Received June 12, 2019, accepted July 3, 2019, date of publication July 12, 2019, date of current version August 12, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2928325

Smart Contract Based Data Trading Mode Using Blockchain and Machine Learning

WEI XIONG AND LI XIONG
Department of Information Management, School of Management, Shanghai University, Shanghai, China
Corresponding author: Li Xiong (xiongli8@163.com)

ABSTRACT There are two traditional data trading modes, the hosting mode, and the aggregation mode, which depend on the trusted third parties to a large extent. The hosting mode is that the data are completely hosted in the data trading center, so the data trading center retains the data. On the surface, the aggregation mode is that the data trading center is not to retain the data of trading, but actually, it has the ability to retain the data. There is a fundamental difference between the ability to retain the data and the inability to retain the data. These two trading modes cause the data owners to be afraid to share data trading. In this paper, we propose a solution to the data trading mode based on the smart contract using blockchain and machine learning. Our solution takes advantage of the immutability, tamper-proof and traceability of blockchain, the programmability of smart contract, and the verification of data availability by the similarity learning to propose a challenge response mechanism between the data purchaser and the data owner, an off-chain download mechanism between the data purchaser and the data storage service provider, and an arbitration mechanism for the controversy resolution of the data trading. The challenge response mechanism is used to authenticate and authorize the data owner, the off-chain download mechanism is used to authenticate and authorize the data purchaser to download the purchased data, and the similarity learning is used to deal with the controversy over the data availability in the data trading. The design and implementation of data trading smart contract successfully achieved the goal of removing the trusted third party in the data trading, and thus, the problem that the data trading center has the ability to retain the data in the process of the data trading is solved, as well as the automatic payment by using the Ethereum encrypted currency among the trading participants is realized. This paper presents the whole process of smart contract from the design and implementation to the test completion and provides the security analysis and performance evaluation. The full code of smart contract and the ABI interface have been uploaded to the GitHub for the public release.

INDEX TERMS Data trading mode, smart contract, blockchain, machine learning, Ethereum.

I. INTRODUCTION

As many new technologies are integrated into our daily life, such as the mobile and the social network applications, and the smart systems based on the Internet of Things (IoT) (smart home, smart city, smart transportation, smart grid, etc.), a large amount of data will be collected [1]. We have entered the era of big data, the data sharing and trading is the trend of the times and the inevitable demand of the market. As people pay more and more attention to the economic value of big data in improving the utility efficiency and the decision-making, the customer experience and other aspects, some third-party big data trading centers have been established [2]. In order to meet the growth of data demands, the data trading centers provide the data owners and the data purchasers with the interconnected space.

However, the data has its particularity, that is, there is no uniqueness, no clear ownership constraints; once seen, there is the ownership; the data replication is completely undifferentiated. Therefore, when the data owners trade the data through the data trading centers, it is critical that the data trading centers cannot retain the data.

At present, there are two traditional data trading modes [3]. One is the hosted trading mode, in which each data owner hosts his own data to the database of data trading center, and the data trading center trades with the data purchaser. As shown in Figure 1.

After the data owner hosts the data, the data is entirely owned by the data trading center. The subsequent application of data has nothing to do with the data owner, for example,

The associate editor coordinating the review of this manuscript and approving it for publication was Xiang Zhao.

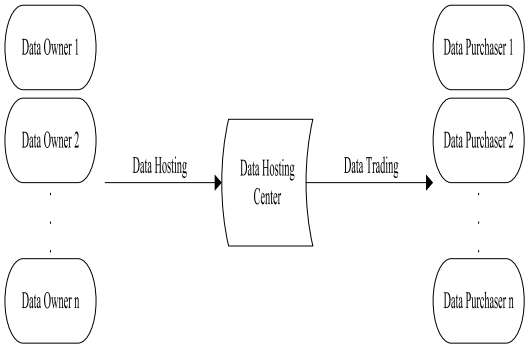

FIGURE 1. The hosted trading mode.

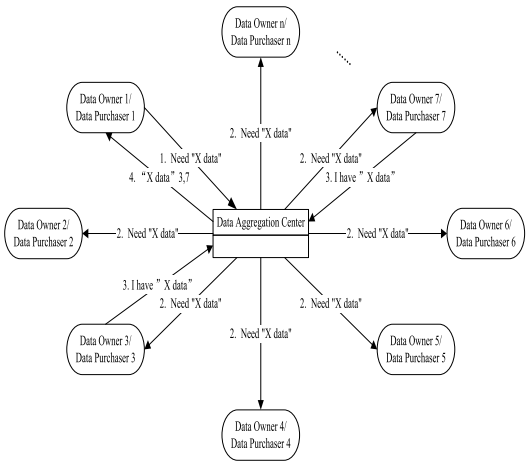

FIGURE 2. The aggregation trading mode.

the object of trading, the number of trading, etc. The rights and interests of data owners depend entirely on the credit of the data trading center.

The other is the aggregation trading mode. The data trading center links the data owners through the API interface. The data owners do not need to host the data to the data trading center beforehand, and the data is managed by the data owners themselves. When a data purchaser needs to purchase the data, he/she interacts with the data owner through the data trading center in real time. The data owner who has the data responds to the information and returns it to the data purchaser by the data trading center. It is worth noting that the data owner in this trading mode is also the data purchaser. As shown in Figure 2.

The trading steps of the aggregation trading mode are as follows:

1. The Data Purchaser 1 sends the request to purchase the ''X data'' to the data trading center.
2. The data trading center sends the request to all the data owners except the Data Purchaser 1.
3. The Data Owner 3 and the Data Owner 7 respond and sent the ''X data'' related information to the data trading center.
4. The data trading center aggregates the ''X data'' related information of the Data Owner 3 and the Data Owner 7 and sends it to the Data Purchaser 1.
5. The Data Purchaser 1 selects the data to be purchased from the ''X Data'' summary information, and obtains the corresponding data from the corresponding API interface after purchase.

On the surface, the trading data in the aggregation trading mode are controlled by the data owners. The data trading center only serves to link and match trading. But after a careful analysis of the API interface data acquisition mechanism, it will be found that the data trading center using the aggregation trading mode actually has the ability and the opportunity to retain the trading data, so it has the data. With the continuous trading and the data accumulation, a data aggregation center has gradually become a data hosting center.

As a trusted third party, the data trading center shows its unwillingness and no incentive to retain the data. But the inability to retain the data and the unwillingness to retain the data are two concepts. The inability to retain the data is that wants to do but no ability to do, which is not a threat. The unwillingness to retain the data is that is capable of doing but not doing, which is a potential threat. This potential threat is due to the centralization of the data trading centers.

The centralized platforms are vulnerable to the single-point failures, lack of transparency and the clear hierarchy of power structure, which are prone to the corruption. In addition, the centralized platforms also rely on the trusted third parties to pay, which make them unreliable and untrustworthy [4].

On the other hand, using the blockchain can achieve the decentralization to improve the traditional data trading mode. The blockchain technology [5] is based on the decentralized distributed ledger and has the characteristics of the immutability and tamper-proof [6]. At present, the blockchain has provided the solutions to some problems in the fields of the finance [7], the medical care [8], the intrusion detection [9], the food industry [10], and the supply chain management [11]. The blockchain security performance includes the transparency, traceability and auditability [12]. The Ethereum [13] is an open blockchain platform. In addition to the event systems and tamper-proof ordered logs, it allows the blockchains to execute code lines and makes them programmable [14], aiming at ensuring the security, adaptability and flexibility. It pioneered the development of smart contract [15], which is a decentralized application that can be programmed and deployed as an ''automation program'' in a blockchain. The trust plays a very important role in the data trading mode and is implemented by the blockchain, because all the trading are transparent and do not require the participation of the trusted third parties.

