | 643,628 | 141,610 | 37,293 | 37,222 |
| $ImageNet_{30}$ | 643,628 | 141,610 | 37,293 | 37,222 |
| $ImageNet_{150}$ | 643,628 | 141,610 | 37,293 | 37,222 |

#### 4.3.2. Evaluations and analysis

Due to off-chain computing and off-chain storage, our Model-ICs can support both ImageNet and MNIST classification tasks without modification. To evaluate the execution costs of *gas* and time, we deploy one Model-IC for each task, set up NN3 and ResNet50 separately, and run ten times the full experiment shown in Fig. 5 for each data set. Table 5 summarizes all deployment and execution *gas* costs. And Fig. 6 shows the total time $T_{total}$ (step 1 to step 11), off-chain computing time $T_{off-cmp}$ (step 6), on-chain control time $T_{on-cl}$ (step 3 + step 9), off-chain interaction time with IPFS $T_{off-IPFS}$ (step 1 + step 5 + step 7 + step 11), and other off-chain interaction time $T_{off-otherint}$ (step 2 + step 4 + step 8 + step 10), where the steps correspond to Fig. 5.

First, compared to Mode 1's IC that encodes NN3 classifier for MNIST, Model-IC costs significantly less *gas* and time to predict the same $MNIST_{30}$, suggesting that on-chain computing is still inefficient at this stage, and Mode 2 successfully reduces the on-chain computational burden. Then, it can be seen that Model-ICs for different data sets have the same development and execution *gas* costs. Although the most complex task of classifying $ImageNet_{150}$ with ResNet50 has the maximum $T_{total}$, this is mainly consumed by off-chain computing, with little difference in on-chain control times $T_{on-cl}$. These results indicate that Mode2's ICs can support complex data sets and AI models with almost constant on-chain computational burden. Therefore, Mode2 is highly reusable and scalable. At last, as shown in Fig. 6(b), $T_{off-IPFS}$ increases remarkably with the size of data set. For the largest $MNIST_{14000}$, $T_{off-IPFS}$ has far surpassed $T_{on-cl}$ and $T_{off-otherint}$, and become the most influential factor affecting $T_{total}$ other than $T_{off-cmp}$. And the more frequent the interaction with IPFS, the more obvious the impact of $T_{off-IPFS}$, which is consistent with our analysis of $Cost_{off-int}$ in Section 3.4. That is, although Mode 2 with off-chain computing is usually more efficient than Mode 1 with on-chain computing, Mode 1 is always worth considering when $Cost_{off-cmp}$ and $Cost_{off-int}$ increase dramatically due to massive collaboration, frequent interactions, and expensive trusted execution environments. Note that our Model-ICs can be applied to larger ImageNet




(a) $T_{off-cmp}$ and $T_{total}$ (b) $T_{on-cl}$, $T_{off-IPFS}$ and $T_{off-otherint}$

**Fig. 6.** The time costs of classification tasks for different data sets, where $MNIST_{30}$, $MNIST_{14000}$, $ImageNet_{30}$ and $ImageNet_{150}$ are 182 kB, 81.3 MB, 3.7 MB and 18.9 MB in size.

data sets, but their $T_{off-cmp}$ and $T_{off-IPFS}$ will be much larger than $T_{on-cl}$ and $T_{off-otherint}$. For the clarity of comparison in Fig. 6, $ImageNet_{30}$ and $ImageNet_{150}$ are selected.

In summary, the experiments in Section 4 not only demonstrate the construction of ICs in two modes, but also validate and supplement the theoretical analysis in Section 3.4. Aldweesh et al. [79] and Nelaturu et al. [80] collect two benchmark databases of deployed Ethereum SCs, whose publicly available deployment $gas$ are in the range of $[90,543,\ 3,544,767]$ and execution $gas$ are in the range of $[22,643,\ 646,727]$. In comparison, the $gas$ costs of Mode 2's Model-ICs are relatively lower than most current SCs, the $gas$ costs of Mode 1's ICs for IRIS are comparable to current SCs, and the $gas$ costs of Mode 1's ICs for MNIST has far exceeded current SCs. These experimental data suggest that the computation of four basic classifiers on MNIST is sufficiently complex for the current Ethereum. In practice, we recommend Mode 1 not encode more complex AI models or data sets on Ethereum. Comparatively, Mode2 can break through these computational limitations to efficiently support complex AI models and data sets, but additional security and communication costs must be considered and optimized.

