Meng Shen
Liehuang Zhu
Ke Xu

Blockchain: Empowering Secure Data Sharing

Springer

Blockchain: Empowering Secure Data Sharing

Meng Shen • Liehuang Zhu • Ke Xu

Blockchain: Empowering Secure Data Sharing

Springer

Meng Shen
School of Computer Science
Beijing Institute of Technology
Beijing, China

Liehuang Zhu
School of Computer Science
Beijing Institute of Technology
Beijing, China

Ke Xu
Department of Computer Science
Tsinghua University
Beijing, China

ISBN 978-981-15-5938-9 ISBN 978-981-15-5939-6 (eBook)
https://doi.org/10.1007/978-981-15-5939-6

This Springer imprint is published by the registered company Springer Nature Singapore Pte Ltd.
The registered company address is: 152 Beach Road, #21-01/04 Gateway East, Singapore 189721, Singapore

Preface

With the development of big data, data sharing has become increasingly popular and important in optimizing resource allocation and improving information utilization. However, the expansion of data sharing means there is an urgent need to address the issue of the privacy protection—an area where the emerging blockchain technology offers considerable advantages. Although there are a large number of research papers on data sharing modeling and analysis of network security, there are few books dedicated to blockchain-based secure data sharing.

Filling this gap in the literature, the book proposes a new data-sharing model based on the blockchain system, which is being increasingly used in medical and credit reporting contexts. It describes in detail various aspects of the model, including its role, transaction structure design, secure multiparty computing and homomorphic encryption services, and incentive mechanisms and presents corresponding case studies. The book explains the security architecture model and the practice of building data sharing from the blockchain infrastructure, allowing readers to understand the importance of data sharing security based on the blockchain framework, as well as the threats to security and privacy. Further, by presenting specific data sharing case studies, it offers insights into solving data security sharing problems in more practical fields.

This book is expected to help readers have a better understanding of learning importance of data sharing security from the framework of blockchain and the threats to security and privacy, as well as learning the security architecture model and practice of building data sharing schemes from the blockchain infrastructure. Hopefully, this book can motivate more innovative solutions for solving data security sharing problems in more practical fields.

Beijing, China — Meng Shen
Beijing, China — Liehuang Zhu
Beijing, China — Ke Xu
March 15, 2020

Acknowledgements

The authors would like to acknowledge our editors for their help and guidance, Prof. Mohsen Guizani and Prof. Xiaojiang Du for their insightful and valuable advices, the graduate students at BIT, Xiangyun Tang, Zhang Jie, Huisen Liu, Junxian Duan, Anqi Sang, and Rungeng Sun, for their contributions to the accomplishment of this book.

The authors would like to acknowledge the great support from National Natural Science Foundation of China under Grants 61972039 and 61872041, Beijing Natural Science Foundation under Grant 4192050, China National Funds for Distinguished Young Scientists under Grant 61825204, Zhejiang Lab Open Fund with No. 2020AA3AB04, Beijing Outstanding Young Scientist Program with No. BJJWZYJH01201910003011, and Beijing National Research Center for Information Science and Technology (BNRist) with No. BNR2019RC01011.

The authors dedicate this book to their families for their continuous love and support.

Contents

Chapter 1
Introduction

With the development of big data era, data sharing has become increasingly popular and important, which can optimize resource allocation and improve information utilization. But with the expansion of data sharing, privacy protection need to be solved urgently. This chapter presents the data sharing background first, following with the typical data sharing scenarios. Then a concise review of existing solutions for data sharing is given. Finally, data sharing requirements and challenges are summarized.

1.1 Background

With the increase in the amount of data and hardware computing power, not only the effectiveness of traditional machine learning algorithms has improved dramatically, but powerful new machine learning algorithms (i.e., deep learning) have also advanced. Machine Learning (ML) technology have provided services in many areas such as driverless cars, movie recommendations, and intelligent robots [4, 11], where deep learning solves a variety of difficult tasks previously solved hardly by traditional machine learning.

The amount of training data plays a key role in improving the model accuracy [12, 16, 33]. Experiments performed by Google on 300 million images show that the model's performance increases linearly with the amount of training data [30]. While single institution generally has insufficient data for training ML models, which needs to gather training data from multiple data sources [1–3]. For example, in a product recommendation service based on a machine learning algorithm, the product seller has data on the product and data about the user's purchase, but it does not have data on the user's purchasing ability and payment habits. Thus, the product seller seeks the institutions that own the demanded data for data sharing and better data modeling results. However, the data usually involves user privacy, which prevents the sharing of user data. Besides, the proposed contract clearly stipulates the scope of data transaction and data protection obligations, which poses new challenges to data sharing. European Union implemented *General Data Protection Regulation* on May 25, 2018, which aims to protect users' personal privacy and data security [31]. *Cybersecurity Law of the People's Republic of China* has been implemented in 2017 point out that network operators must not disclose, tamper with, or destroy the personal information they collect [6].

On the other hand, in recent years, many different kinds of blockchain platforms, such as HyperLedger, Ethereum, and EOS, have been proposed and applied to a variety of security application scenarios. Blockchain is an open and distributed ledger taking the form of a list of blocks originally designed for recording transactions in cryptocurrency systems, which enables reliable transactions among a group of untrusted participants. Blockchain has several desirable features, making it inherently suitable for reliable data sharing:

- Decentralized. As a distributed ledger, blockchian is built on a peer-to-peer network and does not require a trusted third party or central administrator. Multiple copies of data recorded in the ledger exist in the system, avoiding data loss at a single point of failure.
- Tamper-resistant. Blockchain uses consensus protocols, i.e., Proof of Work (PoW) to manage the right to create new blocks. As a result, data manipulation is impractical in terms of computational overhead, making the data recorded in the block immutable.
- Traceability. The remaining participants can easily verify transactions between two parties in blockchain system, and any transaction can be tracked.

Although blockchain has multiple advantages on data sharing, the potential attacks on data privacy make it far from perfect for being a data sharing platform. In blockchain, all transactions are recorded in blocks in plaintext, thus exposing sensitive information in transactions to all participants (including opponents) [8, 23, 25, 26]. Therefore, when using blockchain as a data sharing platform, security and privacy issues should be carefully addressed.

Data sharing optimizes resource allocation and improve information utilization, while privacy concern is imminent. As an emerging technology, blockchain has outstanding advantages in solving the privacy issues of data sharing. Although there are a lot of research papers on data sharing and network security, there are few books on blockchain-based secure data sharing.

This book proposes a secure data-sharing model based on the blockchain system. Various aspects of the model, including its role, transaction structure design, secure multi-party computing and homomorphic encryption services, and incentive mechanisms, are described in detail in the following chapters. And corresponding case studies for the functions of the proposed model are provided.

1.2 Typical Data Sharing Scenarios

Data sharing has a long history in various research areas. The roles in data sharing consists of data providers and data users, where data providers generally do not face users directly. For example, Twitter licenses its data to companies Gnip, DataSift, and NTT DATA for sale [7]. At present, the data intermediary in the United States is a big industry, and the scale of transaction data in 2012 has reached 150 billion US dollars [36]. The key to data transactions is the requirement for data quality, including the accuracy, authenticity, integrity, and consistency of data.

Data sharing appears in various industries [24, 37, 38, 42], such as wise medical, Internet of Vehicle, logistics management and credit agencies, etc. In the field of wise medical, for a certain rare disease, the case data owned by a single hospital cannot well support medical research. Sharing rare cases that are held locally by multiple hospitals can improve the effectiveness of medical analysis. In Internet of Vehicle, the vehicle manufacturer, vehicle management agency and vehicle social network application provide services for users together. If one of the Internet of Vehicle service provider desires to utilize user data for analyzing users' vehicle use behaviors, it need to gather data from different data sources of service providers.

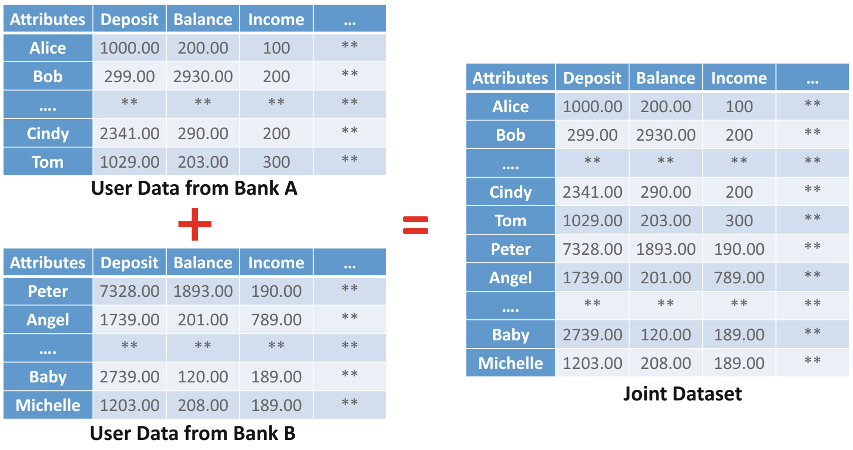
From the perspective of data format, data sharing scenarios can be divided as homogeneous data sharing and heterogeneous data sharing.

- Homogeneous data sharing. Multiple data owners have datasets with the same data format (or same data attributes) and they want to conduct data analysis on their joint datasets.

Two regional banks (Bank A and Bank B) may have very different user groups from their respective regions, and their users intersections are small. But the two regional banks have similar business, which means that the user data are collected from the two regional banks have similar attributes. If they want to conduct an analysis on user data for user market research or other purposes, using their own local data may be insufficient for a comprehensive analysis. The two regional banks could joint the two user datasets to obtain a larger data set for a comprehensive analysis. Figure 1.1 visualizes this homogeneous data sharing of the two regional banks.

- Heterogeneous data sharing. Multiple data owners have datasets with the different data format (or different attributes) and they want to conduct ML model training on their joint dataset.

An example of heterogeneous data sharing have two companies: one is a bank and the other is an e-commerce company. Their user intersections are large. The bank records the users' financial attributes, such as income and balance, etc. The e-commerce company keeps user browsing history and purchase history, etc. The two companies have different attribute spaces. Suppose the e-commerce company wants to construct a product purchase prediction model. Then if the e-commerce company could construct the prediction model on the joint dataset of the bank and



Attributes	Deposit	Balance	Income	...
Alice	1000.00	200.00	100	**
Bob	299.00	2930.00	200	**
....	**	**	**	**
Cindy	2341.00	290.00	200	**
Tom	1029.00	203.00	300	**

User Data from Bank A

+

Attributes	Deposit	Balance	Income	...
Peter	7328.00	1893.00	190.00	**
Angel	1739.00	201.00	789.00	**
....	**	**	**	**
Baby	2739.00	120.00	189.00	**
Michelle	1203.00	208.00	189.00	**

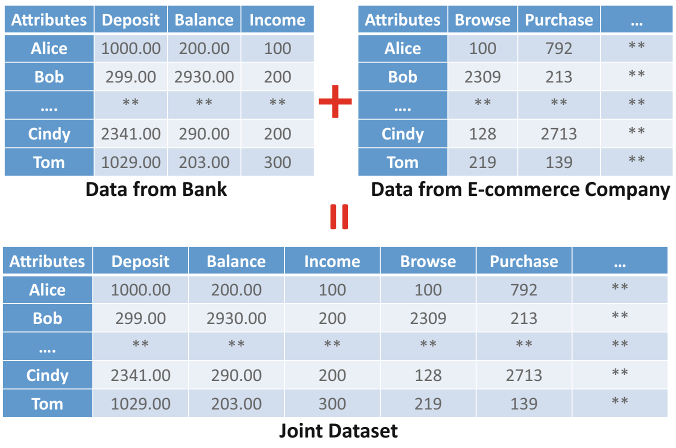
User Data from Bank B

=

Attributes	Deposit	Balance	Income	...
Alice	1000.00	200.00	100	**
Bob	299.00	2930.00	200	**
....	**	**	**	**
Cindy	2341.00	290.00	200	**
Tom	1029.00	203.00	300	**
Peter	7328.00	1893.00	190.00	**
Angel	1739.00	201.00	789.00	**
....	**	**	**	**
Baby	2739.00	120.00	189.00	**
Michelle	1203.00	208.00	189.00	**

Joint Dataset

Fig. 1.1 Homogeneous data sharing



Attributes	Deposit	Balance	Income
Alice	1000.00	200.00	100
Bob	299.00	2930.00	200
....	**	**	**
Cindy	2341.00	290.00	200
Tom	1029.00	203.00	300

Data from Bank

+

Attributes	Browse	Purchase	...
Alice	100	792	**
Bob	2309	213	**
....	**	**	**
Cindy	128	2713	**
Tom	219	139	**

Data from E-commerce Company

=

Attributes	Deposit	Balance	Income	Browse	Purchase	...
Alice	1000.00	200.00	100	100	792	**
Bob	299.00	2930.00	200	2309	213	**
....	**	**	**	**	**	**
Cindy	2341.00	290.00	200	128	2713	**
Tom	1029.00	203.00	300	219	139	**

Joint Dataset

Fig. 1.2 Heterogeneous data sharing

the e-commerce company, the model can be more accurate than constructing on the dataset of the e-commerce company only, which are visualized on Fig. 1.2.

There are two mainly computation modes for data sharing: federated learning and secure multi-party computation.

- Federated learning. Multiple data owners wish to train a machine learning model by consolidating their respective data. In federated learning model, data owners collaboratively train a model without expose their data to each other. In most federated learning methods, data owners collaboratively train models by exchanging model parameters.

- secure multi-party computation. Secure multi-party computation aims at creating methods for parties to jointly compute a function over their inputs while keeping those inputs private. For ensuring security, the methods of secure multi-party computation usually employ homomorphic encryption or differential privacy for protecting participants' privacy from each other without requiring a trusted third party.

1.3 Taxonomy of Existing Solutions

Nowadays, existing solutions for data security protection among collaborative learning in the collaborative learning process through various cryptographic algorithms. With the development of blockchain technology, blockchain has been gradually applied to the process of collaborative learning of multiple parties with its technical advantages such as decentralization and immutability. This section will introduce and briefly analyze current representative studies (Table 1.1).

1.3.1 Multi-party Collaborative Learning Without Blockchain

In recent years, the amount of data in various industries has surged. In order to make great use of this data, different institutions collaborate on machine learning and

Table 1.1 Summary of existing typical solutions

References	Collaborative learning methods	Privacy-preserving techniques
[9]	LIR	Secure multi-party computation
[21]	LOR	Homomorphic encryption
[20]	SVM & LIR & LOR & DL	Secure multi-party computation
[19]	LIR & LOR & DL	Secure multi-party computation
[1]	DL	Differential privacy
[11]	SVM	Homomorphic encryption
[32]	Federated learning	Differential privacy
[17]	Federated learning	Homomorphic encryption
[10]	Federated learning	Differential privacy
[13]	Federated learning	Blockchain
[14]	Federated learning	Blockchain
[43]	Federated learning	Blockchain
[18]	Federated learning	Homomorphic encryption & Blockchain
[27]	SVM	Homomorphic encryption & Blockchain
[28]	SVM	Homomorphic encryption & Blockchain
[35]	DL	Threshold Paillier & Blockchain

deep learning in a secure manner. At the same time, many corresponding studies are under research. These studies focus on different machine learning and deep learning methods, including linear regression (LIR) [5], logistic regression (LOR) [12, 41], naive bayes [16], support vector machines (SVM) [34, 40], deep learning (DL) [29], and so on. In terms of the required data types, the multi-party data participating in collaborative computing mainly includes heterogeneous data and homogeneous data. Some methods can be used for both heterogeneous data and homogeneous data, such as [20], and some methods can only be applied to one of heterogeneous data [9] or homogeneous data [5].