The solution we propose in this paper mainly focuses on the use of the blockchain, smart contract and similarity learning [16] to design the solution to prove that the data trading centers are unable to retain the data of trading in the process of the data trading. The main focus of our solution is to eliminate the needs of the trusted third parties and overcome the trust problem among the participants in the data trading process.

The main contributions of this paper can be summarized as follows:

1. In order to eliminate the problem that the data trading center can retain the data in the process of the data trading, we propose a decentralized, secure and trusted data trading mode solution and framework, and provide the security analysis and performance evaluation. Our solution takes advantage of the key functions of the Ethereum blockchain, smart contract, similarity learning, and Inter-Planetary File System (IPFS) [17].
2. In the process of designing the Ethereum smart contract, we design various algorithms to realize the automation of Ethereum encrypted currency payment, encourage the participants to act honestly, and use the machine learning algorithm to solve disputes about the availability of the data of trading.
3. We have implemented the smart contract and released the source code to the GitHub [18] (https://github.com/106968687/DT-SC).

The rest of the paper is arranged as follows: The related works are discussed in Section II. The proposed blockchain solution is introduced in Section III. The solution implementation and test works are presented in Section IV. The security analysis and performance evaluation of the implemented solution are discussed in Section V. Finally, the conclusion and future work are introduced in Section VI.

II. RELATED WORK

In this section, we will review and summarize the existing blockchain-based solutions in the literature on the data trading modes.

Yang [3] based on the research of the cryptography and blockchain technology, a new decentralized data trading platform is designed, which guarantees that the platform has no opportunity to view, copy and retain the data of trading, and realizes the safe flow of the trading data only in both sides of the trading. His solution only proposes a decentralized data trading platform concept using the blockchain technology, and the concept has not been verified.

Wang *et al.* [19] studied the blockchain technology and the sharing and opening mechanism, and discussed the data operation and management mode based on the blockchain technology to ensure that the data is not copied and retained by the third party and can be traded safely. Their solution is a discussion of the blockchain technology for the data trading without any technical implementation content.

Lu *et al.* [20] aiming at the shortcomings of the traditional data trading modes in the data forgery and the data retention, the architecture of the trusted data trading platform based on the blockchain technology under the collaborative mechanism is designed, which provides the ideas and methods for designing and developing a trusted data trading platform. Their solution does not design any algorithm, but only through language description, so it can not verify its feasibility.

All of these existing blockchain-based data trading modes are conceptual in nature, with no specific implementation of the relevant algorithms or technologies. These existing concepts and solutions do not provide a complete solution to the problem of the data retention in the data trading center during the data trading. On the other hand, our proposed the blockchain-based data trading mode solution is decentralized, transparent and does not involve the trusted third parties, which can solve the problem of the data retention in the data trading center. In addition, our solution features the automatic payment and offers the refunds and compensations in the event of a data availability controversy.

III. PROPOSED BLOCKCHAIN SOLUTION

In this section, we will introduce and prove the blockchain solution that the data trading center cannot retain the data of trading during the data trading process. Based on the Ethereum blockchain, smart contract, similarity learning and IPFS, the scheme realizes that the data trading center cannot retain the data in the process of data trading, and has the function of automatic payment and can handle the controversy of data availability.

A. DATA TRADING MODE OVERVIEW

The proposed blockchain solution focuses on the transfer of trading data between the data owner and the data purchaser, and the data trading center cannot retain the data. We assume that the data trading centers, data owners, data purchasers, and data storage service providers are all the Ethereum users who have the Ethereum addresses and know how to create and publish the smart contracts, perform the functions of smart contracts and execute the transactions, and contribute to communicate in the chain through the smart contracts. In addition, all the participating entities must agree to the terms and conditions for the data trading in the Ethereum smart contract, with the IPFS hash of these terms and conditions as part of the smart contract created. Since it is expensive to store large chunks of data on the blockchain, and IPFS is a decentralized, open source, point-to-point method for storing large amounts of data in an effective way, so the blockchain is only used to store the hash which is provided by the IPFS for the storage files [21]. If all the participating entities agree on the terms and conditions of the data trading, they can participate in the trading. After the trading starts, the smart contract will collect the trading deposit from the data owner and the data purchaser to ensure the behavior honesty of the entity during the trading. All the participating entities are summarized as follows:

1. Data trading center (DTC): A professional organization for the data trading that can create a smart contract for the data trading to ensure the professionalism.
2. Data owner (DO): The entity that owns the data resource and is willing to make the data trading.
3. Data purchaser (DP): The entity that has a need for the data resource and is willing to acquire it through the data trading.

4. Data storage service provider (DSSP): The data storage service provider provides a file server to store the data resource. After confirming the data trading, it provides the service for the data purchaser to download the purchased data resource.
5. Arbitration institution (ARB): An arbitration institution is a trusted entity of a data trading center, data owner, data purchaser, and data storage provider. The arbitration institution has the ability to prove that the smart contract is to ensure that the code complies with the terms and conditions of the data trading agreed between all the participating entities, and the arbitration institution's address will be included in the smart contract. In addition, the arbitration institution also has the function of resolving the dispute over the availability of trading data between the data purchaser and the data owner through the machine learning algorithm.

The data trading center creates a smart contract for the data trading and has the authority to execute the functions of smart contract, then the arbitration institution verifies the conditions and terms of smart contract. If all the participating entities agree to the terms and conditions of the contract, the data owner and the data purchaser can conduct the data trading. The data purchasers broadcast the data resource requirements to the entire network, the data owners monitor the network, and the data owners with the data resources respond to the data purchasers through the Ethereum addresses. Then the data purchaser verifies whether the owner of the Ethereum address has the required the data resource through the smart contract, and confirms the real owner of the Ethereum address through a challenge-response mechanism. After the verification, the data purchaser sends a request for purchasing data and pays the data price to the smart contract, then the data owner pays the trading deposit that the same amount as the data price. Then the smart contract automatically generates the only token that can download the data resource from the data storage service provider and has a valid time. After the data purchaser downloads the data resource, the data storage service provider will send the data resource download completed information to the smart contract, and then the data purchaser will send the download data resource confirmed information to the smart contract to end the trading and settle the payment. Finally, if the trading is successful, the smart contract will return the deposit paid by the data owner and pay the share of the profits of the trading parties deserve. If the trading is controversial, the trading parties wait for the arbitration institution to intervene and wait for the arbitration result to decide whether the trading is successful or not. The data trading mode is shown in Figure 3.

B. SMART CONTRACT'S FUNCTION

The code within the smart contract provides the following functionality.