## 5. Conclusion

In this paper, we focus on an important type of Smart Contracts (SCs) designed to accomplish blockchain-based AI tasks, i.e., Intelligent Contracts (ICs), and propose a constructive framework for their construction and application. Specifically, our framework formulates two modes to construct ICs: encoding AI models of Mode 1 and scheduling AI collaboration of Mode 2. And we propose to help blockchain respond to different AI tasks through the dynamic and optimal configuration of the constructed ICs, thus encouraging AI-driven blockchain intelligence. For typical AI tasks of classifying IRIS, MNIST, and ImageNet data sets, we implement and evaluate two modes of ICs on Ethereum, respectively. Then, we illustrate an optimal configuration process of Mode 1's ICs, and an automatic response process of Mode 2's ICs. All experimental results demonstrate the effectiveness and feasibility of our framework. Moreover, parallel comparisons of the two modes are conducted from theoretical and experimental perspectives to guide mode selection. They indicate that Mode 1 with on-chain computing outperforms on-chain autonomy, transparency, and security, but is constrained by the computing capabilities of current blockchain systems. Whereas Mode 2 with off-chain computing excels in supporting complex AI models and data sets, but has additional security and communication costs. We estimate the theoretical boundaries of Ethereum SCs in Mode 1. Also, our experimental data show that Mode 1's on-chain computing on MNIST data set is sufficiently complex for the current Ethereum.

## CRediT authorship contribution statement

**Liwei Ouyang:** Conceptualization, Methodology, Software, Visualization, Writing – original draft, Writing – review & editing. **Wenwen Zhang:** Conceptualization, Writing – review & editing, Supervision. **Fei-Yue Wang:** Conceptualization, Writing – review & editing, Supervision.

## Declaration of competing interest

No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.compeleceng.2022.108421.

## Data availability

The data supporting our findings are publicly available in https://archive.ics.uci.edu/ml/datasets/iris, http://yann.lecun.com/exdb/mnist/, https://image-net.org/challenges/LSVRC/2012/.

## Acknowledgments

This work is supported in part by National Key R&D Program of China (2020YFB1600400), in part by Natural Science Foundation of China (U1811463), and the Science and Technology Development Fund, Macau SAR (File No. 0050/2020/A1).