How to solve the problem of multi-party privacy protection in collaborative computing is the focus of these methods. At present, two methods are popular: homomorphic encryption and differential privacy. Homogeneous encryption algorithm is used to achieve secure multi-party calculations, ensuring that data can still be calculated in the encrypted state, and correct calculation results can be obtained after decryption. Gascon et al. [9] train a linear regression classifier with vertically partitioned datasets by a hybrid protocol. In this protocol, garbled circuits are used in the two-party computation. A crypto service provider is needed in two-party cases, and a crypto service provider and an evaluator is needed at the same time in multi-party case. Nikolaenko et al. [21] design a privacy-preserving ridge regression algorithm which can be divided into two phases and each phase uses homomorphic encryption and Yao garbled circuits separately. In the algorithm, an evaluator and a crypto service provider are essential to realize the algorithm. Mohassel et al. [20] present a protocol for privacy preserving machine learning and this protocol is able to support several machine learning algorithms such as linear regression, logistic regression and neural network. In this scheme, two servers collect data from data providers and train a model in a secure way by two-party computation. At the same time, the two servers cannot collude. Mohassel et al. [19] construct a framework where three servers are necessary to train linear regression, logistic regression and neural network models based on three-party computation.

However, each of these methods has a disadvantage: it requires the introduction of one or more third parties to participate in the calculation process. On the one hand, the introduction of third parties will cause additional communication overhead and reduce computing efficiency. On the other hand, the introduction of third parties has potential privacy disclosure issues. For example, Francisco-Javier et al. [11] use a two-server model based on partial homomorphic encryption to solve the privacy protection problem when multiple data providers train SVM models. In addition, in the real application process, setting up such a trusted third round is not feasible. In terms of computing efficiency, a large number of encryption, decryption, and homomorphic operations are designed in the calculation process after the homomorphic encryption algorithm is introduced. Therefore, the overall training time will be greatly increased compared to conventional training conditions.

Through the scheme of differential privacy algorithm, sensitive data is protected by adding noise. Abadi et al. [1] apply differential privacy to protect sensitive information in datasets during deep learning. Although differential privacy is an efficient method of privacy protection, the introduction of perturbations has a negative impact on the accuracy of the final trained classifier. For some machine learning algorithms, the negative impacts on the final training results cannot be ignored. In contrast, homomorphic encryption is a more accurate privacy protection scheme.

In the above scheme, the data needs to be shared more or less to complete the calculation. With the development and maturity of the federal learning method, these multi-party collaborative training algorithms can greatly protect the privacy of the original data, and it has been cited in more and more studies [15, 39]. However, the simple federal learning method is still not perfect in terms of privacy protection. In recent years, there has been many researches to apply differential privacy and homomorphic encryption algorithms to the process of federal learning, so as to meet security requirements.

Liu et al. [17] proposed a secure federated transfer learning algorithm. This method achieves important data privacy security in the process of multi-party collaboration for deep learning through the addition of homomorphic encryption algorithms. The differential privacy is introduced into the process of federated learning, which ensures that the parameters of the client during training are not leaked, and the balance of security and model effect is achieved [10, 32].

1.3.2 Multi-party Collaborative Learning with Blockchain

Many fields across the world have applied blockchain to data sharing solutions, such as the medical industry, the energy industry [22], and so on. Compared with the traditional data sharing schemes in their respective fields, the application of blockchain technology has improved the trust and security issues in the data sharing process to a certain extent. Based on a secure data sharing method using blockchain, decentralized multi-party collaborative learning methods are studied.

The decentralized property of the blockchain fits the distributed learning characteristics of federated learning, so many studies have applied the blockchain to the federated learning process to ensure the transparency, security, and auditability of the model process.

Mendis et al. [18] avoid introducing any central server through the blockchain, and at the same time further strengthens the security in the federal learning process through homomorphic encryption algorithms. In the end, the solution is verified to prove its usability and efficiency performance in different scenarios.

Zhu et al. [43] consider the potential of byzantine equipment in the traditional federated learning process, which causes unpredictable results to the training results, a deep learning algorithm for blockchain is proposed. This method adds a layer of security to the classic method to solve the problem of Byzantine equipment. The consensus mechanism running in the blockchain has a natural advantage in solving this problem. In the end, the scheme is experimentally verified to prove the safety and effectiveness of the method.

In addition to the privacy protection in the process of deep learning with multi-party collaboration, the issue of incentives for participants is also considered [35]. Without reasonable incentives, data providers will not be willing to provide data for training. This article proposes a secure and motivating deep learning training program based on blockchain. With the help of blockchain, data confidentiality, computational auditability, and effective incentives for participants are implemented.

Kim et al. [14] pointed out that in the classic centralized federal learning process, the time communication between each device and the central institution is very large, and the central institution also increases the risk of data leakage. This article uses blockchain technology to replace the central server and is responsible for the entire data exchange during training. Experiments show that this decentralized federated learning mechanism has improved communication overhead.

Related researches on traditional machine learning algorithms are also underway. Shen et al. [27] apply partial homomorphic encryption algorithm to design secure computing components, and based on these components, the SVM training process is implemented. At the same time, the solution introduces the blockchain as a data sharing platform, avoiding the introduction of trusted third parties, while ensuring the transparency of the training process. This method uses homogeneous datasets. Experimental results show that the method has advantages in accuracy and efficiency. Shen et al. [28] uses a partial homomorphic encryption algorithm, a secure SVM training method is implemented based on heterogeneous datasets. In this solution, the blockchain plays a huge role, as a data sharing platform in the model training process, ensuring that the data is open and immutable.

1.4 Data Sharing Requirements and Challenges

Data is an important driving force in the stage of big data era for social progress and industrial development. With the promotion and application of big data technology and artificial intelligence technology, traditional industries are continuously empowered through statistical analysis, machine learning model training, and deep neural networks, which have improved efficiency and reduced costs, and solved a series of industry pain points. For example, it has considerable advantages in application fields such as mobile payment, online credit reporting, and e-commerce. The progress and development of the industry has brought great convenience and improvement to people's lives in certain scenarios.

1.4.1 Integrity of Data Sharing

For a single entity, there are huge challenges when using data, for the terrible integrity of the data. All industries and industries have accumulated a large amount of data, and these data can play an important role when they are shared. From the perspective of data integrity, the integrity of data depends on the number of data cases, the number of data attributes of a data case, and the quality of the data.

Insufficiency of Data Instances When an entity has small size of datasets in data instances, it needs other entities's data to enlarge the datasets size. The attributes held by each entity are the same. We call these data homogeneous data. For example, in a medical scenario, multiple medical institutions share patient medical data, and the data samples of all patients contain the same attributes. Similarly, the label values of the sample data are determined according to an unified standard. Heterogeneous data refers to the intersection of the data samples provided by each data source, but the data samples of each data source contain different data attributes. Therefore, through the homogeneous data sharing, the integrity of the data can be improved from the perspective of expanding the number of data cases.

Insufficiency of Data Attributes The attributes of data owned by a single institution are limited. In some scenarios, the data of other institutions are needed to expand the attribute dimension of its own data. These data are the same instance, but there is a complementary relationship on the attributes. We call these data heterogeneous data. Similarly in the medical field, multiple medical institutions have data for a certain patient, and the data attributes of these data are different. Therefore, the data integrity can be improved from the perspective of expanding the number of instance attributes of the data through heterogeneous data sharing.

Inferior of Data Quality During the process of data collection, transmission, processing, and labeling, the data quality varies due to technical level, data volume, and data source. Therefore, through data sharing, data from other institutions are obtained, so as to comprehensively compare and screen out higher quality data, and ultimately improve the quality of data analysis and model training results. Improved data quality also helps improve data integrity.

1.4.2 Security of Data Sharing

It faces serious privacy leakage problems during the sharing process. Privacy leakage may happen during the process of data transmission, data storage and so on. The direct consequences of data privacy breaches include: reduced data value and leakage of sensitive information. The above two problems may lead to more serious consequences including: stricter data sharing restrictions, data providers are afraid, unwilling or unable to share data.

Loss of Data Value The current data sharing method is mainly copy data sharing. The data is directly copied to the data requester through a certain method, and the data requester directly obtains the original data of the data. Although some privacy protection measures such as encryption can be used to ensure that the privacy of the data is not leaked during transmission, after the valuable data contained in the original data is diffused to other participants, the data cannot maintain its original value. This problem is particularly acute when sharing high-value data.

Leakage of Sensitive Data Another problem with replicated data sharing is the leakage of sensitive data. Even though the shared data may not contain private data, after the data is acquired by malicious parties, some sensitive information may be statistically obtained through certain data analysis methods. The leakage of this information can have serious consequences to some extent.

Due to the above reasons, many regions and countries have introduced more and more strict measures to restrict data sharing. From a positive perspective, these measures have ensured the user's privacy to a certain extent, but at the same time they have hindered the data sharing in many industries.

1.4.3 Usability of Data Sharing

Many privacy protection schemes are currently used to protect privacy leaks during data sharing. The introduction of cryptography schemes ensure that the privacy of data is not leaked. However, after the introduction of the privacy protection scheme, data sharing efficiency decreased, data calculation results were inaccurate, and the need to introduce trusted third parties may reduced the data availability.

Low Efficiency When the cryptographic algorithm is introduced, the data sharing mode is changed from replication sharing mode to computation sharing mode. However, a large number of encryption and decryption operations are involved in the calculation process. These operations will reduce the calculation efficiency. When the amount of data increases, the excessive calculation time cannot meet the needs of practical applications.

Low Accuracy The introduction of differential privacy protection calculations will lead to the introduction of noise, which will cause the final calculation result to be biased. During the machine learning or deep learning process, the error of calculation will affect the model parameters, and will ultimately affect the classification effect of the model, resulting in reduced accuracy of classification, unable to meet the needs of the actual scene.

Indispensable Third Party Most secure collaborative training schemes based on homomorphic encryption have to rely on a trusted third party to ensure the smooth progress of calculation processes. However, it is difficult to find such a trusted third party in real scenarios. As a result, the applicability of these schemes is very poor, which cannot meet the requirements of cost performance and efficiency in applications.

1.5 Organization

In the following pages, the solutions to the problems encountered in the data sharing will be introduced in detail. The second chapter describes the overall structure of the blockchain, and briefly describes the application of the blockchain in data sharing. The third chapter introduces the hierarchical structure of the blockchain-based data sharing system, and gives an introduction to the functions of each layer. Chapter 4 details the homogeneous data sharing scheme based on the blockchain, and Chap. 5 details the heterogeneous data sharing scheme based on the blockchain. Chapter 6 provides a detailed introduction to secure data retrieval based on the blockchain. Chapter 7 introduces the incentive scheme in the process of data sharing based on blockchain.

References

1. M. Abadi, A. Chu, I. Goodfellow, H.B. McMahan, I. Mironov, K. Talwar, L. Zhang, Deep learning with differential privacy, in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16 (ACM, New York, NY, USA, 2016), pp. 308–318
2. Y. Aono, T. Hayashi, L. Trieu Phong, L. Wang, Scalable and secure logistic regression via homomorphic encryption, in *Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy*, CODASPY '16 (ACM, New York, NY, USA, 2016), pp. 142–144
3. Y. Aono, T. Hayashi, L. Trieu Phong, L. Wang, Privacy-preserving logistic regression with distributed data sources via homomorphic encryption. IEICE Trans. Inf. Syst. **99**(8), 2079–2089 (2016)
4. M.d. Cock, R. Dowsley, A.C. Nascimento, S.C. Newman, Fast, privacy preserving linear regression over distributed datasets based on pre-distributed data, in *Proceedings of the 8th ACM Workshop on Artificial Intelligence and Security*, AISec '15 (ACM, New York, NY, USA, 2015), pp. 3–14
5. M.d. Cock, R. Dowsley, A.C. Nascimento, S.C. Newman, Fast, privacy preserving linear regression over distributed datasets based on pre-distributed data, in *Proceedings of the 8th ACM Workshop on Artificial Intelligence and Security* (ACM, 2015), pp. 3–14
6. R. Creemers, Cybersecurity law of the people's republic of china (third reading draft). *China Copyright and Media*, 2016
7. Data transaction research report. https://blog.naaln.com/2019/12/report-data-transaction/. Accessed February 23, 2020
8. F. Gao, L. Zhu, M. Shen, K. Sharif, Z. Wan, K. Ren, A blockchain-based privacy-preserving payment mechanism for vehicle-to-grid networks. IEEE Netw. **32**(6), 184–192 (2018)
9. A. Gascón, P. Schoppmann, B. Balle, M. Raykova, J. Doerner, S. Zahur, D. Evans, Secure linear regression on vertically partitioned datasets. IACR Cryptology ePrint Archive **2016**, 892 (2016)
10. R.C. Geyer, T. Klein, M. Nabi, Differentially private federated learning: A client level perspective. arXiv: Cryptography and Security, 2017
11. F.-J. González-Serrano, Á. Navia-Vázquez, A. Amor-Martín, Training support vector machines with privacy-protected data. Pattern Recognit. **72**, 93–107 (2017)

12. T. Graepel, K. Lauter, M. Naehrig, Ml confidential: Machine learning on encrypted data, in *Information Security and Cryptology – ICISC 2012*, ed. by T. Kwon, M.-K. Lee, D. Kwon (Springer, Berlin, Heidelberg, 2013), pp. 1–21
13. H. Kim, J. Park, M. Bennis, S. Kim, On-device federated learning via blockchain and its latency analysis. arXiv: Information Theory, 2018
14. H. Kim, J. Park, M. Bennis, S. Kim, Blockchained on-device federated learning. IEEE Commun. Lett., 1–1 (2019)
15. T. Li, A.K. Sahu, A. Talwalkar, V. Smith, Federated learning: challenges, methods, and future directions. arXiv: Learning, 2019
16. X. Liu, R. Lu, J. Ma, L. Chen, B. Qin, Privacy-preserving patient-centric clinical decision support system on naive bayesian classification. IEEE J. Biomed. Health Inform. **20**(2), 655–668 (2016)
17. Y. Liu, T. Chen, Q. Yang, Secure federated transfer learning. arXiv: Learning, 2018
18. G.J. Mendis, Y. Wu, J. Wei, M. Sabounchi, R. Roche, Blockchain as a service: A decentralized and secure computing paradigm. arXiv: Cryptography and Security, 2018
19. P. Mohassel, P. Rindal, Aby 3: a mixed protocol framework for machine learning, in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security* (ACM, 2018), pp. 35–52
20. P. Mohassel, Y. Zhang, Secureml: A system for scalable privacy-preserving machine learning, in *2017 IEEE Symposium on Security and Privacy (SP)* (IEEE, 2017), pp. 19–38
21. V. Nikolaenko, U. Weinsberg, S. Ioannidis, M. Joye, D. Boneh, N. Taft, Privacy-preserving ridge regression on hundreds of millions of records, in *2013 IEEE Symposium on Security and Privacy* (IEEE, 2013), pp. 334–348
22. A. Pieroni, N. Scarpato, L.D. Nunzio, F. Fallucchi, M. Raso, Smarter city: Smart energy grid based on blockchain technology. Int. J. Adv. Sci. Eng. Inf. Technol. **8**(1), 298–306 (2018)
23. M. Shen, Y. Deng, L. Zhu, X. Du, N. Guizani, Privacy-preserving image retrieval for medical IoT systems: A blockchain-based approach. IEEE Netw. **33**(5), 27–33 (2019)
24. M. Shen, J. Duan, L. Zhu, J. Zhang, X. Du, M. Guizani, Blockchain-based incentives for secure and collaborative data sharing in multiple clouds. IEEE J. Sel. Areas Commun. **38**(6), 1229–1241 (2020)
25. M. Shen, H. Liu, L. Zhu, K. Xu, H. Yu, X. Du, M. Guizani, Blockchain-assisted secure device authentication for cross-domain industrial IoT. IEEE J. Sel. Areas Commun. **38**(5), 942–954 (2020)
26. M. Shen, B. Ma, L. Zhu, R. Mijumbi, X. Du, J. Hu, Cloud-based approximate constrained shortest distance queries over encrypted graphs with privacy protection. IEEE Trans. Inf. Forensics Secur. **13**(4), 940–953 (2017)
27. M. Shen, X. Tang, L. Zhu, X. Du, M. Guizani, Privacy-preserving support vector machine training over blockchain-based encrypted iot data in smart cities. IEEE Internet Things J. **6**(5), 7702–7712 (2019)
28. M. Shen, J. Zhang, L. Zhu, K. Xu, X. Tang, Secure svm training over vertically-partitioned datasets using consortium blockchain for vehicular social networks. IEEE Trans. Veh. Technol. **69**(6), 5773–5783 (2020)
29. R. Shokri, V. Shmatikov, Privacy-preserving deep learning, in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security* (ACM, 2015), pp. 1310–1321
30. C. Sun, A. Shrivastava, S. Singh, A. Gupta, Revisiting unreasonable effectiveness of data in deep learning era, in *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 843–852, Oct 2017
31. C. Tikkinen-Piri, A. Rohunen, J. Markkula, Eu general data protection regulation: changes and implications for personal data collecting companies. Comput. Law Secur. Rev. **34**(1), 134–153 (2018)
32. S. Truex, N. Baracaldo, A. Anwar, T. Steinke, H. Ludwig, R. Zhang, Y. Zhou, A hybrid approach to privacy-preserving federated learning. arXiv: Learning, 2018