1. addDO(DO.EOA, DO.DataName, DO.DataOverview): This function can only be performed by the owner of smart contract to add the data owners and their own data resources related information to the smart contract. It takes the data owner's Ethereum address, the data resource name and the data resource overview as the input parameters.
2. removeDO(DO.EOA): This function can only be performed by the owner of the smart contract to delete the information about the data owner and his/her own data resource from the smart contract. It takes the Ethereum address of the data owner as the input parameter.
3. requestGetData(): This function can only be performed by the data purchasers, who can send the data purchase requests and pay data prices through this function.
4. payDeposit(DP.EOA): This function is executed by the data owner. With the Ethereum address of data purchaser as the input parameter, the trading deposit can be paid by executing this function.
5. purchaserRefund(): This function can only be performed by the data purchaser, who can refund the data price paid before the data owner pays the trading deposit.
6. generateDownloadDataToken(DP.EOA, TokenValidity): This function can only be invoked within the smart contract itself. This function takes the Ethereum address of the data owner and the token validity period as the input parameters. The function outputs the input and generates the token that can download the data resource.
7. sendDownloadedInformation(DP.EOA): This function can only be performed by the data storage service provider, which takes the Ethereum address of data purchaser as the input parameter and outputs the input. When the function has executed, the message that the data purchaser has completely downloaded the data resource will be sent to the smart contract.
8. DPComfirmedResult(): This function can only be performed by the data purchaser. It takes the integer value as the input parameter, such as 1 represents a successful trading, 2 represents the data purchaser cannot download controversy, and 3 represents the data is unavailable controversy.
9. downloadControversyResolutionAndPayment(DP.EOA, bool): This function can only be performed by the arbitration institution, which takes the Ethereum address of the data purchaser and the Boolean value of the arbitration result as the input parameters. This function resolves the dispute over whether the data purchaser has successfully downloaded the data resource to decide whether to pay or refund.
10. availabilityControversyResolutionAndPayment(DP.EOA, bool): This function can only be performed by the arbitration institution, which takes the Ethereum address of the data purchaser and the Boolean value of the arbitration result as the input parameters. This function resolves the data purchaser has the data availability controversy to decide whether to pay or refund and compensate.
11. settlementAndPayment(DP.EOA): This function can only be invoked in the smart contract itself. The function takes the Ethereum address of the data purchaser as the

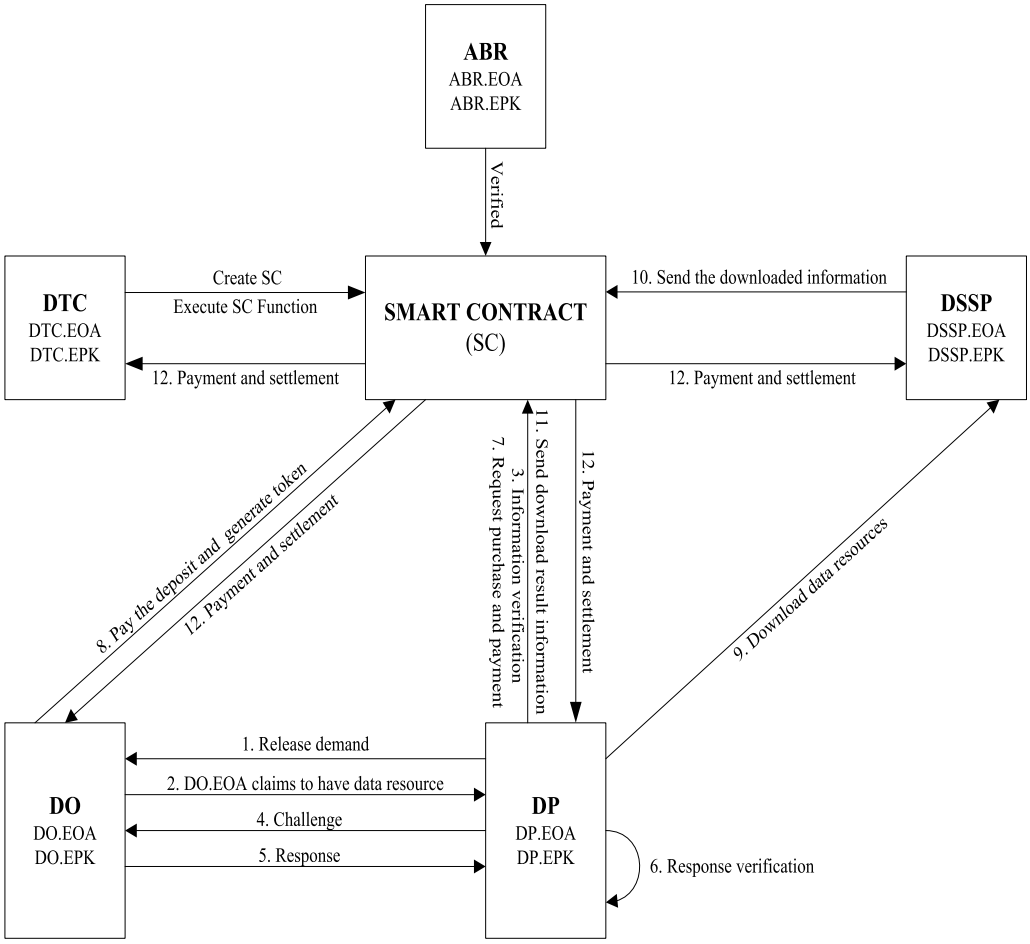

FIGURE 3. The data trading mod.

input parameter. After executing this function, the data trading is completely successful. The data owner will return the trading deposit through the smart contract and get a share of the profits. At the same time, other participants will also receive a share of the profits.

12. changeSmartContractStatus(): This function can only be performed by the owner of the smart contract to disable the smart contract. Once the smart contract is deployed on the blockchain, it will always exist, so the instructions for the use of smart contract are very important.

C. CHALLENGE RESPONSE MECHANISM

When the data purchaser broadcasts the required data resource to the whole network, the data owner responds and claims to own the data resource, then the challenge response mechanism is executed. The structure of the mechanism is summarized as follows:

1. Claim: The data owner use the Ethereum address DO. EOA to claim to the data purchaser that he/she has the data resource.
2. Information Verification: In order to verify whether the DO.EOA really owns the required data resource and to verify who the real owner of the DO.EOA is, the data purchaser can access the smart contract through the interface of smart contract to verify the confirmation.
3. Challenge: The data purchaser chooses arbitrary information I and requests the signature of the data owner.
4. Response: The data owner signs the arbitrary information I with the DO.EOA and the DO.EPK. The signature is defined by S = Sign (DO.EOA, DO.EPK, I), so only the data owner with the DO.EPK can create the correct S. The data owner then sends S to the data purchaser.
5. Response Verification: After the data purchaser receives the signature S from the data owner, the ResponseVerify (DO. EOA, I, S) function is used to verify the signature. If the validation is successful, the data owner can send the purchase data resource request to the smart contract and pay the data price.

The challenge response mechanism can be executed offline, and the signature and verification functions of the source code of the Ethereum can also be used [22]. The signature verification of the real owner of the DO.EOA is shown in Figure 4 and Figure 5.

FIGURE 4. Signing a message.

FIGURE 5. Verifying a signature.

D. DATA RESOURCE DOWNLOAD PROCESS

If the data resources are stored in the blockchain, the high storage cost will be unbearable [23]. Therefore, the feasible solution is that the data purchasers can download the purchased data resources from the data storage service providers by an off-chain downloading. To download a data resource from a data storage service provider, you need to get a token for downloading the data resource. The token is generated by the generateDownloadDataToken () function in the smart contract after the data purchaser requests to purchase the data and pay the data price as well as the data owner pays the same deposit as the data price. The generated token is unique to the data purchaser. The token is a hash created using keccak 256 with built-in Solidity [24] functionality. We will use the data purchaser's Ethereum address (DP.EOA), the data owner's Ethereum address (DO.EOA), the data trading center's Ethereum address (DTC.EOA), the data resource name (DO.DataName), the block time-stamp (BTS) and the token validity (TV) as the generation token's hash components. Hence, the token is unique to each data purchaser, that is, token = keccak 256 (DP.EOA, DO.EOA, DTC.EOA, DO.DataName, BTS, TV).