## References

[1] Wang Fei-Yue. Blockchain intelligence: cornerstone of the future smart economy and smart societies [keynotes]. In: Proceedings of the 2nd World Intelligence Congress. 2018.
[2] Dinh Thang N, Thai My T. AI and blockchain: A disruptive integration. Computer 2018;51(9):48–53.
[3] Pandl Konstantin D, Thiebes Scott, Schmidt-Kraepelin Manuel, Sunyaev Ali. On the convergence of artificial intelligence and distributed ledger technology: A scoping review and future research agenda. IEEE Access 2020;8:57075–95.
[4] Zheng Zibin, Hong-Ning Dai, Jiajing Wu. Blockchain intelligence: Methods, applications and challenges. Springer Nature Singapore Pte Ltd.; 2021.
[5] Özyilmaz Kazim Rifat, Doğan Mehmet, Yurdakul Arda. IDMoB: IoT data marketplace on blockchain. In: 2018 crypto valley conference on blockchain technology. CVCBT, IEEE; 2018, p. 11–9.
[6] Somy Nishant Baranwal, Kannan Kalapriya, Arya Vijay, Hans Sandeep, Singh Abhishek, Lohia Pranay, Mehta Sameep. Ownership preserving AI market places using blockchain. In: 2019 IEEE international conference on blockchain (Blockchain). IEEE; 2019, p. 156–65.
[7] Zhao Lingchen, Wang Qian, Wang Cong, Li Qi, Shen Chao, Feng Bo. Veriml: Enabling integrity assurances and fair payments for machine learning as a service. IEEE Trans Parallel Distrib Syst 2021;32(10):2524–40.
[8] Yang Ruizhe, Yu F Richard, Si Pengbo, Yang Zhaoxin, Zhang Yanhua. Integrated blockchain and edge computing systems: A survey, some research issues and challenges. IEEE Commun Surv Tutor 2019;21(2):1508–32.
[9] Nguyen Dinh C, Ding Ming, Pham Quoc-Viet, Pathirana Pubudu N, Le Long Bao, Seneviratne Aruna, Li Jun, Niyato Dusit, Poor H Vincent. Federated learning meets blockchain in edge computing: Opportunities and challenges. IEEE Internet Things J 2021.
[10] Beck Roman, Czepluch Jacob, Lollike Nikolaj, Malone Simon. Blockchain – the gateway to trust-free cryptographic transactions. In: 2016 24th European conference on information systems. ECIS, 2016, p. 153.
[11] Kurtulmus A Besir, Daniel Kenny. Trustless machine learning contracts; evaluating and exchanging machine learning models on the ethereum blockchain. 2018, arXiv preprint arXiv:1802.10185.
[12] Chen Xuhui, Ji Jinlong, Luo Changqing, Liao Weixian, Li Pan. When machine learning meets blockchain: A decentralized, privacy-preserving and secure design. In: 2018 IEEE international conference on big data (Big data). IEEE; 2018, p. 1178–87.
[13] He Ningyu, Wu Lei, Wang Haoyu, Guo Yao, Jiang Xuxian. Characterizing code clones in the ethereum smart contract ecosystem. In: International conference on financial cryptography and data security. Springer; 2020, p. 654–75.
[14] Nakamoto Satoshi. Bitcoin: a peer-to-peer electronic cash system. 2008, https://bitcoin.org/bitcoin.pdf.
[15] Yuan Yong, Wang Fei-Yue. Blockchain and cryptocurrencies: Model, techniques, and applications. IEEE Trans Syst Man Cybern: Syst 2018;48(9):1421–8.
[16] Szabo Nick. Smart contracts. 1994, https://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart.contracts.html.
[17] He Daojing, Deng Zhi, Zhang Yuxing, Chan Sammy, Cheng Yao, Guizani Nadra. Smart contract vulnerability analysis and security audit. IEEE Netw 2020;34(5):276–82.
[18] Cook Victor, Painter Zachary, Peterson Christina, Dechev Damian. Read-uncommitted transactions for smart contract performance. In: 2019 IEEE 39th international conference on distributed computing systems. ICDCS, IEEE; 2019, p. 1960–70.
[19] Bünz Benedikt, Agrawal Shashank, Zamani Mahdi, Boneh Dan. Zether: Towards privacy in a smart contract world. In: International conference on financial cryptography and data security. Springer; 2020, p. 423–43.
[20] Giancaspro Mark. Is a 'smart contract'really a smart idea? Insights from a legal perspective. Comput Law Secur Rev 2017;33(6):825–35.
[21] Wang Shuai, Ouyang Liwei, Yuan Yong, Ni Xiaochun, Han Xuan, Wang Fei-Yue. Blockchain-enabled smart contracts: architecture, applications, and future trends. IEEE Trans Syst Man Cybern: Syst 2019;49(11):2266–77.
[22] Jordan Michael I, Mitchell Tom M. Machine learning: Trends, perspectives, and prospects. Science 2015;349(6245):255–60.
[23] LeCun Yann, Bengio Yoshua, Hinton Geoffrey. Deep learning. Nature 2015;521(7553):436–44.
[24] Sutton Richard S, Barto Andrew G. Reinforcement learning: An introduction. MIT Press; 2018.
[25] Miiller Y. Decentralized artificial intelligence. Decent AI 1990;3–13.
[26] Durfee Edmund H. Distributed problem solving and planning. In: ECCAI advanced course on artificial intelligence. Springer; 2001, p. 118–49.
[27] Ferber Jacques, Weiss Gerhard. Multi-agent systems: An introduction to distributed artificial intelligence, Vol. 1. Addison-Wesley Reading; 1999.
[28] Yang Qiang, Liu Yang, Chen Tianjian, Tong Yongxin. Federated machine learning: Concept and applications. ACM Trans Intell Syst Technol 2019;10(2):1–19.
[29] Shi Weisong, Cao Jie, Zhang Quan, Li Youhuizi, Xu Lanyu. Edge computing: Vision and challenges. IEEE Internet Things J 2016;3(5):637–46.
[30] Kang Jiawen, Yu Rong, Huang Xumin, Wu Maoqiang, Maharjan Sabita, Xie Shengli, Zhang Yan. Blockchain for secure and efficient data sharing in vehicular edge computing and networks. IEEE Internet Things J 2018;6(3):4660–70.
[31] Liang Xueping, Shetty Sachin, Tosh Deepak, Kamhoua Charles, Kwiat Kevin, Njilla Laurent. Provchain: A blockchain-based data provenance architecture in cloud environment with enhanced privacy and availability. In: 2017 17th IEEE/ACM international symposium on cluster, cloud and grid computing. CCGRID, IEEE; 2017, p. 468–77.
[32] Wang Yuntao, Su Zhou, Zhang Ning. BSIS: Blockchain-based secure incentive scheme for energy delivery in vehicular energy network. IEEE Trans Ind Inf 2019;15(6):3620–31.
[33] Kim Hyesung, Park Jihong, Bennis Mehdi, Kim Seong-Lyun. Blockchained on-device federated learning. IEEE Commun Lett 2019;24(6):1279–83.
[34] Majeed Umer, Hong Choong Seon. Flchain: Federated learning via MEC-enabled blockchain network. In: 2019 20th Asia-Pacific network operations and management symposium. APNOMS, IEEE; 2019, p. 1–4.
[35] Cui Laizhong, Su Xiaoxin, Ming Zhongxing, Chen Ziteng, Yang Shu, Zhou Yipeng, Xiao Wei. CREAT: Blockchain-assisted compression algorithm of federated learning for content caching in edge computing. IEEE Internet Things J 2020;1. http://dx.doi.org/10.1109/JIOT.2020.3014370.
[36] Tasca Paolo, Hayes Adam, Liu Shaowen. The evolution of the bitcoin economy: Extracting and analyzing the network of payment relationships. J Risk Financ 2018.