33. J. Vaidya, B. Shafiq, W. Fan, D. Mehmood, D. Lorenzi, A random decision tree framework for privacy-preserving data mining. IEEE Trans. Dependable Secure Comput. **11**(5), 399–411 (2014)
34. J. Vaidya, H. Yu, X. Jiang, Privacy-preserving svm classification. Knowl. Inf. Syst. **14**(2), 161–178 (2008)
35. J. Weng, J. Weng, M. Li, Y. Zhang, W. Luo, Deepchain: Auditable and privacy-preserving deep learning with blockchain-based incentive. IACR Cryptology ePrint Archive **2018**, 679 (2018)
36. Q. Wu, Big data sharing and opening face three challenges. https://www.huxiu.com/article/198724.html. Accessed February 23, 2020
37. K. Xu, L. Lv, T. Li, M. Shen, H. Wang, K. Yang, Minimizing tardiness for data-intensive applications in heterogeneous systems: A matching theory perspective. IEEE Trans. Parallel Distrib. Syst. **31**(1), 144–158 (2019)
38. L. Xu, K. Xu, Y. Jiang, F. Ren, H. Wang, Throughput optimization of tcp incast congestion control in large-scale datacenter networks. Comput. Netw. **124**, 46–60 (2017)
39. Q. Yang, Y. Liu, T. Chen, Y. Tong, Federated machine learning: concept and applications. arXiv: Artificial Intelligence, 2019
40. H. Yu, J. Vaidya, X. Jiang, Privacy-preserving svm classification on vertically partitioned data, in *Pacific-Asia Conference on Knowledge Discovery and Data Mining* (Springer, 2006), pp. 647–656
41. J. Zhang, Z. Zhang, X. Xiao, Y. Yang, M. Winslett, Functional mechanism: regression analysis under differential privacy. Proc. VLDB Endowment **5**(11), 1364–1375 (2012)
42. B.-K. Zheng, L.-H. Zhu, M. Shen, F. Gao, C. Zhang, Y.-D. Li, J. Yang, Scalable and privacy-preserving data sharing based on blockchain. J. Comput. Sci. Technol. **33**(3), 557–567 (2018)
43. X. Zhu, H. Li, Y. Yu, Blockchain-based privacy preserving deep learning, in *International Conference on Information Security and Cryptology* (Springer, Cham, 2018), pp. 370–383

Chapter 2
Blockchain and Data Sharing

Data sharing is very common in the era of big data and artificial intelligence as the convergence of data is the most prerequisite section of big data and machine intelligence happens based on high volume data feed. Blockchain is a new emerging technology, which holds vast possibilities and benefits to improve traceability and accountability of its stored data. It has significant advantages in unlocking the full potential of data sharing in multi-party scenarios. In this chapter, we introduce the basics of blockchain, which are organized into five layers: data layer, network layer, consensus layer, incentive layer and application layer. The details of each layer are elaborated accordingly. Afterwards, we emphasize the strengths of blockchain, including integrity, immutability, decentralization and verifiability. Finally, we make a list of the threats and challenges of blockchain to be overcame.

2.1 Blockchain Overview

A blockchain [34] is a distributed ledger of all transactions that have ever been executed. A block is the basic part of a blockchain which records some or all of the transactions occurs in a time period, and once completed, goes into the blockchain as permanent database. A new generated block contains the most recent executed transactions. Blocks are linked to each other (chained block) in proper linear, chronological order with every block containing a hash of the previous block. To use conventional banking as an analogy, the blockchain is like a full history of banking transactions. Transactions are entered chronologically in a blockchain just the way bank transactions are. Meanwhile, blocks, are like individual bank statements. The full copy of the blockchain has records of every transaction ever executed.

From the perspective of how the ledger is maintained, blockchain can be divided into three categories, which are public blockchain, consortium blockchain and private blockchain, respectively. In public blockchain systems, the ledger is open to

M. Shen et al., *Blockchain: Empowering Secure Data Sharing*,
https://doi.org/10.1007/978-981-15-5939-6_2

the public. Anyone can be the maintainer of the ledger if they want. Bitcoin [17, 32] and Ethereum [27, 31] are two typical public blockchain systems. A selected set of nodes are responsible for the maintenance of the ledger in consortium blockchain systems. These nodes usually represent their own organizations that constitute a consortium, which cooperate for the same goal. Hyperledger Fabric [4, 10] is the most widely used consortium blockchain [8, 14]. As for private blockchain systems, only nodes from a single organization have the permission to maintain the ledger. Thus, the ledger in a private blockchain is fully controlled by a standalone organization.

Blockchain has some general components [1, 3, 7]. Generally, the architecture of blockchain can be organized as five tiers, which are data layer, network layer, consensus layer, incentive layer and application layer as shown in Fig. 2.1.

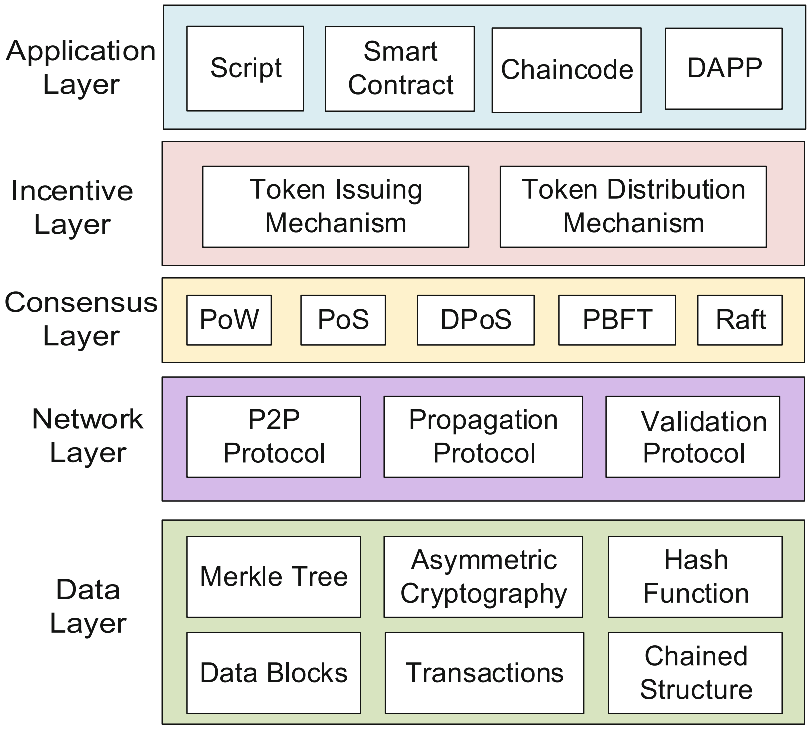


Fig. 2.1 An overview of blockchain architecture

2.1.1 *Data Layer*

Data layer contains the data structure of blockchain and the related techniques exploited to build such construction.

- Chained Structure

Block [3] is the fundamental components of blockchain. Each a block records transactions generated during a period. To provide immutability of the ledger data, each block record the hash of the previous block in its block head, and then compute the hash of the current block. Thus, blocks are linked linearly in chronological order as shown in Fig. 2.2.

- Data Blocks

An example of Bitcoin block structure [3] is shown in Fig. 2.3. A block is composed of the block header and the block body. Specifically, block header includes fields of block version, merkle tree root hash, timestamp, nBits, nonce and previous block hash. Block version indicates the validation rules nodes to follow. Merkle root hash is computed on all the transactions in the block. Timestamp records current time since January 1, 1970. NBits indicates the validation of a block hash. Nonce is adjusted by the miner to validate a block. Previous block hash is a 256-bit hash value that points to the previous block.

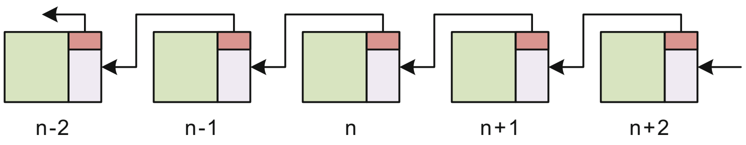


Fig. 2.2 Chained structure of blockchain. Each block points to the previous one using the hash value

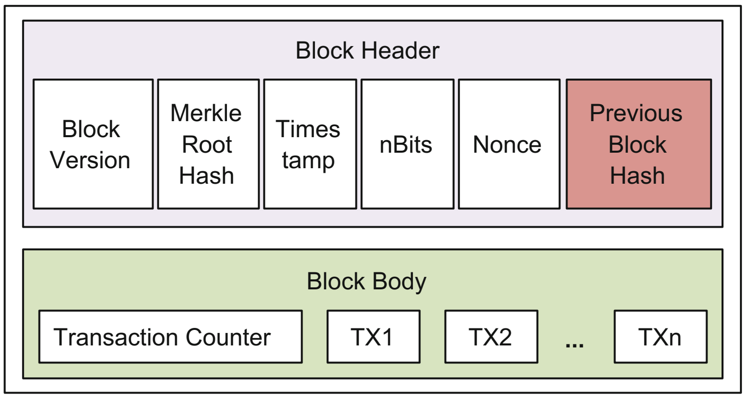


Fig. 2.3 Block structure of Bitcoin

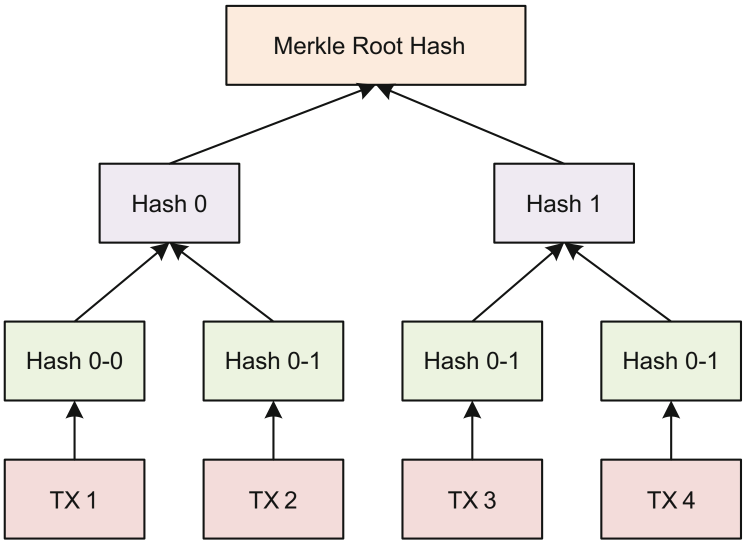


Fig. 2.4 An example of merkle tree construction based on four transactions

- Merkle Tree

The merkle tree [3] is constructed by pairing each transaction with one other transaction and hashing on them. The computed hashes are also paired with one other hash and hased again. Such process repeats until only one hash remains, i.e., the merkle root. Figure 2.4 shows an example of how a merkle tree is constructed.

- Hash Function

The hash function is exploited to iteratively compute the hash of merkle root. Besides, each block is hashed whose value is stored in the following block. In this way, it is almost impossible to modify the ledger once the data is written. Hash-256 algorithm is used in Bitcoin [3].

- Transactions

There exists two kinds of transactions [3], i.e, coinbase transactions and common transactions. Coinbase transactions are created by miners. Common transactions can be generated by anyone who sends transaction proposal. Common transactions contain the data generated from business applications.

- Asymmetric Cryptography

In blockchain systems, signature mechanism is the mainly used asymmetric cryptography [3]. Bitcoin address is derived from the public key of a signature key pair. Every transaction is signed by the sender using its signature private key to prove its origin.

2.1.2 Network Layer

Network layer includes the protocols utilized in the blockchain network, i.e., the Peer-to-Peer (P2P) protocol exploited to construct the blockchain network, the propagation protocol exploited to broadcast transaction proposals and new executed transactions, and the validation protocol exploited to validate the transactions before being committed to the ledger.

- P2P Protocol

All the blockchain network nodes are organized using certain P2P protocol. Nodes in a P2P network [3] has equal power and perform the same tasks. Usually, they have no central administrator as each node holds a copy of files—acting as an individual peer. A P2P protocol prescribes how nodes are organized. P2P protocol describes the mechanism that a node discoveries and connects with the other peers when it starts at the first time. The protocol describes the strategy that a node updates its connections to new peers when the connected peers are off-line. Also, the protocol describes the mechanism that a node synchronizes its local data from other peers.

- Propagation Protocol

The propagation protocol is usually acted as a sub-part of P2P protocol, which performs the data synchronization task. In blockchain systems, transaction proposals, executed transactions or blocks are needed to be broadcast to others when a node initiates or receives such data. In bitcoin, when a miner discoveries a new block, it broadcasts the new block to its peers using certain prescribed methods.

- Validation Protocol

Similarly, the validation protocol is more likely to be the specifications that describe the rules that transactions are structured and how they are organized into a block. For instance, in bitcoin, a miner would validate that the bitcoin of the input addresses are unspent. Peers would re-validate the transaction and the hash value of the new added block before they append it to their local blockchain.

2.1.3 Consensus Layer

Blockchain is a distributed ledger. Different from the traditional centralized ledger systems, there is no central authority present to validate and verify the executed transactions. Nonetheless, blockchain have to provide consistent ledger data to its users, and this is where consensus comes into play. The consensus is core part of any blockchain network.

Consensus [30] is a procedure through which all the peers in a blockchain network reach an common agreement on the present state of the distributed without centralized authority. In this way, blockchain network achieve reliability and construct trust among unknown peers in a distributed environment. Essentially, the consensus guarantees that every new added block to each peers are consistent, upon which is agreed by all the peers in the blockchain network.

Here, we discuss several most typical consensus algorithms, which are Proof of Work (POW), Proof of Stake (PoS), Delegated Proof of Stake (DPoS), Practical Byzantine Fault Tolerance (PBFT) and Raft. We discuss how they work in a precise manner.

- PoW

PoW [29] is the consensus algorithm used in the Bitcoin [17]. Each node of the network is calculating the hash value in the block header. There is a random number (Nonce) in the block header so that miners can change to get different hash values. The consensus requires that the computed hash value must satisfied with the threshold set in the nBits field. Once a miner gets such hash value, it has the right to generate a new block and broadcast the block to others. All the other nodes would verify the correctness of the hash value. The new block will be attached to the main block chain by other nodes only it is verified.