The token generated above is used by the data purchaser for the data download authentication when the data purchaser accesses the file server of the data storage service provider. While the data download authentication, the message communicated between the data purchaser and the data storage service provider is connected with the signature, which is done using the data purchaser's private key (DP.EPK), namely DPSigned (DPMsgDSSP) = DP.EPK (Hash (DPMsgDSSP)). The message sent by the data purchaser to the data storage service provider connected with his/her signature includes the data purchaser's token (DP.token), the Ethereum address (DP.EOA), the message sent the time-stamp (DP.TS) and the internet protocol address (DP.IP), and the data storage service provider's Ethereum address (DSSP.EOA), i.e. DPMsg DSSP = DP.token & DP.EOA & DP.TS & DP.IP & DSSP.EOA.

The data storage service provider will validate the message received from the data purchaser using the information received through the smart contract. After verifying the message sent by the data purchaser, the data storage service provider will reply to a message, which includes the internet protocol address (DSSP.IP), the message verificated the time-stamp (DSSP.TS), the Ethereum address (DSSP.EOA), and the Ethereum address of the data purchaser (DP.EOA), namely DSSPMsgDP = DSSP.IP & DSSP.TS & DSSP.EOA & DP.EOA. This reply message is signed by the data storage service provider's private key and sent to the data purchaser, namely DSSPSigned (DSSPMsgDP) = DSSP.EPK (Hash (DSSPMsgDP)). The purpose of the reply message is that the data purchaser can verify the data storage service provider. After the mutual validation is successfully completed, the open Secure Socket Layer (SSL) connection is used for the data exchange and the data resource is transferred from the data service provider to the data purchaser. The data resource download process is shown in Figure 6.

E. SIMILARITY LEARNING

After the data purchaser purchases the data, besides the controversy about the success of the download, there is also a controversy about the availability of the purchased data, that is, whether the actual data is consistent with its claims. Therefore, the arbitration institution can judge the availability of the data and arbitrate the controversy through the similarity learning in the machine learning, i.e. the distance metric learning. For decades, the distance metric learning has been extensively studied, which can greatly improve the performance of the classification, the clustering and the retrieval tasks [25]. Therefore, it is widely used in the computer vision, the information retrieval systems and the bioinformatics [26]. The distance metric learning is to learn the distance relationship between the specified data from the pairwise similar or dissimilar points. To determine the availability of the data, we need to verify the distance between the features of the actual data and its declared the features.

In the data availability verification, we can define the similarity of the specified data features and effectively learn the distance function $d(X, Y)$. Let $X_i \in \mathrm{N}^m$ represents the actual data eigenvectors, where m is the number of eigenvectors. Let $X = \{X_1, \ldots, X_n\}$ represents the sample set of the actual data feature points, where n is its size. Let $Y_i \in \mathrm{N}^m$ denotes the declared data eigenvectors, where m is the number of eigenvectors. Let $Y = \{Y_1, \ldots, Y_n\}$ denotes the sample set

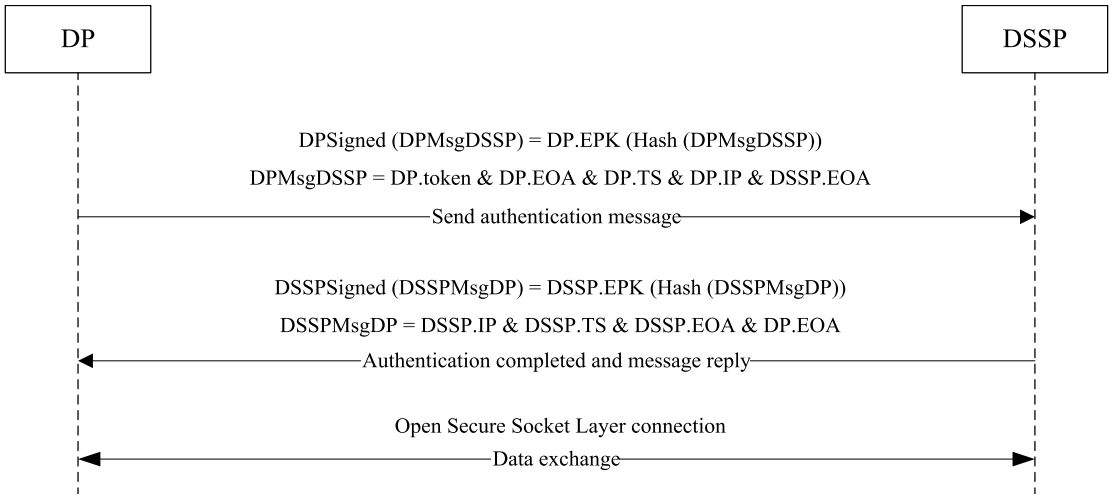

FIGURE 6. The data resource download process.

of the declared data feature points, where n is its size. The Mahalanobis distance [16] between any two vectors X, Y can be expressed as $d_M(X, Y) = \sqrt{(X-Y)^{\mathrm{T}}M(X-Y)}$, where M is the parameter matrix of the distance metric. We use the distance metric learning to calculate the similarity between the actual data features and their declared data features. When there are many metric learning tasks, the distance metric learning algorithm can be extended to the multi-task metric learning settings [27]. In the multi-task metric learning setting [16], the distance of metric learning task t is defined as $d_t(X, Y) = \sqrt{(X-Y)^{\mathrm{T}}(M_0+M_t)(X-Y)}$, and the specific regularization term is defined as $\min_{M_0,\ldots,M_T} \gamma_0||M_0-I||_F^2 + \sum_{t=1}^{T}\gamma_t||M_t||_F^2$, where I represents the identity matrix and F represents the Frobenius norm, that is, the square root of the square sum of the absolute values of matrix elements, and the parameter γ_t controls the regularization of the M_t, where $T \in \mathrm{N}, t = 0, \ldots, T$.

The multi-task metric learning can effectively improve the distance metric performance of the similarity learning [28]. The arbitration institution can use the machine learning algorithm off-chain for the final arbitration of the data availability controversy.

IV. SMART CONTRACT'S IMPLEMENTATION AND TESTING

The Remix IDE [29] is an integrated environment for the smart contract writing and debugging using the Solidity programming language that is the most widely developed language for writing the smart contract, so this paper will use the Remix IDE for the smart contract writing and debugging. This section focuses on the implementation and testing of the smart contract.

A. IMPLEMENTATION

The data purchaser first broadcasts the required data resource to the whole network of the blockchain. The data owner responds to the data purchaser through the Ethereum address. The data purchaser uses the smart contract to query the authenticity of the response, and verifies the digital signature of the data owner through the challenge response mechanism, so as to confirm whether the data owner is the real owner of the Ethereum address. After the data owner authentication is completed, if the data purchaser agrees to the conditions and terms of the smart contract, he/she can make a request to purchase the data from the smart contract and pay the data price as a credit deposit. Then the data owner will pay the same credit deposit as the data price, which plays a restraining role in the activity honesty to each participant. After the credit deposit is paid, the smart contract will automatically create a unique token with timeliness for the data purchaser and send it to the data purchaser. The data purchaser uses the token to download the data resource by an off-chain way from the data storage service provider. After the data purchaser downloads, the data storage service provider will confirm the download by calling the function in the smart contract, and send the confirmation information to the smart contract. After the confirmation of the data storage service provider is completed, the data purchaser will confirm the download by calling the function in the smart contract, and send the confirmation information to the smart contract.