[37] Linoy Shlomi, Stakhanova Natalia, Matyukhina Alina. Exploring ethereum's blockchain anonymity using smart contract code attribution. In: 2019 15th international conference on network and service management. CNSM, 2019, p. 1–9. http://dx.doi.org/10.23919/CNSM46954.2019.9012681.

[38] Su Moting, Zhang Zongyi, Zhu Ye, Zha Donglan. Data-driven natural gas spot price forecasting with least squares regression boosting algorithm. Energies 2019;12(6):1094.

[39] Salah Khaled, Rehman MHabib Ur, Nizamuddin Nishara, Al-Fuqaha Ala. Blockchain for AI: Review and open research challenges. IEEE Access 2019;7:10127–49.

[40] Oliva Gustavo A, Hassan Ahmed E, Jiang Zhen Ming Jack. An exploratory study of smart contracts in the Ethereum blockchain platform. Empir Softw Eng 2020;25(3):1864–904.

[41] Bartoletti Massimo, Pompianu Livio. An empirical analysis of smart contracts: platforms, applications, and design patterns. In: International conference on financial cryptography and data security. Springer; 2017, p. 494–509.

[42] Wöhrer Maximilian, Zdun Uwe. Design patterns for smart contracts in the ethereum ecosystem. In: 2018 IEEE international conference on internet of things (iThings) and IEEE green computing and communications (GreenCom) and IEEE cyber, physical and social computing (CPSCom) and IEEE smart data (SmartData). IEEE; 2018, p. 1513–20.

[43] Hu Teng, Liu Xiaolei, Chen Ting, Zhang Xiaosong, Huang Xiaoming, Niu Weina, Lu Jiazhong, Zhou Kun, Liu Yuan. Transaction-based classification and detection approach for Ethereum smart contract. Inf Process Manage 2021;58(2):102462.