- PoS

PoS [28] is an alternative to the PoW for energy-saving as PoS can greatly save computing resources. Miners must prove their ownership of the amount of tokens. A user is more likely to be selected as a miner as long as he owns more tokens. It is believed that users with more tokens have less motivation to attack the network. However, it can ultimately leads to violate the original intention of decentralization, since the user who owns the most tokens can dominate the network. Compared to PoW, PoS significantly saves energy and is more efficient. Thus, many blockchains that adopt PoW at the beginning gradually transit to PoS. For example, Ethereum [31] is trying to transform from Ethrash (A variant of PoW) to Casper (A variant of PoS).

- DPoS

DPoS [12] is a variant of PoS that can provide high scalability at the cost of sacrificing the full decentralization characteristic. In DPoS, only a fixed number of elected nodes (called block producers or witnesses) are selected to produce blocks in a round-robin order. These block producers are elected by all the network participants, who can get a number of votes proportional to the number of the tokens they own, i.e., their stake. Alternatively, users can choose to delegate their votes to another voter, who will further vote target block producers on their behalf. The number of block producers in a DPoS network depends on the concrete rules of that chain. A block producer is possible to be voted out power if it is found to be malicious. In such condition, voters will not vote for them in the next round. By significantly limiting the number of block producer, DPoS gains high scalability and is able to handle transaction throughput that is multiple orders of magnitude greater that today's PoW. However, it is noted that DPoS sacrifices decentralization for high throughput. DPoS is now exploited in the Bitshares [19].

- PBFT

Pratical Byzantine Fault Tolerance (PBFT) is a replication algorithm introduced in 1999 by Barbara Liskov and Miguel Castro [5], which was designed to work efficiently in asynchronous systems tolerating byzantine faults. A PBFT system can function well on the condition that the maximum number of malicious nodes must less than or equal to 1/3 of all the node in the system. PBFT operates in rounds, which are broken into 3 phases: request, pre-prepared, prepared and commit. In PBFT systems, only one node being the primary node and others referred to as secondary nodes. In each round, a primary node would be selected according to some rules. Once the primary node receives a request from a client, the primary node broadcasts the request to the secondary. The nodes (both primary and secondaries) perform the service requested and then send back a reply to the client. The request is served successfully only the client receives $m+1$ replies with the same result, where m is the allowed maximum number of faulty nodes. Hyperledger Fabric supports pluggable PBFT consensus module.

- Raft

Raft [18] is a consensus algorithm that is designed to be easy to understand. It's equivalent to (multi-)Paxos in fault-tolerance and performance. The difference is that it's decomposed into relatively independent subproblems, and it cleanly addresses all major pieces needed for practical systems. To enhance the understandability, Raft separates the key components of consensus, for instance, the leader election, log replication, etc. Faft uses the "leader and follower" model, in which a leader is dynamically selected among the ordering service nodes. The messages of the followers are replicated from the unique leader. Raft systems can sustain the loss of nodes, including the leader node, as long as there is a majority of ordering nodes remaining. Thus, Raft is saide to be "crash fault tolerant". Hyperledger Fabric supports Raft as a pluggable consensus module.

2.1.4 Incentive Layer

Token is most widely used to implement the incentive mechanisms in blockchain systems, e.g., BTC in Bitcoin system, ETH in Ethereum system, etc. Tokens usually have monetary value, which can be exchanged with legal currency. It is not necessary to all kinds of blockchain. Actually, public blockchain would exploit token to stimulate the nodes to mine the new block because it's computation-consuming to package transactions into the new created block. In consortium blockchain, the peers are usually self-motivated as they work collaboratively to achieve the same goal. Nonetheless, it is also feasible to circulate tokens in a consortium blockchain, which is usually for better management of digital assets.

The issuing and distribution of tokens are the major parts of incentive mechanisms [15]. Token issuing mechanism prescribes how the token is generated, which would increase of total quantity of token. Token distribution mechanism prescribes how the existing tokens are circulated among different users.

In Bitcoin system, the maximum and total amount of bitcoins that can ever exist is 21 million. Right now, every time a new block generates, 12.5 bitcoins are issued to the miner for circulation. Such mining awards is reduced every time 210,000 blocks are generated. Bitcoins can be circulated by simple money transfer from one to another bitcoin address, or by transaction fees payed for miner from transaction sender.

Token in Ethereum system is more complex than Bitcoin. 72,000,000 ETHs is pre-mined into circulation. After that, new ETHs are mined into circulation as block rewards, uncle rewards and uncle referencing rewards. Block rewards are given to those who mines a new block on th main chain. Uncle rewards are given to those who mines a new uncle block, which are mined out lately after the blocks in the main chain. Bitcoin completely discards these uncle blocks. Uncle referencing rewards are give to miner who references those uncle blocks. Such kind of token issuing mechanism reduces the waste of computing. BTCs can be simply transfer from one to another ethereum address. If a user want his smart contract to be launched by the Ethereum nodes, ETCs need to be attached for supporting the execution of the smart contract.

Hyperledger Fabric is the most widely used consortium blockchain. There is no intrinsic token in the published version. Nonetheless, the official documents declaim that Fabric is support for adding token into its system. Since the token is not necessary in consortium blockchain, we do not discuss it here.

2.1.5 Application Layer

Developers can develop their business application using the underlying blockchain as the append-only database. Such business logic is implemented using the programming mechanism provide by blockchain systems, e.g., script, smart contract, chaincode, etc.

- Script

Script [3] is a mini programming language provided in bitcoin, which is used as a locking mechanism for bitcoins. A locking script is place on every output. Correspondingly, an unlocking script must be provided in a transaction to unlock an output when it is used as an input. A script consists of two parts: data and OPCODES. OPCODES are simple functions that operate on the data.

- Smart Contract

Programs that run on the Ethereum Virtual Machine (EVM) are commonly referred to as smart contracts [7]. Solidity is the most popular language for writing smart contracts on Ethereum. Smart contracts are usually consuming Ether when they are running. They can control digital assets for creating new kinds of financial applications. Once smart contracts are deployed to Ethereum, they will always run as programmed, which means they cannot be further modified. A developer should deploy a new smart contract when he wants to upgrade or patch the original one.

- Chaincode

Chaincode [6] functions similarly to smart contract. It is a terminology in Hyperledger Fabric. Chaincode is a program written in Go, node.js, or Java that implements a prescribed interface. In Hyperledger Fabric, chaincode runs in a dependent docker container that is isolated from the other processes. The blockchain ledger is initialized and managed by chaincodes. Applications need to invoke chaincodes to change the ledger state.

- DAPP

DAPPs are kind of applications that encapsulate the script, smart contract, or chaincode, which are invoked by SDK provided by the official or a third party. They can usually provide user-friendly interfaces to ordinary users. In ethereum, gaming and exchange are the most two popular DAPPs. There are many case studies in hyperledger fabric, which extends blockchain into more and more sectors, e.g., manufacturing, healthcare, small business, supply chain, digital identity, retailers and so on.

2.2 Strengths of Blockchain

- Integrity of Transactions

As a distributed ledger system, blockchain records and validate each and every transaction made, which makes it secure and reliable. In the scenario of data sharing, data consumers depend heavily on the reliability and trusthworthiness of data, which is known as data integrity. The data integrity involves the maintenance of, and the assurance of the accuracy and consistency of data over its entire life-cycle. Data sharing [33] operation and the actual sharing data can be some extent encoded as part transactions and further being recorded by the blockchain. In such condition, the data consumer can put high trust on the sharing data as the blockchain provides good data integrity [35].

- Immutability of Ledger

By design, blockchain is inherently resistant to the modification of data. The immutability of blockchain ledger means that once the transactions are recorded into blockchain, it cannot be edited or deleted. In addition, blockchains are also a timekeeping mechanism for the data, thus the history of data is easily reportable. As long as the data is shared upon a blockchain system, the recored sharing data is believed to be the original data provided by the data owner. Because of the characteristic of immutability, it's easy to compute the profits of the data providers referring to the history of data sharing [22] transactions. Besides, data providers are encouraged to share valuable data for higher repay.

- Decentralization of Ledger Maintenance

In blockchain systems, there is no central authority or third-party to intentionally record the ledger. Instead, the blockchain ledger is maintained by a number of peers. In such way, it is hard for an entity to modify or forge the ledger data, which is one of the most drawbacks of centralized systems. For better maintenance, data providers and consumers can negotiate the rules that all the network participants must conform to. Any transactions that violate the prescribed rules are refused.

- Verifiability of Identity

In consortium systems, the blockchain ledger is maintained by a number of pre-selected nodes who consist of a consortium. Only legitimate users can participates in the network. In Hyperledger Fabric, in particular, there is a member mechanism designed to control the network participants and authorize proper privileges to them. Auditing of network participants can be useful in data sharing scenario [13]. A consortium can choose to authorize the related user to participate their network. Thus, it is believed that the new entrant is more likely to be an honest entity.

2.3 Security and Privacy Threats to Blockchain

- Attacks on Blockchain

There exists lots of attacks [11] on blockchain systems, e.g., 51% attack, eclipse attack, selfish attack and so on. A miner with more computing resources will calculate to find the nonce value faster than the others, which may leads to 51% attack. In eclipse attack, a dishonest node try to control the victims from the rest of the peers in the blockchain network. Selfish mining attack happens when a dishonest miner has enough computing power to discover more blocks than the public chain.

- Privacy Leakage of Transaction Data

All transactions recorded on blockchain are stored in clear text. By using a blockchain explorer, the data contained in each field of a transaction can be easily known by others. The blockchain ledger is open to all the peers, so each peer can know what data is stored in the ledger. In this way, sensitive data may be exposed to public [16]. It may hinder users from sharing data as some users only want their shared data can be only be visible to restricted users instead of everyone else. Thus, data privacy-preserving [9, 21, 26] should be carefully considered.

- Limits on Data Representation

All the data to be stored on blockchain is formated into a prescribed fields. In this way, it is convenient for nodes to process transactions. However, data in different sectors [20, 25] are totally different. A flexible data formate storage mechanism which is generic to sectors is needed when data sharing involves participants spanning cross different sectors [23, 24].

- Limits on Data Storage

Individual blockchain ledger are typically on the order of kilobytes [2] and only limited data are held. However, the blockchain's log structure implies that all the transaction operations are recorded in the blockchain. That's also what users expect when they are sharing data with others. Many data operations happens when the data are circulated through data consumers. Thus, there is a great demand of storage not only for sharing data but also for data manipulation.

- Latency on Transaction Settlement

Unlike the centralization systems controlled by a single node, blockchain systems are maintained by lots of nodes that are geographically dispersive. Transactions need to be validated by lots of nodes before they are committed. Moreover, blockchain systems using probabilistic consensus algorithm, enough blocks must be added after the block encapsulating the committed transactions to make sure the non-reversibility. This leads to low efficiency of transaction settlement. However, in some real practice of data sharing, higher throughput of transactions are needed.

2.4 Summary

In this chapter, we introduce the basics of blockchain, which are organized into five layers: data layer, network layer, consensus layer, incentive layer and application layer. This five layers almost include everything of blockchain. The details of each layer are elaborated accordingly. We emphasize the strengths of blockchain, which include integrity, immutability, decentralization and verifiability. These characteristics may enable blockchain to unlock the full potential of data sharing in multi-party scenarios. Finally, we make a list of the threats and challenges, which should be carefully considered.

References

1. A blockchain platform for the enterprise, 2020. [Online; accessed 17-April-2020]
2. M. Ali, J. Nelson, R. Shea, M.J. Freedman, Blockstack: A global naming and storage system secured by blockchains, in *2016 {USENIX} Annual Technical Conference ({USENIX} {ATC} 16)*, pp. 181–194, 2016

3. Bitcoin developer reference, 2020. [Online; accessed 17-April-2020]
4. C. Cachin et al., Architecture of the hyperledger blockchain fabric, in *Workshop on Distributed Cryptocurrencies and Consensus Ledgers*, vol. 310, p. 4, 2016
5. M. Castro, B. Liskov, et al., Practical byzantine fault tolerance, in *OSDI*, vol. 99, pp. 173–186, 1999
6. Chaincode tutorials, 2020. [Online; accessed 17-April-2020]
7. Developer resources, 2020. [Online; accessed 17-April-2020]
8. K. Gai, Y. Wu, L. Zhu, M. Qiu, M. Shen, Privacy-preserving energy trading using consortium blockchain in smart grid. IEEE Trans. Ind. Inf. **15**(6), 3548–3558 (2019)
9. F. Gao, L. Zhu, M. Shen, K. Sharif, Z. Wan, K. Ren, A blockchain-based privacy-preserving payment mechanism for vehicle-to-grid networks. IEEE Netw. **32**(6), 184–192 (2018)
10. H.A.W. Group et al., Hyperledger architecture volume 1: Introduction to hyperledger business blockchain design philosophy and consensus, 2017
11. T.T. Huynh, T.D. Nguyen, H. Tan, A survey on security and privacy issues of blockchain technology, in *2019 International Conference on System Science and Engineering (ICSSE)* (IEEE, 2019), pp. 362–367
12. D. Larimer, Delegated proof-of-stake (dpos). *Bitshare whitepaper*, 2014
13. H. Li, L. Zhu, M. Shen, F. Gao, X. Tao, S. Liu, Blockchain-based data preservation system for medical data. J. Med. Syst. **42**(8), 141 (2018)
14. Y. Li, L. Zhu, M. Shen, F. Gao, B. Zheng, X. Du, S. Liu, S. Yin, Cloudshare: towards a cost-efficient and privacy-preserving alliance cloud using permissioned blockchains, in *International Conference on Mobile Networks and Management* (Springer, 2017), pp. 339–352
15. L. Luu, J. Teutsch, R. Kulkarni, P. Saxena, Demystifying incentives in the consensus computer, in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pp. 706–719, 2015
16. G. Malavolta, P. Moreno-Sanchez, A. Kate, M. Maffei, S. Ravi, Concurrency and privacy with payment-channel networks, in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 455–471, 2017
17. S. Nakamoto, Bitcoin: A peer-to-peer electronic cash system. Technical report, Manubot, 2019
18. D. Ongaro, J. Ousterhout, In search of an understandable consensus algorithm (extended version), 2013
19. F. Schuh, D. Larimer, Bitshares 2.0: General overview. Accessed February-2019.[Online]. Available: https://cryptorating.eu/whitepapers/BitShares/bitshares-general.pdf, 2017
20. M. Shen, G. Cheng, L. Zhu, X. Du, J. Hu, Content-based multi-source encrypted image retrieval in clouds with privacy preservation. Futur. Gener. Comput. Syst. (2018)
21. M. Shen, Y. Deng, L. Zhu, X. Du, N. Guizani, Privacy-preserving image retrieval for medical iot systems: A blockchain-based approach. IEEE Netw. **33**(5), 27–33 (2019)
22. M. Shen, J. Duan, L. Zhu, J. Zhang, X. Du, M. Guizani, Blockchain-based incentives for secure and collaborative data sharing in multiple clouds. IEEE J. Sel. Areas Commun. **38**(6), 1229–1241 (2020)
23. M. Shen, H. Liu, L. Zhu, K. Xu, H. Yu, X. Du, M. Guizani, Blockchain-assisted secure device authentication for cross-domain industrial IoT. IEEE J. Sel. Areas Commun. **38**(5), 942–954 (2020)
24. M. Shen, B. Ma, L. Zhu, X. Du, K. Xu, Secure phrase search for intelligent processing of encrypted data in cloud-based iot. IEEE Internet Things J. **6**(2), 1998–2008 (2018)
25. M. Shen, B. Ma, L. Zhu, R. Mijumbi, X. Du, J. Hu, Cloud-based approximate constrained shortest distance queries over encrypted graphs with privacy protection. IEEE Trans. Inf. Forensics Secur. **13**(4), 940–953 (2017)
26. M. Shen, X. Tang, L. Zhu, X. Du, M. Guizani, Privacy-preserving support vector machine training over blockchain-based encrypted iot data in smart cities. IEEE Internet Things J. **6**(5), 7702–7712 (2019)