In the process of confirming the data trading, if the data purchaser is satisfied, the payment will be settled. At this time, the data owner will receive the returned credit deposit and the share of profits, and other participants will also share in the profits. However, if the data purchaser is not satisfied, for example, the data downloaded by the data purchaser does not conform to its description in terms of the availability, then the arbitration institution will acquire the token of the data purchaser and use it to download the same data resource. Then, the arbitration institution will decide whether the data availability is consistent with its description by using the similarity learning method off-chain. The arbitration result of the arbitration institution will determine whether the data purchaser is entitled to a refund and the credit deposit of the

data owner. In addition, if the data owner fails to pay the trading credit deposit in time within the valid data trading time range, the data purchaser may request a refund. The data trading sequence diagram is shown in Figure 7.

The sequence diagram describes the complete process from the publication of the data purchaser's requirements to the completion of the settlement and payment, as well as the functions called and events occurred in the smart contract. Among them, Situation 1 describes the satisfaction of the data purchaser, the parties involved in the trading are not in dispute, and the data trading is successful, thus the settlement and payment. Situation 2 describes a data purchaser who claims to be unable to download the data successfully and requests a refund. At this time, the arbitration institution intervenes to download the same data through the same token. If the download is successful, the settlement payment will be made as usual; if the download is unsuccessful, the data purchaser will get a refund. Situation 3 describes that the data purchaser claims that the downloaded data availability does not match its description. At this time, the arbitration institution intervenes and verifies the data availability through the similarity learning method. If the data availability is consistent with its description, the payment will be settled as usual. If the data availability is not consistent with its description, the data purchaser will get a refund and obtain a credit deposit of the data owner to the trading.

Next, the important algorithms used in the code will be described in detail:

1) REQUEST FOR PURCHASE OF DATA RESOURCE

The data purchaser first uses the IPFS hash provided by the smart contract to check the terms and conditions of the data trading contract for purchasing the data resource. If the terms and conditions of the data trading contract are agreed, the data purchaser requests purchase to the data resource by paying the data price. After the data purchaser pays the data price, the data owner pay the same trading credit deposit as the data price. After all the participants have paid the deposit, the smart contract generates a timeliness and unique token that can send it to the data purchaser and download the data resource. After the data storage service provider verifies the identity of the data purchaser through the token, the data purchaser can begin to download the data resource. It is noteworthy that the download of the data resource is completed by an off-chain way. The algorithm of the data purchaser requesting to purchase the data resource is shown in Algorithm 1.

Algorithm 1 Request for Purchase of Data Resource

Input: EOA_1 data price, contractStatus, DP status;
Output: DP has paid data price.
1: *EOA is the se of all the Etherum addresses stored in the smart contact.*
2: *The access restriction for the data purchasers is DP.EOA* $\notin$ *EOA.*
3: *dataPrice is the contact's status.*
4: *contractStatus is the contract's status.*
5: *DPStatus is the data purchaser's status.*
6: **if** msg. value = dataPrice **then**
7: **if** contractStatus = waiting for DP **then**
8: *SC.EOA receives msg.value ether from DP.EOA.*
9: *DPStatus changes to that of DP who has paid the dataPrice.*
10: *Create a notification that DP has successfully paid the dataPrice.*
11: **else**
12: *Revert contract status and display errors.*
13: **end if**
14: **else**
15: *Revert contract status and display errors.*
16: **end if**

2) PAYMENT AND SETTLEMENT FOR SUCCESSFUL TRADING

Once the data purchaser downloads the data resource from the data storage service provider, the data storage service provider notifies the smart contract by calling the function in the smart contract. However, the completion of payment settlement also requires the data purchaser to notify the smart contract by calling the function in the smart contract that there is no dispute about the data trading. He/She is satisfied with the trading, and then the smart contract confirms the trading information, and then the trading is completed. Then the data owner will receive the returned credit deposit and the share of profits, and other participants will also share in the profits. The algorithm of the payment and settlement for the successful trading is shown in Algorithm 2.

3) PAYMENT AND SETTLEMENT FOR CONTROVERSIAL TRADING

a: CONTROVERSY OVER THE SUCCESS OF DATA DOWNLOAD

If the confirmation result of the data purchaser is the data fails to download successfully, then the arbitration institution intervenes in the arbitration, and the arbitration institution downloads the data from the data storage service provider by obtaining the token of the data purchaser. If the arbitration institution can download the data normally, the arbitration result will be a successful trading and pay the settlement. If the arbitration institution cannot download the data normally, the arbitration result will be a unsuccessful trading. At this time, the data purchaser will be refunded, and the data owner's deposit will also be refunded. The algorithm for dealing with the controversy about the success of data downloading is shown in Algorithm 3.

b: DATA AVAILABILITY CONTROVERSY

If the confirmation result of the data purchaser is that the data is not available, then the arbitration institution intervenes in the arbitration. The arbitration institution uses the similarity

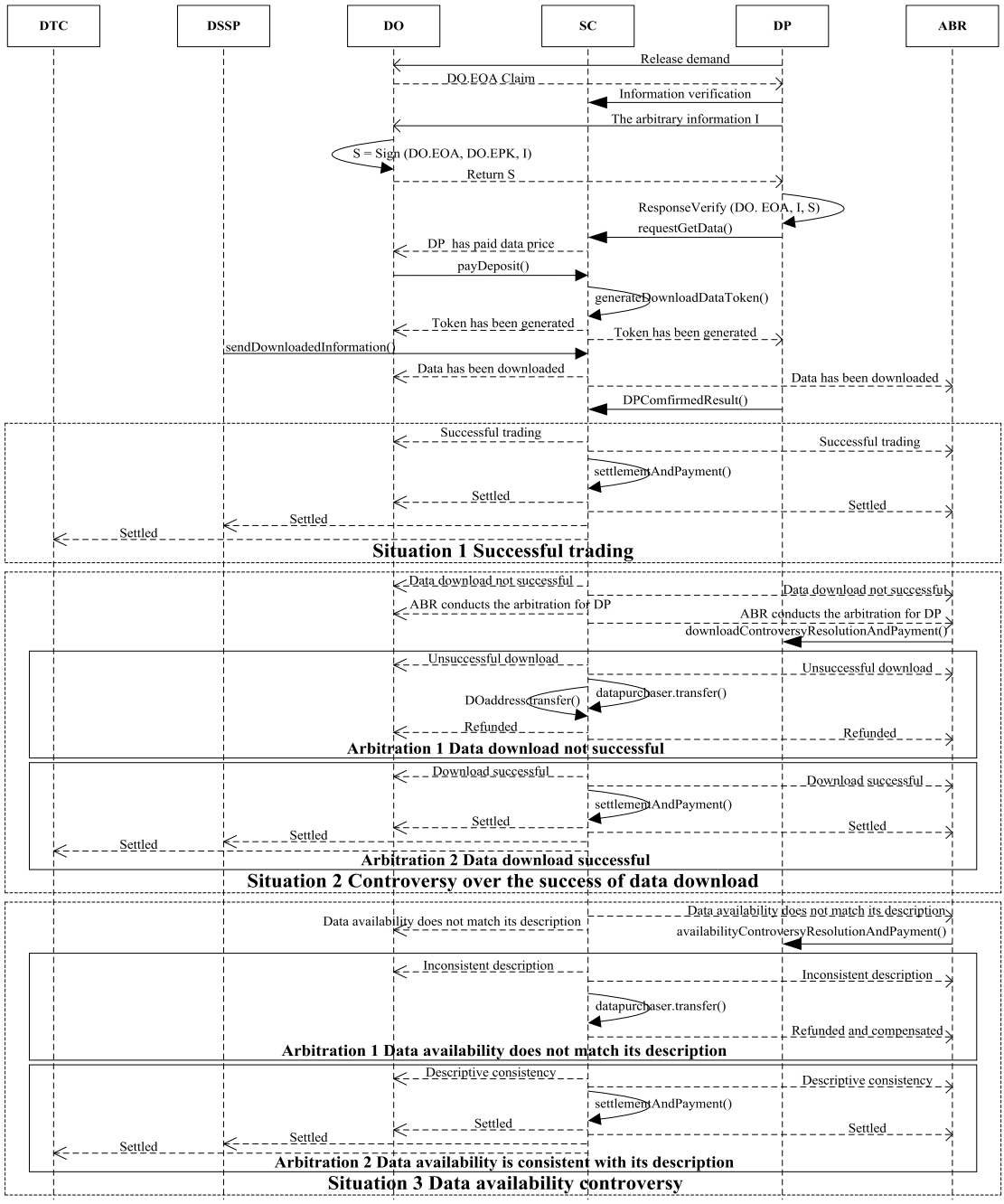

FIGURE 7. The data trading sequence diagram.