[44] Zhang Weishan, Lu Qinghua, Yu Qiuyu, Li Zhaotong, Liu Yue, Lo Sin Kit, Chen Shiping, Xu Xiwei, Zhu Liming. Blockchain-based federated learning for device failure detection in industrial IoT. IEEE Internet Things J 2020;8(7):5926–37.

[45] Sarpatwar Kanthi, Sitaramagiridharganesh Ganapavarapu Venkata, Shanmugam Karthikeyan, Rahman Akond, Vaculin Roman. Blockchain enabled AI marketplace: The price you pay for trust. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops. 2019, p. 1–10.

[46] Zou Weiqin, Lo David, Kochhar Pavneet Singh, Le Xuan-Bach Dinh, Xia Xin, Feng Yang, Chen Zhenyu, Xu Baowen. Smart contract development: Challenges and opportunities. IEEE Trans Softw Eng 2019;47(10):2084–106.

[47] Zheng Zibin, Xie Shaoan, Dai Hong-Ning, Chen Weili, Chen Xiangping, Weng Jian, Imran Muhammad. An overview on smart contracts: Challenges, advances and platforms. Future Gener Comput Syst 2020;105:475–91.

[48] Harris Justin D, Waggoner Bo. Decentralized and collaborative AI on blockchain. In: 2019 IEEE international conference on blockchain (Blockchain). IEEE; 2019, p. 368–75.

[49] Stark Josh. Making sense of ethereum's layer 2 scaling solutions: State channels, plasma, and truebit. 2018, https://medium.com/l4-media/making-sense-of-ethereums-layer-2-scaling-solutions-state-channels-plasma-and-truebit-22cb40dcc2f4.

[50] Korkmaz Caner, Kocas Halil Eralp, Uysal Ahmet, Masry Ahmed, Ozkasap Oznur, Akgun Baris. Chain fl: Decentralized federated machine learning via blockchain. In: 2020 second international conference on blockchain computing and applications. BCCA, IEEE; 2020, p. 140–6.

[51] Brune Philipp. Towards an enterprise-ready implementation of artificial intelligence-enabled, blockchain-based smart contracts. 2020, arXiv preprint arXiv:2003.09744.

[52] Surya Mahenda Metta, Wibowo Avenia Clarissa, et al. Converging artificial intelligence and blockchain technology using oracle contract in ethereum blockchain platform. In: 2020 fifth international conference on informatics and computing. ICIC, IEEE; 2020, p. 1–5.

[53] Openzeppelin. OpenZeppelin github homepage. 2022, https://github.com/OpenZeppelin/openzeppelin-contracts.

[54] Ouyang Liwei, Yuan Yong, Wang Fei-Yue. Learning markets: an ai collaboration framework based on blockchain and smart contracts. IEEE Internet Things J 2022;9(16):14273–86.

[55] Papadimitriou Christos H, Steiglitz Kenneth. Combinatorial optimization: Algorithms and complexity. Courier Corporation; 1998.

[56] Wood Gavin, et al. Ethereum: A secure decentralised generalised transaction ledger (Berlin Version 47f1752). In: Ethereum project yellow paper. 2022, p. 1–41.

[57] Ethereum. Ethereum document. 2022, https://ethereum.org/en/developers/docs/blocks/#block-size.

[58] Chollet François, et al. Keras. 2015, https://keras.io.

[59] Simonyan Karen, Zisserman Andrew. Very deep convolutional networks for large-scale image recognition. 2014, arXiv preprint arXiv:1409.1556.

[60] He Kaiming, Zhang Xiangyu, Ren Shaoqing, Sun Jian. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016, p. 770–8.

[61] Szegedy Christian, Vanhoucke Vincent, Ioffe Sergey, Shlens Jon, Wojna Zbigniew. Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016, p. 2818–26.

[62] Howard Andrew G, Zhu Menglong, Chen Bo, Kalenichenko Dmitry, Wang Weijun, Weyand Tobias, Andreetto Marco, Adam Hartwig. Mobilenets: Efficient convolutional neural networks for mobile vision applications. 2017, arXiv preprint arXiv:1704.04861.

[63] Xie Junfeng, Yu F Richard, Huang Tao, Xie Renchao, Liu Jiang, Liu Yunjie. A survey on the scalability of blockchain systems. IEEE Netw 2019;33(5):166–73.