27. M. Shen, J. Zhang, L. Zhu, K. Xu, X. Tang. Secure SVM training over vertically-partitioned datasets using consortium blockchain for vehicular social networks. IEEE Trans. Veh. Technol. **69**(6), 5773–5783 (2020)
28. Wikipedia contributors, Proof of stake — Wikipedia, the free encyclopedia, 2019. [Online; accessed 9-February-2020]
29. Wikipedia contributors, Proof of work — Wikipedia, the free encyclopedia, 2019. [Online; accessed 9-February-2020]
30. Wikipedia contributors, Consensus (computer science) — Wikipedia, the free encyclopedia, 2020. [Online; accessed 9-February-2020]
31. G. Wood et al., Ethereum: A secure decentralised generalised transaction ledger. Ethereum Project Yellow Paper **151**(2014), 1–32 (2014)
32. B. Zheng, L. Zhu, M. Shen, X. Du, J. Yang, F. Gao, Y. Li, C. Zhang, S. Liu, S. Yin, Malicious bitcoin transaction tracing using incidence relation clustering, in *International Conference on Mobile Networks and Management* (Springer, 2017), pp. 313–323
33. B.-K. Zheng, L.-H. Zhu, M. Shen, F. Gao, C. Zhang, Y.-D. Li, J. Yang, Scalable and privacy-preserving data sharing based on blockchain. J. Comput. Sci. Technol. **33**(3), 557–567 (2018)
34. Z. Zheng, S. Xie, H. Dai, X. Chen, H. Wang, An overview of blockchain technology: Architecture, consensus, and future trends, in *2017 IEEE International Congress on Big Data (BigData Congress)* (IEEE, 20170), pp. 557–564
35. L. Zhu, B. Zheng, M. Shen, S. Yu, F. Gao, H. Li, K. Shi, K. Gai, Research on the security of blockchain data: A survey. arXiv preprint arXiv:1812.02009, 2018

Chapter 3
Layered Data Sharing Architecture with Blockchain

With the development of blockchain technology, innovative application scenarios for data sharing combined with blockchain are emerging on a large scale [15, 22, 24]. However, due to the high research and development costs of building data sharing blockchain applications in different scenarios [14], this chapter proposes a layered data sharing architecture with blockchain. First, the architecture is outlined, including design principles, functional overview, and layered elaboration. Then, the design ideas of each layer are described in detail according to the overall architecture, including the design of the entity layer, data repository layer, service layer and application layer. The description of each layer not only helps to understand the content of the layer, but also contributes to understand the relationship between the layers. At the end of this chapter, a scenario example based on this architecture is given and analyzed.

3.1 Architecture Overview

The design principles of layered data sharing architecture with blockchain include the following four items:

- High security: Design secure computing components through homomorphic encryption. The operations of data sharing and calculation are completed through the secure computing component. Ensure that data privacy and calculated median privacy are not leaked during data sharing.
- High availability: Collaborative calculation results based on shared data ensure high accuracy. Compared with the calculation results on traditional plaintext data, the calculation results based on this architecture have no loss of accuracy.

M. Shen et al., *Blockchain: Empowering Secure Data Sharing*,
https://doi.org/10.1007/978-981-15-5939-6_3

- Decentralization: Based on the alliance chain, the process of sharing data and multi-party collaborative training models does not require the participation of trusted third parties and rely on smart contracts for management.
- High scalability: The number of participants supported can be expanded according to actual application requirements. The types of participants and data supported are diverse. It can meet the application requirements of different scenarios without affecting overall performance.

The overall function of layered data sharing architecture with blockchain is mainly reflected in four aspects, as shown in Fig. 3.1, from the bottom up, it is the entity layer, the data repository layer, service layer and application layer. The architecture is based on IoT equipment, vehicle network equipment, etc., and surrounds underlying cloud platforms, such as public, private, and hybrid clouds.

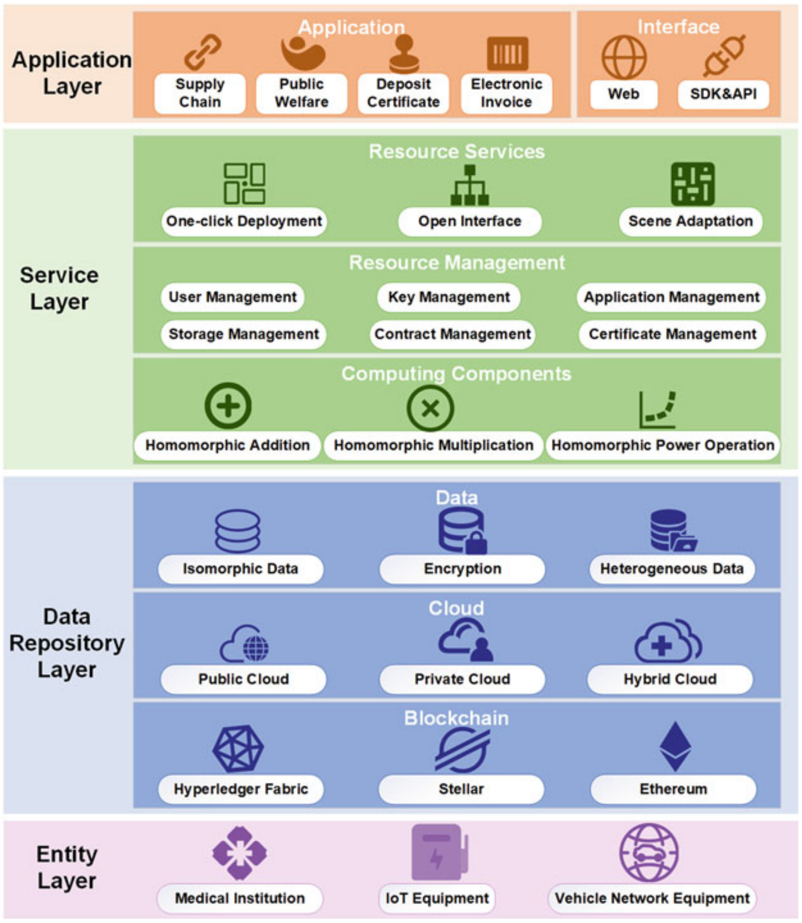


Fig. 3.1 Blockchain-based layered data sharing architecture

And the underlying blockchain platforms, such as Hyperledger Fabric [1, 3] and Stellar. This architecture is designed to provide one-stop services to effectively implement rapid construction and deployment of blockchain applications, and rapid implementation of smart contracts and scenarios. Its driving force lies in the combination of secure data sharing with different industries, especially in the financial and medical fields, such as trusted storage and smart healthcare. Specifically, the functions of each layer are described below.

- Entity layer: The entity layer is the entity participating in data sharing and calculation. They have different roles in different application fields. These entities play the role of data provider or data requester in the process of data processing. At the same time, the data provider needs to cooperate with the data requester to complete the safe training process.
- Data repository layer: The data layer is mainly responsible for data storage. After the data provider of the entity layer encrypts the data, it will upload it to the data layer. The data repository layer is based on the consortium chain such as fabric. At the same time, for the data not uploaded to the blockchain, this part will be stored on the cloud platform.
- Service layer: The service layer is mainly responsible for the identity management of the participants in the entity layer. Through the alliance chain platform and other identity authentication measures, the identity of the participants is guaranteed to be reliable and the data on the consortium blockchain is not leaked. At the same time, it provides all kinds of safe computing operations for the shared data of the secret state. The key management and contract management in the operation process are also completed by the service layer.
- Application layer: Implement machine learning algorithm and other functions based on various secure computing operations, and ultimately serve different applications, including supply chain, etc. At the same time, it provides interfaces to other industries to provide available functions.

3.2 Design of Entity Layer

The lowest layer of this layered data sharing architecture is the entity layer. In order to meet the needs of researchers for different types of data and different application scenarios, this layer is designed to include multiple entities such as medical institutions [10, 17], IoT equipments in smart cities [19, 20], and vehicle network equipments (i.e. roadside devices, mobile terminals, vehicles) [6, 21]. So as to provide a basis for data source for the adaptation of various scenarios.

The researcher collects the required data from the entities in the target scene domain for subsequent research. Therefore, the entity layer is a valuable asset for researchers. Researchers usually obtain large amounts of data directly from them, then perform certain special preprocessing on the obtained data, and then store the data.

After being authenticated, the participants in the entity layer join the alliance chain as nodes, and then they can share or query the data on the alliance chain. Because the architecture supports various types of physical devices, these devices belong to different industries, and there are more than one entity group participating in data sharing, so there are more than one alliance chain formed. On the one hand, these alliance chains can be independent of each other, and on the other hand, data isolation between different groups can be achieved on one alliance chain by setting up channels.

The identities of the participants can be data providers, and they can also be data requesters. Participants who are data providers need to pre-process the data. These include standardization of data formats and data encryption. The private key for data encryption is saved by itself, and the public key will be shared with other participants in the alliance chain. Because the amount of data supported for each transaction is limited, in order to improve the performance of shared data, the original confidential data will not be directly uploaded to the transaction through the chain, but the hash value of the confidential data will be uploaded to the blockchain.

For pre-processed data, upload it to the data layer through smart contracts. As a participant of the data requester, the data is obtained from the blockchain through smart contracts. If the data is hashed, then the dense data will be obtained by off-chain transmission. The hash value of the dense data is compared with the hash value obtained on the chain to verify the data content.

3.3 Design of Data Repository Layer

In order to meet the needs of different blockchain underlying technologies, the underlying blockchain of the data repository layer of this layered data sharing architecture is designed to be compatible with multiple open source blockchain engine technologies, such as Hyperledger Fabric and Stellar [5, 8]. Researchers can choose freely according to the scene, and each type of data sharing blockchain technology has its own characteristics. Among them, Fabric-based technology has a highly modular and configurable design, supports pluggable consensus protocols, and can use mainstream development languages to develop smart contracts. Stellar-based technology has features, such as security, trustworthiness, autonomous controllability, high availability and flexible adaptation. It has strong financial attributes and is therefore suitable for financial business scenarios. The consensus mechanism determines the implementation method and application scenario of the data consistency of the blockchain. The underlying architecture supports multiple consensus algorithms, which can be selected based on different needs [12, 13].

In addition, the data repository layer of the layered data sharing architecture is designed to enable researchers to collaboratively deploy data sharing blockchains on public, private, and hybrid clouds. This cross-cloud networking capability makes the deployment of secure data sharing blockchain easier and more flexible, the diversity of infrastructure construction for data sharing blockchain application scenarios is improved. At the same time, cloud storage has unparalleled advantages in terms of scalability, pay-as-you-go, reliability, availability, and security [11]. Different application scenarios have different requirements for the storage system. The design of this architecture can meet different needs and provide multiple storage solutions. Cloud services follow standard blockchain underlying protocols, helping to build a cloud platform that is compatible with network protocols [16, 18, 25], which can reducing strong dependence on underlying technologies and improve the credibility of the blockchain architecture itself.

The data involved in this layer is collected from the equipment at the entity layer of the architecture. After processing such as encryption, data types such as homogeneous data and heterogeneous data are formed, and then the corresponding data is stored.

3.4 Design of Service Layer

Relying on the support of the underlying secure data sharing blockchain, the service layer design of the layered data sharing architecture abstracts a series of service modules and integrates service resources. In general, it includes three modules: resource service, resource management, and computing components. The resource service component mainly helps researchers quickly deploy secure data sharing blockchain technology, provides rich functions, and reduces the barriers to entry for data sharing blockchains. Among them, the one-click deployment and scene adaptation functions allow users to choose a suitable deployment method according to different scenarios. The resource management service mainly manages users and certificates, and manages on-chain contracts. The computing component service is designed based on homomorphic encryption algorithms (addition homomorphism, multiplication homomorphism, threshold addition homomorphism), helping researchers to solve problems such as sharing calculation intermediate values based on data in the data storage layer, and reading data from the chain. The service layer focuses on the following services:

- Privacy Protection Service

As the secure data sharing function has higher requirements for management and privacy protection, the architecture uses digital certificate-based management, as well as multi-chain isolation, information encryption, and smart contract control to protect private information [9, 26, 27].

- Contract Management

Data sharing blockchain business capabilities need to revolve around the core of smart contracts to implement functions such as smart contracts, security isolation, business definition, and digital protocols [2, 4, 23]. The smart contract integrated development and debugging process designed by this architecture, as shown in Fig. 3.2.

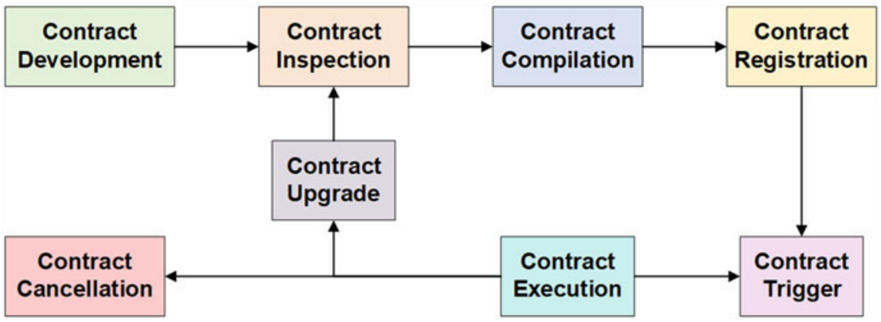


Fig. 3.2 Smart contract service process

• Key Management

Ensure the security of data from three aspects. The first is channel security. During the key transmission process, the API interface enforces SSL/TLS two-way authentication. The second is access security, which provides a complete access control policy. Operations blocked by the policy are prohibited access. The third is storage security, with a complete data encryption system. The root private key is distributed into X shares through key distribution technology, and Y ($X \geq Y$) is required to unlock the data.

3.5 Design of Application Layer

In order to meet the different needs of different researchers, the application layer of the layered data sharing architecture also designed a web console and SDK and API interface. The web console is suitable for business application scenarios. The external API interface adopts the openAPI standard, and a multi-language version of the SDK is provided to easily interface with external systems.

Data sharing applications are decoupled through interfaces and based on rich service interfaces, this architecture can support multiple data sharing business scenarios to meet the needs of each scenario.

The architecture is currently being implemented in multiple data sharing scenarios, including electronic bills, supply chain finance, smart healthcare, and charity. In the following, the electronic data scenario of trusted certificate for secure data sharing blockchain is introduced as an example.

Driven by policy, economic, and social factors, the electronic certificate market has shown a rapid development trend. Traditional electronic data certificate has problems such as long process of fairness and records, slow evidence identification, and the risk of fraud in records, etc., and the high cost of rights maintenance in the process of data preservation. The decentralized features of the blockchain, immutable data on the chain, shared and searchable on-chain records, and the multi-party trust and data sharing mechanism provided by the blockchain, make blockchain technology a good solution for secure data sharing problem. It is helpful to effectively solve the problems of notarization, long data records and business processes, numerous documents, and data falsification and tampering in the real data sharing business scenario. For example, Spain's Stampery uses blockchain technology to help judicial authorities, notarization, arbitration, auditing and other authoritative institutions to record and deposit various legal documents [7].

Figure 3.3 shows the electronic data scenario architecture of the secure data sharing trusted certificate. This electronic data trusted certificate platform architecture can provide one-stop service for trusted certificate of electronic data, with features such as simplicity, ease of use, openness and flexibility, multi-service application, and credible guarantee. It uses electronic signatures, trusted timestamps, hashing, blockchain and other technologies to ensure the integrity and reliability of electronic data, while enhancing the legal validity of electronic documents. Electronic agreements, contracts, orders, emails, web pages, voice, pictures, and other electronic data can be stored in certificates, which are suitable for multiple industries. Users don't need to care about the underlying details of the blockchain to quickly realize the trusted certificate of secure data based on the blockchain.