learning algorithm by an off-chain way to check whether the data is consistent with its description. If the verification result is that the data is consistent with its description, the arbitration result is the successful trading and pay the settlement; if the verification result is that the data is inconsistent with its description, the arbitration result is the unsuccessful trading and the data owner is suspected of the fraud. At this time, the data purchaser can not only get a refund, but also get the trading deposit paid by the data owner as the fraud compensation. The algorithm of the data availability controversy processing is shown in Algorithm 4.

B. TESTING AND VALIDATION

In the part of testing and validating the smart contract, we will build the private chain of the Ethereum. Through the Ethereum wallet, we will test four main functions, including a data purchaser paying the trading data price, a data owner paying the trading deposit, successful trading payment settlement, disputed trading payment settlement. The Ethereum addresses of the data trading center, the data owner, the data storage service provider and the arbitration institution are "0xd813351258D8A53314E55b12c3Cf11C98dA8E7D4", "0x09BdFdBAc10253e988b4c7197f0faf44Ea7F8479",

Algorithm 2 Payment and Settlement of Successful Trading

Input: EOA,DSSP,EOA,DP,EOA,ETC,EOA,DO,EOA, *result* **Output**: *payment and setrlement done successfully*,
1: *EOA is the set fo the Ethereum addresses of the parties involved in the trading*.
2: *result is the confirmation result of the data parchaser*.
3: **if** result == 1 or 2 or 3 **then**
4: **if** result == 1 **then**
5: *A successful trading*.
6: *Return back the deposit of DO.EOA*.
7: *Pay DTC.EOA,DSSP.EOA and DO.EOA's the share of the profits*.
8: **else**
9: **if** result == 2**then**
10: *The arbitration institution intervenes in the data download controversy*.
11: *Turn to Algorithm 3*.
12: **else**
13: *The arbitration institution intervenes in the data availability controversy*
14: *Turn to Algorithm 4*.
15: **end if**
16: **end if**.
17: *The current status of DP is that the trading has been completed*.
18: **else**
19: *Revert contract status and display errors*.
20: **end if**

"0x380851065aBf6F833eFBE9e059aF9D2F31baCD54" and "0xd06B32822f5F1838E0Bb05CBEC803889eFDd9380" respectively. The smart contract implements the restrictive functions because it contains modifiers in the source code, which limits the specific functions of the different entities. All the functions can only be performed by a specific entity with a specific Ethereum address. All the restricted functions with modifiers have been successfully tested. If a specific entity without a specific Ethereum address requests to perform the functions in the smart contract, the execution will fail, and the behavior will not be recorded in the smart contract and blockchain.

1) REQUEST FOR PURCHASE DATA

The data purchaser uses the requestGetData() function to pay the data price, at which time the data purchaser is in the paid status. We use the Ethereum address of the data purchaser is "0x9549E34316ab06B205711B2eF1Ea5D078C6e8E5f". Our initial data price is 3 ether, so the data purchaser pays 3 ether. At this point, the data purchaser's data purchase request has been completed. Figure 8 (a) shows the original balance of the data purchaser, Figure 8 (b) shows the request purchase data function, Figure 8 (c) shows the balance of the data purchaser after the data price has been paid. The balance is less than 17 ether because of the need to pay the gas when the execution of the smart contract function, which indicates that the data purchaser has successfully requested.

Algorithm 3 Controversy Over the Success of Data Download

Input:*EOA,ABR,EOA,DP.EOA*
Output:*Pay the settlements V Refund*
1: EOA is *the set of the Ethereum addresses of the parties involved in the trading.*
2: it DP.EOA's confirmation result == 2 **then**
3: *Create a notification to send to ABR.*
4: ABR *uses the token E DP.EOA to download the data,*
5: AR *is the data download arbitration result,*
6: if AR = true **then**
1: The result of arbitration is that the data cannot download,
8: *Return back the data price paid by DP.EOA.*
9: *Return the deposit paid by DO.EOA.*
10: **else**
11: *The result of arbitration is that the data* can *download,*
12: *Return back the deposit ofDO.EOA.*
13: *Pay DTC.EOA. DSSP.EOA and DO.EOA s the share of the profits,*
14: **end if**
15: *The current status of DP is that the trading has been completed.*
16: **else**
17: *Revert contract status and display errors.*
18: **end if**

2) PAYMENT OF THE TRADING DEPOSIT

The data owner pays the deposit by using the payDeposit() function, at which time the data purchaser status shows that the data owner trading deposit has been paid. After the payment of the trading deposit is completed, the token required for downloading data is automatically generated by calling the internal function of the smart contract. Figure 9 (a) shows the original balance of the data owner. Figure 9 (b) shows the balance of the data owner after the payment of the trading deposit. The balance of the data owner is less than 27 ether due to the need to pay the gas for the execution of the smart contract. This indicates that the data owner has successfully paid the trading deposit. Figure 9 (c) shows the occurrence of the token generation event, indicating that the token has been automatically generated.

3) PAYMENT AND SETTLEMENT OF A SUCCESSFUL TRADING

When the data storage service provider sends the confirmation message that the data has been downloaded and the data purchaser sends the confirmation message that the downloaded data has no controversy, this means that the trading is successful and the payment will be settled automatically. Figure 10 (a) shows the balance of the data trading center, the data owner and the data service provider before the trading, and Figure 10 (b) shows the balance of every party after the data trading success. Figure 10 (c) shows the occurrence of the settlement and payment event, indicating