[64] Kotsiantis Sotiris B, Zaharakis Ioannis D, Pintelas Panayiotis E. Machine learning: a review of classification and combining techniques. Artif Intell Rev 2006;26(3):159–90.

[65] Ganache. Ganache github homepage. 2022, https://github.com/trufflesuite/ganache.

[66] Truffle. Truffle github homepage. 2022, https://github.com/trufflesuite/truffle.

[67] Ethereum. The document of solidity v0.8.7. 2022, https://docs.soliditylang.org/en/v0.8.7/.

[68] Benet Juan. Ipfs-content addressed, versioned, p2p file system. 2014, arXiv preprint arXiv:1407.3561.

[69] Web3js. Web3.js - Ethereum JavaScript API. 2022, https://web3js.readthedocs.io/en/v1.7.3/.

[70] Fisher Ronald A. The use of multiple measurements in taxonomic problems. Ann Eugen 1936;7(2):179–88.

[71] LeCun Yann, Cortes Corinna, Burges CJ. MNIST handwritten digit database, Vol. 2. ATT Labs; 2010, [Online]. Available: http://yann.lecun.com/exdb/mnist.

[72] Devasena C Lakshmi, Sumathi T, Gomathi VV, Hemalatha M. Effectiveness evaluation of rule based classifiers for the classification of iris data set. Bonfring Int J Man Mach Interface 2011;1(Special Issue Inaugural Special Issue):05–9.

[73] Deng Li. The mnist database of handwritten digit images for machine learning research [best of the web]. IEEE Signal Process Mag 2012;29(6):141–2.

[74] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E. Scikit-learn: Machine Learning in Python. J Mach Learn Res 2011;12:2825–30.

[75] Mikhail Vladimirov. ABDK libraries for solidity. 2022, https://github.com/abdk-consulting/abdk-libraries-solidity.

[76] Investing. The historical data of ethereum. 2022, https://www.investing.com/crypto/ethereum/historical-data.

[77] Deng Jia, Dong Wei, Socher Richard, Li Li-Jia, Li Kai, Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. Ieee; 2009, p. 248–55.

[78] Russakovsky Olga, Deng Jia, Su Hao, Krause Jonathan, Satheesh Sanjeev, Ma Sean, Huang Zhiheng, Karpathy Andrej, Khosla Aditya, Bernstein Michael, Berg Alexander C, Fei-Fei Li. ImageNet Large Scale Visual Recognition Challenge. Int J Comput Vis (IJCV) 2015;115(3):211–52. http://dx.doi.org/10.1007/s11263-015-0816-y.

[79] Aldweesh Amjad, Alharby Maher, Solaiman Ellis, van Moorsel Aad. Performance benchmarking of smart contracts to assess miner incentives in Ethereum. In: 2018 14th European dependable computing conference. EDCC, IEEE; 2018, p. 144–9.

[80] Nelaturu Keerthi, Beillahit Sidi Mohamed, Long Fan, Veneris Andreas. Smart contracts refinement for gas optimization. In: 2021 3rd conference on blockchain research & applications for innovative networks and services. BRAINS, IEEE; 2021, p. 229–36.

**Liwei Ouyang** received the B.S. degree in automation from Xi'an Jiaotong University, Xi'an, China, in 2018. She is currently pursuing the Ph.D. degree in social computing with the State Key Laboratory for Management and Control of Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, China. Her current research interests include social computing, blockchain and smart contracts.

**Wenwen Zhang** received his bachelor degree in 2014, and the Ph.D. degree in 2021 both from Xi'an Jiaotong University, Xi'an, China. He is currently a Postdoctoral Teaching Fellow at School of Computer Science, Shaanxi Normal University, Xi'an, China. His research interests include representation learning, computer vision and machine learning.

**Fei-Yue Wang** professor at the Institute of Automation, Chinese Academy of Sciences, director of the State Key Laboratory for Management and Control of Systems, director of China Economic and Social Security Research Center at University of Chinese Academy of Sciences. His research interest covers theories, methods, and applications for robotics, AI, intelligent control, parallel systems, social computing, parallel intelligence, and knowledge automation.