The bottom layer of the architecture platform can cooperate with multiple parties to consensus, store, and endorse the data, which can make the electronic data more credible, reduce the time required to judge the authenticity of the data, reduce the need for judicial authentication or notarization, and can effectively

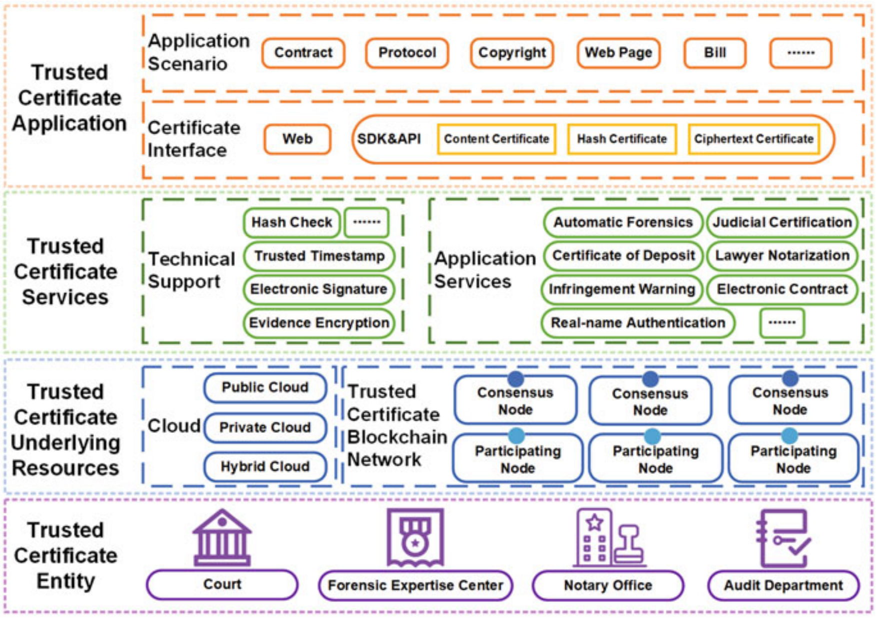


Fig. 3.3 Electronic data scenario architecture of blockchain trusted certificate

improve the rights of the parties effectiveness. Obtain important electronic data files such as business orders and protocols through an automated process, extract their digital fingerprints through a hash algorithm, store them on a secure data sharing blockchain trusted certificate platform, and record them in the blockchain. Utilize its characteristics of decentralization and non-tampering, combined with electronic signatures, time stamping and other technologies to ensure the integrity and credibility of electronic data, protect the judicial rights of the architecture platform and users, and improve efficiency. Promote the protection of rights and intellectual property rights, and form a good atmosphere of honesty and trustworthiness in the whole society.

3.6 Summary

In general, layered data sharing architecture with blockchain strives to provide researchers with a secure data sharing blockchain service foundation and helps researchers build highly secure and trusted blockchain services.The architecture is designed to be compatible and supports multiple blockchain engines, which enhances the capabilities of rights management, security control and privacy protection according to the needs of different data sharing scenarios.

In the future, we hope that the layered data sharing architecture with blockchain can continue to help the innovation of data sharing scenarios while reducing the threshold of blockchain technology application. Utilize the characteristics of blockchain technology such as decentralization, credibility, process-driven and transaction traceability to further realize secure data sharing and highly unified information. Users can conveniently, efficiently and securely enjoy the value of blockchain technology, make value linkers, make due contributions to the prosperity of cloud technology and blockchain technology, and actively promote the blockchain community and ecology Construction.

References

1. E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich, et al., Hyperledger fabric: a distributed operating system for permissioned blockchains, in *Proceedings of the Thirteenth EuroSys Conference*, pp. 1–15, 2018
2. V. Buterin et al., A next-generation smart contract and decentralized application platform. White Paper **3**, 37 (2014)
3. C. Cachin et al., Architecture of the hyperledger blockchain fabric, in *Workshop on Distributed Cryptocurrencies and Consensus Ledgers*, vol. 310, p. 4, 2016
4. Ethereum, A next-generation smart contract and decentralized application platform. Accessed 30 December 2019. https://github.com/ethereum/wiki/wiki/White-Paper, 2019
5. H. Fabric, Hyperledger fabric website. Accessed 30 December 2019. https://www.hyperledger.org/projects/fabric, 2019

6. F. Gao, L. Zhu, M. Shen, K. Sharif, Z. Wan, K. Ren, A blockchain-based privacy-preserving payment mechanism for vehicle-to-grid networks. IEEE Netw. **32**(6), 184–192 (2018)
7. R. Kastelein, Blockchain startup stampery announces 600k dollar investment round led by draper and associates. Accessed 30 December 2019. https://www.the-blockchain.com/2015/11/23/blockchain-startup-stampery-announces-600k-investment-round-led-by-draper-associates/, 2015
8. A. Kiayias, G. Panagiotakos, On trees, chains and fast transactions in the blockchain, in *International Conference on Cryptology and Information Security in Latin America* (Springer, 2017), pp. 327–351
9. A. Kosba, A. Miller, E. Shi, Z. Wen, C. Papamanthou, Hawk: The blockchain model of cryptography and privacy-preserving smart contracts, in *2016 IEEE Symposium on Security and Privacy (SP)* (IEEE, 2016), pp. 839–858
10. H. Li, L. Zhu, M. Shen, F. Gao, X. Tao, S. Liu, Blockchain-based data preservation system for medical data. J. Med. Syst. **42**(8), 141 (2018)
11. Y. Li, L. Zhu, M. Shen, F. Gao, B. Zheng, X. Du, S. Liu, S. Yin, Cloudshare: Towards a cost-efficient and privacy-preserving alliance cloud using permissioned blockchains, in *International Conference on Mobile Networks and Management* (Springer, 2017), pp. 339–352
12. J. Mattila, The blockchain phenomenon—the disruptive potential of distributed consensus architectures. Technical report, ETLA working papers, 2016
13. A. Miller, J.J. LaViola Jr., Anonymous byzantine consensus from moderately-hard puzzles: A model for bitcoin. Available on line: http://nakamotoinstitute.org/research/anonymous-byzantine-consensus, 2014
14. M. Pilkington, Blockchain technology: principles and applications, in *Research Handbook on Digital Transformations*, vol. 225 (2016)
15. G. Sachs, Blockchain—putting theory into practice, in *the-blockchain.com*, pp. 25–32 (2016)
16. P.K. Sharma, M.-Y. Chen, J.H. Park, A software defined fog node based distributed blockchain cloud architecture for iot. IEEE Access **6**, 115–124 (2017)
17. M. Shen, Y. Deng, L. Zhu, X. Du, N. Guizani, Privacy-preserving image retrieval for medical iot systems: A blockchain-based approach. IEEE Netw. **33**(5), 27–33 (2019)
18. M. Shen, J. Duan, L. Zhu, J. Zhang, X. Du, M. Guizani, Blockchain-based incentives for secure and collaborative data sharing in multiple clouds. IEEE J. Sel. Areas Commun. **38**(6), 1229–1241 (2020)
19. M. Shen, H. Liu, L. Zhu, K. Xu, H. Yu, X. Du, M. Guizani, Blockchain-assisted secure device authentication for cross-domain industrial IoT. IEEE J. Sel. Areas Commun. **38**(5), 942–954 (2020)
20. M. Shen, X. Tang, L. Zhu, X. Du, M. Guizani, Privacy-preserving support vector machine training over blockchain-based encrypted iot data in smart cities. IEEE Internet Things J. **6**(5), 7702–7712 (2019)
21. M. Shen, J. Zhang, L. Zhu, K. Xu, X. Tang, Secure SVM training over vertically-partitioned datasets using consortium blockchain for vehicular social networks. IEEE Trans. Veh. Technol. **69**(6), 5773–5783 (2020)
22. M. Swan, *Blockchain: Blueprint for a New Economy* (O'Reilly Media, 2015)
23. N. Szabo, Smart contracts: building blocks for digital markets. EXTROPY: J. Transhumanist Thought (16) **18**, 2 (1996)
24. M. Walport, Distributed ledger technology: beyond block chain (a report by the uk government chief scientific adviser). *UK Government* (2016)
25. Q. Xia, E.B. Sifah, K.O. Asamoah, J. Gao, X. Du, M. Guizani, Medshare: Trust-less medical data sharing among cloud service providers via blockchain. IEEE Access **5**, 14757–14767 (2017)
26. B.-K. Zheng, L.-H. Zhu, M. Shen, F. Gao, C. Zhang, Y.-D. Li, J. Yang, Scalable and privacy-preserving data sharing based on blockchain. J. Comput. Sci. Technol. **33**(3), 557–567 (2018)
27. G. Zyskind, O. Nathan, et al., Decentralizing privacy: Using blockchain to protect personal data, in *2015 IEEE Security and Privacy Workshops* (IEEE, 2015), pp. 180–184

Chapter 4
Secure Homogeneous Data Sharing Using Blockchain

Machine learning (ML) technology has been widely used in many smart city areas, which employs a large amount of data from various IoT devices. As a typical ML model, Support Vector Machine (SVM) supports efficient data classification, which is applied in many application of real world, such as disease diagnosis and anomaly detection. Training SVM classifiers usually requires to collect labeled IoT data from multiple entities, which leads to data privacy concern. Most existing solutions are based on an implicit assumption that an ideal server could be employed for reliably collecting data from multiple data providers, but it is often not the case. To bridge the gap between ideal assumptions and realistic constraints, this chapter proposes secureSVM, a privacy-preserving SVM training scheme over blockchain-based encrypted IoT data. SecureSVM utilizes the blockchain techniques to build a secure and reliable data sharing platform among multiple data providers, where IoT data is encrypted and then recorded on a distributed ledger. After detailing the background of homogeneous IoT data sharing, we introduce the proposed method. Then, we present the comprehensive experiments for demonstrating the efficiency of secureSVM.

4.1 Background Description

Support Vector Machine (SVM) is a well-known supervised learning model among all ML models, which efficiently performs data classification on many data analysis applications. Therefore, SVM has been adopted in many fields to solve the real classification problem in IoT-based smart cities. In the case of personal healthcare, health records monitored by wearable IoT sensors can be fed to the SVM classifier for accurate diagnosis. In the field of network intrusion detection, the SVM classifier can be used to identify anomalies from a series of traffic data originating from communication between IoT devices [24, 29].

M. Shen et al., *Blockchain: Empowering Secure Data Sharing*,
https://doi.org/10.1007/978-981-15-5939-6_4

The construction of a supervised ML classifier (such as SVM) is called a training phase. From a set of labeled samples, training phase trains a specific classifier. Since training data sets owned by a single entity [38, 40], such as a hospital, are often limited in terms of data volume and variety, for a long time, a keen need for an effective mechanism is to train ML classifiers using a combination of data sets collected from multiple institutions.

However, since the privacy concerns around data privacy, integrity and ownership, different institutions refuse to share their data with others. Sharing these privacy data is seen as confidential information leakage [2, 18, 42]. What's more, during data sharing for training, data records may be tampered with or modified without authorization by potential attackers, resulting in inaccurate classifiers. Finally, because participants have access to the shared dataset and can be replicated freely by others, the data provider may lose control of the data [33, 34].

A large amount of research attention has been attracted on the data privacy issues of training ML classifiers. Existing methods utilize differential privacy or cryptographic algorithms to protect the data privacy during ML training process [1, 6, 27, 36, 39]. These methods relying on an assumption that the training data can be collected reliably from multiple data providers for further analysis. But these methods pay little attention to the concerns of data integrity and ownership.

To bridge the gap between ideal assumptions and realistic constraints, we proposed secureSVM which employs blockchain techniques to build a secure data sharing platform. Blockchain is essentially database of records designed to allow the sharing of tamper-proof records among multiple parties [21, 28, 30]. Permanent and immutable records on the blockchain can be audited for data and can be used to confirm ownership of data records. This also helps quantify the contribution of individual data providers and design incentive strategies to encourage sharing of training data.

Incorporating blockchain into the ML training process remains a daunting task. The first challenge is to design an appropriate training data format that is able to accommodated on a blockchain. The second challenge is the construction of a secure training algorithm employing use users's data without revealing sensitive information.

To address the above challenges, secureSVM uses blockchain-based encrypted IoT data and employs a public-key cryptosystem, Paillier, to protect the privacy of IoT data, where data providers encrypt their data locally by their own private keys. By Paillier cryptosystem with blockchain techniques, secureSVM addresses the concerns about data privacy, integrity and ownership. More precisely, IoT data from data owners is encrypted by data owners' private key and then recorded on a distributed ledger. Data analysts who want to train the SVM classifier can access the encrypted data by communicating with the corresponding data owners. To perform training tasks based on encrypted data, this paper constructs security protocols for two key operations in SVM training, namely secure polynomial multiplication and secure comparison.

4.2 Development of Privacy-Preservation ML Training

In recent years, smart cities are incorporating more and more advanced Internet-of-Things (IoT) infrastructures, resulting a huge amount of data gathered from various IoT devices deployed in many city sectors, such as transportation, manufactory, energy transmission, and agriculture [31, 44]. Some institutions desiring to conduct ML modeling on their joint dataset own IoT data with the same data format falling in homogeneous data sharing [2, 10, 41]. To address the challenges posed by homogeneous IoT data processing requirements, many innovations driven by machine learning (ML) technology have been proposed [1, 3, 37].

The typical supervised learning consists of two phases: the training phase and the classification phase. The training phase learns an ML model from a given set of labelled samples. Existing studies on privacy-preserving ML training of homogeneous data sharing are reviewed as below.

ML model training usually involves multiple participants, so the privacy goal is to use the dataset gathered from the involved participants to train a ML model, while protecting the data of a single participant from being learned by other participants. Many solutions have been proposed over the past decades [8, 15, 16, 23].

Differential privacy (DP) is a commonly used technique to protect data privacy in the publishing stage [1, 20, 45]. More specifically, DP ensures the security of published data by adding carefully calculated perturbations to the original data [5, 13, 35]. DP-based solutions can achieve high computational efficiency, as all calculations are performed over plaintext data [17, 25, 43]. Abadi et al. [1] proposed a DP-based deep learning scheme enabling multiple parties to jointly learn a neural network while protecting the sensitive information of their datasets. However, the resulting ML models with DP can be inaccurate as the perturbations inevitably reduce the quality of training data. In addition, perturbations may not protect the data privacy completely, as a bounded amount of sensitive information about each individual training data is exposed.

Many *homomorphic cryptosystem* (HC) based methods are proposed to train ML models based on encrypted data for achieving better privacy guarantees in ML training [15, 23, 37]. HC-based solutions achieve higher data confidentiality than DP-based solutions but have a higher time consumption [4, 8, 16, 22]. Fully homomorphic Encryption (FHE) supports complex calculations of ciphertext (for example, through any combination of addition and multiplication). The current implementation of FHE causes encryption and calculation costs to be too high, making it impractical in practical applications, Comparing to FHE, partial homomorphic encryption (PHE) is more practical, but supports only one type of operation (i.e., addition or multiplication). Therefore, in order to enable complex computations, existing solutions usually rely on a trusted third-party (e.g., the authorization server [15]) or lead to inaccurate models by approximating complex equations with a single type of computation [2, 3].

Serrano et al. [15] proposed a scheme based on a partial homomorphic cryptosystem for secure SVM training with three parties: Data Owners, who own private data; an Application, which wants to train and use an arbitrary machine learning model on the users' data; and an Authorization Server. Graepel et al. [16] proposed a secure machine learning algorithm using a homomorphic encryption scheme to a computing service while retaining confidentiality of the training and test data.