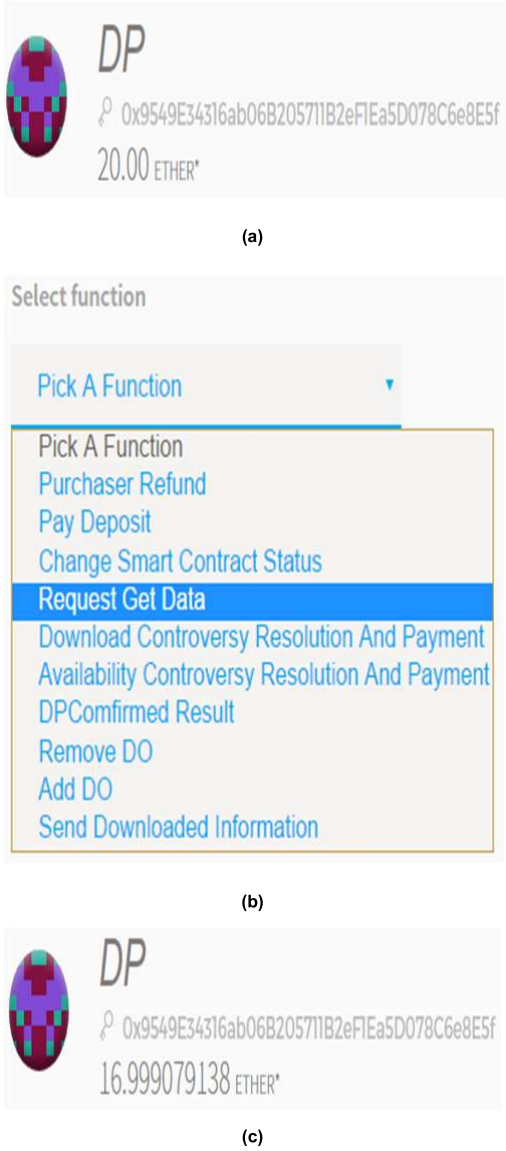

FIGURE 8. (a). The original balance of the data purchaser. (b). The request purchase data function. (c). The balance of the data purchaser after the data price has been paid.

that settlement and payment have been completed. Because the owner of the smart contract is the data trading center, and the private chain has been mining, so the ether coming from the mining has been rising, which result to it is not intuitive to show the income value of the data trading center.

4) DISPUTED PAYMENT SETTLEMENT

a: DATA CANNOT BE DOWNLOADED CONTROVERSY

If the confirmation message sent by the data purchaser is that the data cannot be downloaded, the arbitration institution intervenes in the arbitration. If the arbitration institution can download the data by the off-chain way, the arbitration result will be false, and the payment and settlement will be automatically completed. If the arbitration institution cannot download the data, the arbitration result will be true, and the refund will be automatically completed. Figure 11 shows that the arbitration result is true, the event of automatically completing the refund has occurred.

Algorithm 4 Data Availability Controversy

Input:*EOA.ABR.EOA,DP.EOA*
Output: *Pay the settlement V refund and Compensate*
1: *EOA is the set of the Etherum addresses of the parties involved in the trading.*
2: if DP.EOA's *confirmation result* == 3 then
3: *Create a notification to send to ABR.*
4: *ABR uses the similarity learning to verify the data.*
5: *AR is the data verification arbitration result.*
6: **if** AR = true **then**
7: *The result of arbitration is that the data is unavailable.*
8: *Return the data price paid by DP.EOA.*
9: *Transfer the trading deposit of OD.EOA to DP.EOA*
10: **else**
11: *The result of arbitration is that the data is available.*
12: *Return back the deposit of DO.EOA.*
13: *Pay DTC.EOA.DSSP.EOA and DO.EOA's the share of the profits.*
14: **end if**
15: *The current status of DP is that the trading has been completed.*
16: **else**
17: *Revert contract status and display errors.*
18: **end if**

b: DATA UNAVAILABILITY CONTROVERSY

If the confirmation information sent by the data purchaser is that the data is inconsistent with its description, the arbitration institution intervenes in the arbitration. The arbitration institution uses the machine learning algorithm to verify the data by the off-chain way. If the arbitration result is true, the refund will be automatically completed. It is worth noting that in order to punish the dishonest behavior of the data owner, the deposit will not be returned and the deposit paid by the data owner will be paid to the data purchaser. If the arbitration result is false, the payment and settlement will be automatically completed. Figure 12 (a) shows the original balance of the data owner and the data purchaser. Figure 12 (b) shows the balance of the automatic refund and compensation when the arbitration result is true. Figure 12 (c) shows that the automatic refund and compensation event has occurred when the arbitration result is true.

V. SECURITY ANALYSIS AND PERFORMANCE EVALUATION

A. SECURITY ANALYSIS

This section discusses the security of our proposed blockchain solution in the data trading process. A data purchaser accesses the data storage service provider off-chain

FIGURE 9. (a). The original balance of the data owner. (b). The balance of the data owner after the payment of the trading deposit. (c). The occurrence of the token generation event.

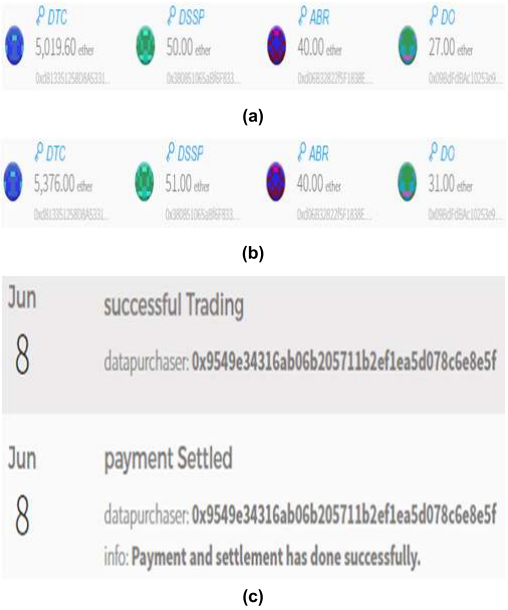

FIGURE 10. (a). The balance of the data trading center, the data owner and the data service provider before the trading. (b). The balance of every party after the data trading success. (c). The occurrence of the settlement and payment event.

by using the time-stamp and the unique token to ensure that only authorized the data purchaser can download the data resource through the secure SSL session, which has good

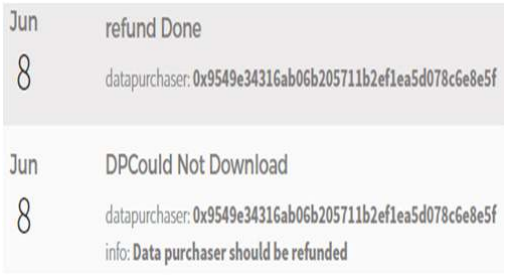

FIGURE 11. The event of automatically completing the refund has occurred.

FIGURE 12. (a). The original balance of the data owner and the data purchase. (b). The balance of the automatic refund and compensation. (c). The automatic refund and compensation event has occurred.

confidentiality and can prevent replay attacks and man in the middle (MITM) attacks. The blockchain is tamper-proof and traceable, and the events occurring in the blockchain are fully recorded in the log. Therefore, the operation information of all entities in the data trading process is fully recorded, which has good integrity and can prevent all entities from denying their own behavior in the trading process. In order to ensure the security of data trading, we also propose a challenge response mechanism to authenticate and authorize the data owner in order to prevent the dishonest behavior of the data owner. At the same time, we propose a mechanism to access and download the data by the off-chain way to authenticate and authorize the data purchaser in order to prevent the dishonest behavior of the data purchaser.

B. PERFORMANCE EVALUATION

In this section, we discuss the flexibility of deploying the smart contract in the Ethereum wallet, disabling the smart contract and executing the functions of smart contract, and

TABLE 1. The gas cost for deploying the smart contract and performing the functions.

Function	Gas Used	Gas Cost (ether)
Deploy contract	2,649,904	0.047698272
addDO()	162,468	0.002924424
removeDO()	39,413	0.000709434
RequestGetData()	21,159	0.000380862
payDeposit()	61,005	0.00109809
purchaserRefund()	53,485	0.00096273
sendDownloadedInformation()	31,752	0.000571536
DPComfirmedResult(result == 1)	61,845	0.00111321
DPComfirmedResult(result == 2)	33,585	0.00060453
downloadControversyResolutionAndPayment(true)	48,462	0.000872316
downloadControversyResolutionAndPayment(false)	62,190	0.00111942
DPComfirmedResult(result == 3)	34,144	0.000614592
availabilityControversyResolutionAndPayment(true)	40,799	0.000734382
availabilityControversyResolutionAndPayment(false)	62,278	0.001121004
changeSmartContractStatus (true)	28,957	0.000521226
changeSmartContractStatus (false)	23,675	0.00042615

estimating the gas cost required. The data trading center only needs to follow the proposed system's Ethereum protocol to create the smart contract and perform the functions, or disable the smart contract, with good flexibility to join or leave the system. Other participating entities, as the users of the Ethereum, can also participate in the system freely. The gas cost for deploying the smart contract and performing the functions are shown in Table 1.

VI. CONCLUSION AND FUTURE WORK

In this paper, we propose a solution and framework of the data trading mode based on smart contract using blockchain and machine learning. Our solution can be used to solve the problem that the data trading center has the ability to retain the data in the traditional data trading mode, so as to protect the rights and interests of the data owner and promote the development of the data trading. Data sales and downloads, data owner's authentication and authorization, data purchaser's authentication and authorization of downloading data off-chain, dispute handling during data trading, automatic payment of trading completion, penalty setting of dishonest behavior are all controlled by various mechanisms and algorithms, which have been programmed in the Remix IDE and tested in the Ethereum wallet. At the same time, we have demonstrated how to implement and test all the functions of smart contract, and provided the security analysis and performance evaluation.

As future work, we will further research how to prevent the data purchaser from resale after data sales. At present, signing a non-resale contract is the solution to the problem of data resale, but signing a contract cannot eliminate the problem of data resale. Therefore, we can design a smart contract to ensure that data will not be resold after sale, which will be the direction of further research in the future.

REFERENCES

[1] F. Liang, W. Yu, D. An, Q. Yang, X. Fu, and W. Zhao, "A survey on big data market: Pricing, trading and protection," *IEEE Access*, vol. 6, pp. 15132–15154, 2018.

[2] X. Cao, Y. Chen, and K. J. R. Liu, "Data trading with multiple owners, collectors, and users: An iterative auction mechanism," *IEEE Trans. Signal Inf. Process. Over Netw.*, vol. 3, no. 2, pp. 268–281, Jun. 2017.

[3] M. J. Yang, "A design of data trading platform based on cryptology and blockchain technology," *Inf. Commun. Technol.*, vol. 10, pp. 24–31, 2016.

[4] H. R. Hasan and K. Salah, "Blockchain-based proof of delivery of physical assets with single and multiple transporters," *IEEE Access*, vol. 6, pp. 46781–46793, 2018.

[5] S. Nakamoto. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System*. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[6] K. Toyod, P. T. Mathiopoulo, I. Sasase, and T. Ohtsuk, "A novel blockchain-based product ownership management system (POMS) for anti-counterfeits in the post supply chain," *IEEE Access*, vol. 5, pp. 17465–17477, 2017.

[7] P. Treleaven, R. G. Brown, and D. Yang, "Blockchain technology in finance," *Computer*, vol. 50, no. 9, pp. 14–17, 2017.

[8] A. Azaria, A. Ekblaw, T. Vieira, and A. Lippman, "MedRec: Using blockchain for medical data access and permission management," in *Proc. Int. Conf. Open Big Data (OBD)*, Aug. 2016, pp. 25–30.

[9] W. Meng, E. W. Tischhauser, Q. Wang, Y. Wang, and J. Han, "When intrusion detection meets blockchain technology: A review," *IEEE Access*, vol. 6, pp. 10179–10188, 2018.

[10] F. Tian, "A supply chain traceability system for food safety based on HACCP, blockchain & Internet of Things," in *Proc. Int. Conf. Service Syst. Service Manage. (ICSSSM)*, Jun. 2017, pp. 1–6.

[11] R. AlTawy, M. ElSheikh, A. M. Youssef, and G. Gong, "Lelantos: A blockchain-based anonymous physical delivery system," Cryptol. ePrint Arch., Tech. Rep. 2017/465, 2017. [Online]. Available: http://eprint.iacr.org/2017/465

[12] K. Biswas and V. Muthukkumarasamy, "Securing smart cities using blockchain technology," in *Proc. IEEE 18th Int. Conf. High Perform. Comput. Commun., IEEE 14th Int. Conf. Smart City, IEEE 2nd Int. Conf. Data Sci. Syst. (HPCC/SmartCity/DSS)*, Dec. 2016, pp. 1392–1393.
[13] *Ethereum is the Blockchain App Platform*. [Online]. Available: https://ethereum.org/
[14] K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the Internet of Things," *IEEE Access*, vol. 4, pp. 2292–2303, 2016.
[15] N. Szabo. (1994). *Formalizing and Securing Relationships on Public Networks*. [Online]. Available: http://www.firstmonday.org/ojs/index.php/fm/article/view/ 548/469
[16] S. Parameswaran and K. Q. Weinberger, "Large margin multi-task metric learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, pp. 1867–1875.
[17] *IPFS is the Distributed Web*. [Online]. Available: https://ipfs.io/
[18] *GitHub is A Hosting Platform for the Open Source and the Private Software Projects*. [Online]. Available: https://github.com/
[19] Y. Z. Wang, J. Hou, and Y. Zhang, "Data management based on block chain technology," *Electron. Des. Eng.*, vol. 27, pp. 87–90 and 95, 2019.
[20] W. J. Lu, Q. Y. Xu, and C. Li, "Research and applications of the coordination mechanism of data exchange in a trusted environment," *Modern Comput.*, pp. 45–51, 2017.
[21] J. Benet, "IPFS—Content addressed, versioned, P2P file system," 2014, *arXiv:1407.3561*. [Online]. Available: https://arxiv.org/abs/1407.3561
[22] J. P. Cruz, Y. Kaji, and N. Yanai, "RBAC-SC: Role-based access control using smart contract," *IEEE Access*, vol. 6, pp. 12240–12251, 2018.
[23] H. R. Hasan and K. Salah, "Proof of delivery of digital assets using blockchain and smart contracts," *IEEE Access*, vol. 6, pp. 65439–65448, 2018.
[24] *Solidity is an Object-Oriented, High-Level Language for Implementing Smart Contracts*. [Online]. Available: https://solidity.readthedocs.io/en/latest/
[25] L. Yang and R. Jin, "Distance metric learning: A comprehensive survey," Michigan State Univ., East Lansing, MI, USA, Tech. Rep., 2006, vol. 2.
[26] A. Bellet, A. Habrard, and M. Sebban, "A survey on metric learning for feature vectors and structured data," 2013, *arXiv:1306.6709*. [Online]. Available: https://arxiv.org/abs/1306.6709
[27] P. Yang, K. Huang, and C.-L. Liu, "Geometry preserving multi-task metric learning," *Mach. Learn.*, vol. 92, pp. 133–175, Jul. 2013.
[28] Y. Zhao, Y. Yu, Y. Li, G. Han, and X. Du, "Machine learning based privacy-preserving fair data trading in big data market," *Inf. Sci.*, vol. 478, pp. 449–460, Apr. 2019.
[29] *Welcome to Remix Documentation*. [Online]. Available: https://remix.readthedocs.io/en/ latest/

WEI XIONG is currently pursuing the Ph.D. degree with the Department of Information Management, School of Management, Shanghai University. His research interests include smart contract, blockchain, machine learning, and information systems.

LI XIONG is currently a full-time Professor with the Department of Information Management, School of Management, Shanghai University. Her main research interests include information systems, cross-border e-commerce, and collaborative innovation.