4.3 Preliminaries

4.3.1 Notation

A unordered dataset D records m records with the size of $|D|$, where x_i is the i-th record in D and y_i is the corresponding label of x_i. Define w and b as two relevant parameters of SVM. ∇_t is the descend gradient in the iterative execution of the SVM training algorithm, λ is the learning rate. $[[m]]$ represents the encryption of m under Paillier. Table 4.1 summarizes the notations used in the following sections.

4.3.2 Homomorphic Cryptosystem

Cryptosystems generally include three algorithms:

- key generation ($KeyGen$).
- encryption (Enc).
- decryption (Dec).

Cryptosystems has a pair of keys (PK, SK). Among the pair of keys of public-key cryptosystems, the public key PK is adopted for encryption, and the private key SK is adopted for decryption.

Table 4.1 Notations

Notations	Explanation
D	Dataset
d	Dataset dimension
x_i	i-th record in dataset
y_i	Class lable
∇_t	Gradient
λ	Learning rate
m	Size of dataset D
w, b	The model parameters
$[[m]]$	The encryption of m under Paillier
$\phi(N)$	The Euler phi-function

Homomorphism means that a cryptographic system can map operations on ciphertext to corresponding plaintext without knowing the decryption key, so it is the homomorphism of the cryptographic system. We formally define the homomorphic nature of a cryptographic system in Definition 4.1.

Definition 4.1 (Homomorphic [19]) A public-key encryption scheme (Gen, Enc, Dec) is homomorphic if for all n and all (PK, SK) output by $Gen(1^n)$. It is possible to define groups $\mathscr{M}$, C (depending on PK only) such that:

(i) *The message space is $\mathscr{M}$, and all ciphertexts output by Enc_{pk} are elements of C.*
(ii) *For any $m_1, m_2 \in \mathscr{M}$, any c_1 output by $Enc_{pk}(m_1)$, and any c_2 output by $Enc_{pk}(m_2)$, it holds that*

$$Dec_{sk}(o(c_1, c_2)) = \sigma(m_1, m_1)$$

We adopt the homomorphic cryptosystem Paillier on our method. Paillier is a public key encryption scheme with homomorphic properties that allows the two important operations of secure addition and secure subtraction. Paillier is an encryption system based on an assumption related to the hardness of factoring. Let p and q are n-bit primes, $N = pq$. The public key is N, and the private key is $(N, \phi(N))$[1] in Paillier. Encryption function in Paillier is

$$c := [[(1+N)^m r^N modN^2]], m \in Z_N \tag{4.1}$$

Decryption function in Paillier is

$$m := [[\frac{[c^{\phi(N)} modN^2 - 1]}{N} \times \phi(N)^{-1} modN]] \tag{4.2}$$

For more details about Paillier, we refer the reader to [19].

4.3.3 Support Vector Machine

SVM is a supervised learning model which gives the maximum-margin hyperplane that might classify the test data [9]. The form of the hyperplane is expressed as:

$$y = w^T x + b, (x_i, y_i) \in D \tag{4.3}$$

$$w^T x_i + b \geq 1, y_i = +1 \tag{4.4}$$

$$w^T x_i + b \geq 1, y_i = +1 \tag{4.5}$$

[1]Let $N > 1$ be an integer. Then Z_N^* is an abelian group under multiplication modulo N. Define $\phi(N) \underline{\underline{def}} \left|Z_N^*\right|$, the order of the group Z_N^*.

The optimization problem of the primary of SVM as follows:

$$\begin{gathered}\min_{\mathrm{w,b}}\frac{1}{2}||w||^2\\ s.t,\ y_i\left(w^T x_i + b\right) \geq 1,\ \ \mathrm{i} = 1, 2, \ldots\ldots m.\end{gathered} \tag{4.6}$$

4.4 System Model

We designed a data-driven ecosystem for the IoT, as shown in Fig. 4.1. The system includes IoT devices, IoT data providers, blockchain-based IoT platforms and IoT data analysts.

- IoT devices are capable of sensing and transmitting valuable data over wireless or wired networks such as ZigBee, 3G/4G, and WiFi. These data cover a wide range of practical applications in smart cities, from environmental conditions to physiological information. Due to the limited computing power of IoT devices, these devices do not participate in the data sharing and analysis process.
- IoT data providers collect all pieces of data from the IoT devices within their own domains. As a valuable asset of data providers, IoT data usually contains sensitive information. Thus, each data provider encrypts its IoT data using partially homomorphic encryption and then records the data on blockchain.
- IoT data providers collect IoT data from IoT devices in their respective domains for machine learning training. As a valuable asset for data providers, IoT data often contain sensitive information. As a result, each data provider uses homomorphic encryption to encrypt their IoT data with their locally held private keys and then records the data on the blockchain.
- IoT data analysts aim to gain insight into IoT data recorded on the blockchain-based platform by making full use of emerging analysis technologies, i.e., building a SVM model. The data analyst will communicate with the corresponding data provider to obtain the parameters for training the SVM classifier.

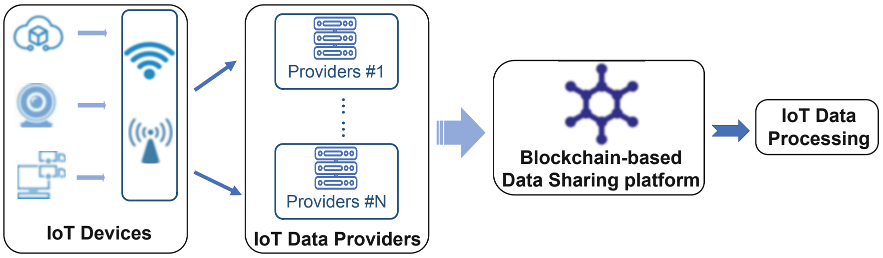


Fig. 4.1 System model of data-driven IoT ecosystem

4.5 Threat Model

As described in Fig. 4.1, each entity and its previous interactions have multiple potential threats. Since we want to design a privacy protection scheme to train a SVM model on the gathered data from multiple IoT providers, we consider the threat to data privacy when interacting between data providers and data analysts.

We see the data analyst as an *honest-but-curious* adversary. That is, the data analyst will honestly follow the pre-designed ML training protocol, but may be curious about the content of the data and try to learn other information by analyzing encrypted data as well as calculated intermediate data.

Based on the sensitive information available to data analysts, we consider the following two threat models with different attack capabilities, which are commonly used in the literature [32, 46]:

- *Known Ciphertext Model.* IoT data analysts can only access encrypted IoT data recorded on blockchain-based platforms. IoT data analysts can also record intermediate results during the execution of security training algorithms.
- *Known Background Model.* In this stronger model, the IoT data analyst is assumed to be aware of more facts than what can be known in the
- *known ciphertext model.* In particular, the IoT data analyst can collude with one or more IoT data providers to infer the sensitive data of other IoT data providers.

We make the following assumptions. Any two or more IoT data providers to collude with IoT data analysts to steal the privacy of other participants; as honest-but-curious adversaries, each participant performs the protocol honestly, but may be interested in private information from other domains. As passive adversaries, any two or more participants may collude with each other, trying to infer as much privacy as possible from the values they learn.

4.6 The Construction of SecureSVM

This method assumes that a data analyst aims to use data collected from multiple IoT data providers to train the SVM model. Each IoT data provider preprocesses the IoT data instance, encrypts it locally using its own private key, and records it in a blockchain-based shared ledger by generating transactions. Data analysts who want to train the SVM model can access the encrypted data recorded in the global ledger. During training, the interaction between the data analyst and each data provider is necessary to exchange intermediate results (Fig. 4.2).

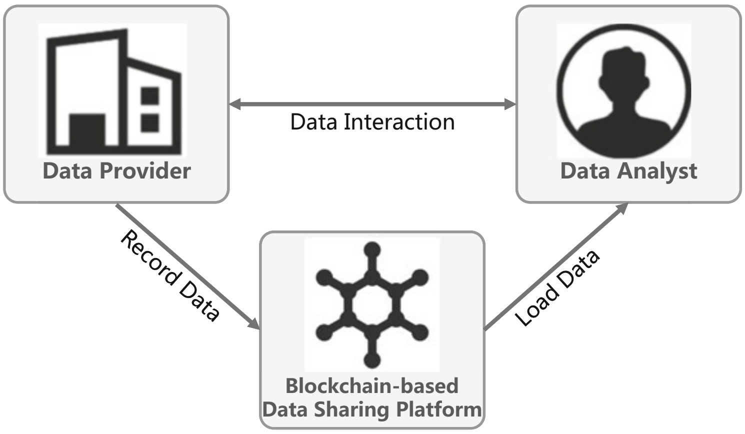


Fig. 4.2 An overview of secureSVM system

4.6.1 Encrypted Data Sharing via Blockchain

In order to promote model training without losing generality, we assume that data instances of the same training task are preprocessed locally and represented by the same eigenvectors [39].

To store encrypted IoT data on blockchain, We defined a particular transaction structure. The transaction format consists of two main fields: input and output. The input fields include the address of the data provider, the encrypted data, and the IoT device type. The corresponding output fields include the data analyst address, the encrypted data, and the IoT device type. The addresses in both fields are hash values of 32 bytes. The encrypted data is from the homomorphic encryption Paillier. Based on the assumption that the private key is 128 bytes, each encrypted data instance stored in the blockchain is 128 bytes long. The IoT device type segment is 4 bytes.

After the new transaction is built, the node representing the data provider in the blockchain network broadcasts the data provider in the P2P network, where the miner can verify the correctness of the transaction. Using existing consensus algorithms (such as the PoW mechanism), a specific miner node is eligible to package the transaction in a new block and add that block to the existing chain. Note that each block may contain multiple transactions.

4.6.2 Building Blocks

Gradient Descent The model parameters of SVM can be solved by several optimization methods. These algorithms include sequential minimum optimization (SMO) and gradient descent (GD). SMO is an optimization method of SVM biquadratic program [26]. Moreover, it performs well for linear SVM and data sparsity. However, the large number of comparisons, dot product and division operations [15] contained in SMO make it complex. Applying SMO in the cryptographic manner lead to terrible computing and communication costs. The SVM optimization algorithm based on GD is simple and efficient, involving a small amount of comparison and vector multiplication. Therefore, we choose GD as the optimization algorithm in SVM training algorithm to optimize the SVM model parameters.

GD converts the SVM primary into an empirical loss minimization problem with a penalty factor.

$$\min_{\mathrm{w,b}} \frac{1}{2}||w||^2 + C\sum_{i=1}^{m} \mathrm{L}\left(\mathrm{w, b}, (x_i, y_i)\right) \tag{4.7}$$

where the right part of the equation is the hinge-loss function

$$C\sum_{i=1}^{m} \mathrm{L}(\mathrm{w, b}, (x_i, y_i)) = C\sum_{i=1}^{m} \max\left\{0, 1 - y_i\left(wx_i - b\right)\right\} \tag{4.8}$$

C is the misclassification penalty that usually takes the value $\frac{1}{m}$.

The basic form of GD is

$$x_{n+1} = x_n - \lambda\nabla\mathrm{Grad}\left(x_n\right) \tag{4.9}$$

The gradient calculation formula of the SVM is exhibited in Eq. (4.10), where $\mathrm{I}|\left(wx + b\right) < 1|$ is the indication function, and if $\left(wx + b\right) < 1$ is true, the value is 1, otherwise it is 0. The steps of the SVM model training algorithm using gradient descent are shown in Fig. 4.3:

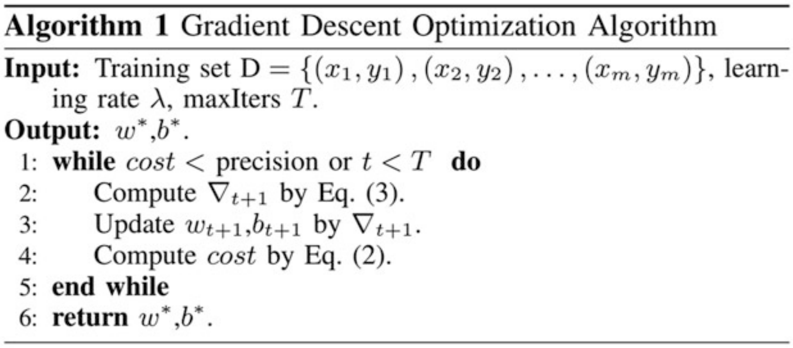
$$\nabla_t = \lambda w_t - \sum_{i=1}^{m} \mathrm{I}|\left(wx_i + b\right) < 1| \times x_i y_i \tag{4.10}$$

Secure Polynomial Multiplication Using Paillier's homomorphic property, secure additions can be described as

$$[[m_1 + m_2]] = [[m_1]] \times [[m_2]] \left(\mathrm{mod}N^2\right) \tag{4.11}$$

and the secure subtraction can be described as

$$[[m_1 - m_2]] = [[m_1]] \times [[(m_2)]]^{-1} \left(\mathrm{mod}N^2\right) \tag{4.12}$$



Algorithm 1 Gradient Descent Optimization Algorithm

```
Input: Training set D = $\{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$, learning rate $\lambda$, maxIters $T$.
Output: $w^*$,$b^*$.
1: while $cost$ < precision or $t < T$ do
2:     Compute $\nabla_{t+1}$ by Eq. (3).
3:     Update $w_{t+1}$,$b_{t+1}$ by $\nabla_{t+1}$.
4:     Compute $cost$ by Eq. (2).
5: end while
6: return $w^*$,$b^*$.
```

Fig. 4.3 Gradient descent optimization algorithm

$[[m]]^{-1}$ is the modular multiplicative inverse, which preforms that

$$[[m]] \times [[m]]^{-1} mod N^2 = 1 \tag{4.13}$$

$[[m]]^{-1}$ can be computed by function

$$\phi(N), [[m]]^{-1} = [[m]]^{\phi(N)-1} \tag{4.14}$$

Thereby, the secure polynomial multiplication can be obtained through ciphertext manipulation, as shown in Eq. (4.15).

$$[[am_1 + bm_2]] = [[m_1{}^a]] \times [[m_2{}^b]] \left(\text{mod} N^2\right) \tag{4.15}$$

The security of secure polynomial multiplication constructed by Paillier depends on Paillier's statistically indistinguishable. Thus, secure polynomial multiplication is statistically indistinguishable, as Paillier is statistically indistinguishable [19].

Secure Comparison For party A and party B participating in the secure comparison algorithm, neither party can obtain any information other than that are inferred from by the input. Our secure comparison protocol is shown in Fig. 4.4.

4.6.3 Training Algorithm of SecureSVM

For secure optimization model parameters, we design a privacy-preserving SVM training algorithm. Suppose there are n data providers and a IoT data analyst. we specify the secure SVM training algorithm in Fig. 4.5.

```
**Algorithm 2** Secure Comparison
**Input A:** $[[a]]$,1.
**Input B:** A pair of keys (PK,SK)
**Output B:** $(a < 1)$.
 1: A uniformly picks three positive integers $r_1$, $r_2$ and $r_3$, where
    $|r_3 - r_2| < r_1$.
 2: A sends $[[ar_1 + r_2]]$ and $[[r_1 + r_3]]$ to B.
 3: B decrypts and compares $(ar_1 + r_2)$ with $(r_1 + r_3)$, and tell the
    results to A.
 4: $a > 1$, if and only if $(ar_1 + r_2) > (r_1 + r_3)$; otherwise $a \leq 1$.
 5: **return** $(a < 1)$
```

Fig. 4.4 Secure comparison

```
**Algorithm 3** Privacy-Preserving SVM Training Algorithm
*P*'s **Input:** $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$.
*C*'s **Input:** learning rate $\lambda$, maxIters $T$, a pair of keys $(\mathsf{PK_c}, \mathsf{SK_c})$.
*C*'s **Output:** $w^*$,$b^*$.
 1: C initializes $w^1$, $b^1$.
 2: **while** $cost <$ precision or $t < T$ **do**
 3:     **for** $i = 1$ to $n$ **do**
 4:         C sends $[[w^t]]$ to $P_i$.
 5:         **for** $j = 1$ to m **do**
 6:             $P_i$ computes $[[y_j(wx_j - b)]]$ by secure polynomial
    multiplication.
 7:             $P_i$ compare $[[y_j(wx_j - b)]]$ with 1 by secure com-
    parison.
 8:             $P_i$ updates $\nabla^i_{t+1}$ and $cost^i_{t+1}$ and send to C.
 9:         **end for**
10:         C decrypts $\nabla^i_{t+1}$ and $cost^i_{t+1}$, and updates $w_{t+1}$,$b_{t+1}$.
11:     **end for**
12: **end while**
13: **return** $w^*, b^*$.
```

Fig. 4.5 Privacy-preserving SVM training algorithm

Except for legal input, each participant cannot infer any sensitive information of other participants from the intermediate results of the algorithm's running process when facing honest-but-curious adversaries or any collusion.

4.7 Security Analysis

This section presents the security analysis under the *known ciphertext model* and the *known background model*. We adopt two security definitions: secure two-party computation [14] and modular sequential composition [7]. We present our security proof according to the ideas of these two definitions. For more details, we refer the reader to [14] for secure two-party computation and [7] for modular sequential composition.

Definition 4.2 (Secure Two-Party Computation [14]) A protocol π privately computes f with statistical security if for all possible inputs (a, b) and simulators S_A and S_B hold the following properties:[2]

$$\{S_A, f_2(a, b)\} \approx \{view_A^{\pi}(a, b), output^{\pi}(a, b)\}$$

$$\{f_1(a, b), S_B\} \approx \{output^{\pi}(a, b), view_B^{\pi}(a, b)\}$$

Theorem 4.1 (Modular Sequential Composition [14]) *Let $F_1, F_2, \ldots, F_n$ be two-party probabilistic polynomial time functionalities and $\rho_1, \rho_2, \ldots, \rho_n$ protocols that compute respectively $F_1, F_2, \ldots, F_n$ in the presence of semi-honest adversaries. Let G be a two-party probabilistic polynomial time functionality and π a protocol that securely computes G in the $(F_1, F_2, \ldots, F_n)$—hybrid model in the presence of semi-honest adversaries. Then, $\pi^{\rho_1, \rho_2, \ldots, \rho_n}$ securely computes G in the presence of semi-honest adversaries.*

4.7.1 Security Proof for Secure Comparison

Two roles are involved in Secure Comparison: A and B. The function is F

$$F([[a]]_B, 1, \mathsf{PK}_B, \mathsf{SK}_B) = (\phi, (a < 1)) \tag{4.16}$$

Proof The view of A is:

$$view_A^{\pi} = ([[a]]_B, \mathsf{PK}_B) \tag{4.17}$$

A receives no message form B, whose view only consists of his input and three random numbers. Hence,

$$S_A^{\pi}((a, 1); F(a, 1)) = view_A^{\pi}([[a]]_B, 1, \mathsf{PK}_B) \tag{4.18}$$

[2] $\approx$ denotes computational indistinguishability against probabilistic polynomial time adversaries with negligible advantage in the security parameter λ.

$[[a]]_B$ is encrypted by PK_B, and the confidentiality of $[[a]]_B$ is equivalent to the used cryptosystem Paillier. Therefore, A cannot infer the value directly.

The view of B is:

$$view_B^{\pi} = ((ar_1 + r_2), (r_1 + r_3), \mathsf{PK}_B, \mathsf{SK}_B) \tag{4.19}$$

S_B^{π} runs as follows:

- Generates l random coins and obtains $[[(m_1, m_2, \ldots, m_l)]]_B$ by PK_B, where l is the length of a.
- B uniformly picks three positive integers c_1, c_2 and c_3, where $|r_3 - r_2| < r_1$.
- Outputs $((mc_1 + c_2), (c_1 + r_3), \mathsf{PK}_B, \mathsf{SK}_B)$

The distribution of (a, r_1, r_2, r_3) and (m, c_1, c_2, c_3) are identical, so the real distribution

$$((ar_1 + r_2), (r_1 + r_3), \mathsf{PK}_B, \mathsf{SK}_B) \tag{4.20}$$

and the ideal distribution

$$((mc_1 + c_2), (c_1 + c_3), \mathsf{PK}_B, \mathsf{SK}_B) \tag{4.21}$$

are statistically indistinguishable. ■

4.7.2 Security Proof for SVM Training Algorithm

The roles involved in Privacy-Preserving SVM Training Algorithm are n IoT data providers P and an IoT data analyst C. Each IoT data providers behaves in the same way. If we can prove that each one of them meets the security requirements, then every data provider meets the security requirements. The function is F:

$$F(D_{P_i}, \mathsf{PK}_C, \mathsf{SK}_C) = (\phi, (w_*, b_*)) \tag{4.22}$$

Proof IoT data provider's view is

$$view_A^{\pi} = \left(D_{P_i}, [[w^t]]_C, \mathsf{PK}_C\right) \tag{4.23}$$

$[[w^t]]_C$ is encrypted by PK_C, and the confidentiality of $[[w^t]]_C$ is equivalent to the used cryptosystem Paillier. So each IoT data provider cannot infer the value directly. Hence,

$$S_P^{\pi}(D_{P_i}; F(a, 1)) = view_P^{\pi}(D_{P_i}, [[w^t]]_C, \mathsf{PK}_C) \tag{4.24}$$

IoT data analyst's view is

$$view_C^{\pi} = (\sum_{i=1}^{m} \mathrm{I}|(wx_i + b) < 1| \times x_i y_i, \sum_{i=1}^{m} \max\{0, 1 - y_i(wx_i - b)\}, w_t, \mathsf{PK}_B, \mathsf{SK}_B) \tag{4.25}$$

Now, we need to discuss the confidentiality of

$$\sum_{i=1}^{m} \mathrm{I}|\,(wx_i + b) < 1| \times x_i y_i \tag{4.26}$$

and

$$\sum_{i=1}^{m} \max\{0, 1 - y_i\,(wx_i - b)\} \tag{4.27}$$

Obviously, both of the equations are no-solution equations for the unknown x_i and y_i. Except for brute force cracking, there is no better way to get the real value of dataset D. We assume that each IoT data provider with a small dataset has 2-dimensional 100 instances, and each dimension is 32 bits.[3] Under this circumstance, the probability that IoT data analyst guesses success is $\frac{1}{2^{n*6400}}$. It is a negligible probability of success [19].

As secure comparison and secure polynomial multiplication used are secure in the honest-but-curious model, we obtain the security using modular sequential composition. ■

4.8 Performance Evaluation

In this section, we use two real-world datasets to evaluate the performance of secureSVM.

4.8.1 Experiment Setting

- Testbed

Each IoT data provider collects all data from IoT devices in its own domain and then performs operations on IoT data. Since IoT providers and the data analyst usually

[3]Typically, single-precision floating-point occupies 4 bytes (32-bit) memory space.

Table 4.2 Statistics of datasets

Datasets	Instances number	Attributes number	Discrete attributes	Numerical attributes
BCWD	699	9	0	9
HDD	294	13	13	0

have adequate computing resources, the experiments are run on a PC equipped with a 4-core Intel i7 (i7-3770 64bit) processor at 3.40 GHz and 8 GB RAM, serving as IoT data providers and an IoT data analysts simultaneously. We implemented secure polynomial multiplication, secure comparison, and secure SVM training algorithm in Java development kit 1.8.

- Dataset

Two real-world datasets are used: Breast Cancer Wisconsin Data Set (BCWD) [12] and Heart Disease Data Set (HDD) [11]. The statistics are shown in Table 4.2. We show the average results of cross-validation of 10 runs. In each cross-validation, we randomly take 80% to train the model, and the remainder for testing.

- Floating Point Setting

SVM training algorithms performs on floating point numbers. However, the operations of cryptosystem are carried out on integers. In order to encrypted data correctly, it is necessary to previously perform a format conversion into an integer representation. Binary floating point number D is expressed as $D = (-1)^s \times M \times 2^E$ in the international standard IEEE 754, where s is the sign bit, M is a significant number, and E is the exponent bit. In our implementation of secureSVM, we employ it to perform the format conversion. We keep two decimal places for these.

- Key Length

The length of keys in public-key cryptosystems is approximated as the security of the cryptosystem can provide. The short key length may be insufficient to provide adequate privacy protection, and too short key may cause the plaintext space to overflow. On the other hand, homomorphic operations runs on the ciphertext. A long key length outputs a long ciphertext, which reduce the efficiency of the homomorphic operation.

Therefore, the key length is an important hyper-parameter where we must consider to avoid the possibility of overflow. In secureSVM, Paillier's key N is set to 1024-bit.

4.8.2 Accuracy Evaluation

Two criterions are for evaluating ML classifiers: Precision P is calculated as

$$P = t_p/(f_p + t_p)$$

Recall R is calculated as

$$R = t_p/(f_n + t_p)$$

t_p is the numbers of relevant (i.e., the positive class) that are classified correctly, f_p is the numbers of irrelevant (i.e., the negative class) that are classified correctly and f_n is the numbers of relevant that are classified incorrectly in the test results.

To evaluate the accuracy of the proposed privacy-preserving SVM training scheme secureSVM whether it reduces the accuracy upon protecting the privacy of each IoT data provider and securely training classifiers, we implemented the general SVM with java named *SVM*. We run SVM and the proposed method secureSVM on the two datasets. Table 4.3 summarizes the results of precision and recall.

The reasons that SVM and secureSVM have a different accuracy have two aspects. The one is in order to allow floating point numbers to be processed by the encryption system during training process, we truncated the precision of floating point numbers. Truncated floating-point numbers may lose some parameter accuracy. The other reason is that during each iteration of stochastic gradient descent, the update parameter records are selected randomly. This randomness makes the model inconsistent.

The proposed method has almost the same accuracy as SVM, as can be seen in Table 4.3. This means that truncated floating-point numbers have a small effect on model accuracy. Besides, BCWD with all numerical attributes and HDD with all discrete attributes shows good robustness of the proposed methods on both types of datasets.

4.8.3 Efficiency

- Building Blocks Evaluation

Table 4.4 shows the running time of the secure comparison and secure polynomial multiplication (SPM) with encrypted datasets, which gives the time consumption of

Table 4.3 Summary of accuracy performance

	Precision		Recall	
Datasets	secureSVM	*SVM*	secureSVM	*SVM*
BCWD	90.35%	90.47%	96.19%	97.24%
HDD	93.89%	93.35%	89.78	90.87%

Table 4.4 Performance of the building blocks in secureSVM

Datasets	Total time	P time	C time	Comparison	SPM
BCWD	3195 s	2233 s	953 s	1769 s	3072 s
HDD	1935 s	1324 s	601 s	1050 s	1825 s

IoT data providers P and data analyst C. The results show that the building blocks proposed in this paper have acceptable time consumption.

As the performance results in Table 4.4, secureSVM has acceptable time consumption when training SVM classifiers which spends less than an hour with encrypted datasets BCWD and HDD. In our experiments, we are not doing parallel processing, where several P run in linear. In practice, P can be performed on parallel processing when interaction with C to improve efficiency. Table 4.4 exhibits the time accumulation spent by several P.

BCWD is a dataset with all numerical attributes and HDD is a dataset with all discrete attributes. Facing with the different types of dataset, secureSVM shows good robustness in terms of time consumption.

- Scalability Evaluation

In secureSVM, several IoT data providers participate in the data sharing and provide IoT data. In this subsection, we evaluate whether the change in the number of P has an effect on the time consumption. We divide the two datasets into several equal parts to simulate the scenarios that several IoT data providers participate in. We run secureSVM on the datasets and observe the time consumption changes with different numbers of P.

The number of IoT data providers is varied from 1 to 5, the results of which are shown in Figs. 4.6 and 4.7, where the x-coordinate represents the number of IoT data providers involved in the computation, and the y-coordinate shows the time consumption.

The time consumption of secureSVM is related to the amount of data and the number of iterations in the gradient descent. In theory, the total amount of data does not increase time consumption. With the same amount of data, the number of P does not affect the time consumption. Figures 4.6 and 4.7 confirm the intuition. Since when running the algorithm, other procedures in host used for the simulation disturb the algorithm at a certain extend, the total time consumption has a slight fluctuation,

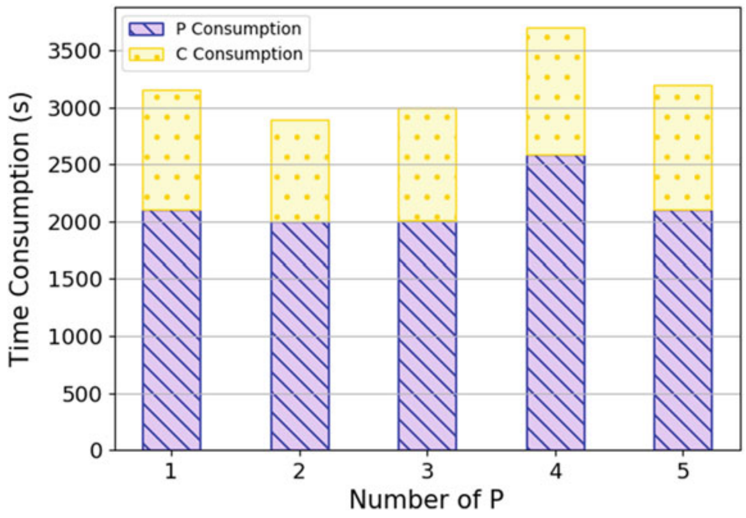


Fig. 4.6 Time consumption with different numbers of *P* on BCWD

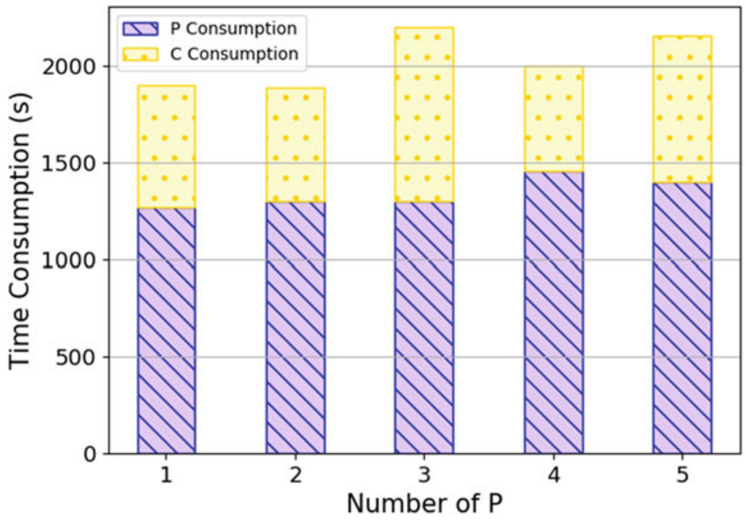


Fig. 4.7 Time consumption with different numbers of *P* on HDD

4.9 Summary

This section presents a novel privacy-preserving SVM training scheme, which tackled the challenges of data privacy and data integrity with blockchain techniques. Paillier is employed for designing the efficient and accurate privacy-preserving SVM training algorithm.

In homogeneous data sharing scenarios, multiple data owners have datasets with the same data format (or same data attributes) and they want to conduct ML modeling on their joint dataset. Labeled IoT data with the same data format from multiple entities contain users' sensitive data, which cannot be gathered directly for ML training. Most existing solutions suppose that training data can be reliably collected from multiple data providers, but reliably collected data is hard. To bridge the gap between ideal assumptions and realistic constraints, this chapter proposes secureSVM, which utilizes the blockchain techniques to build a secure and reliable data sharing platform among multiple data providers, where IoT data is encrypted and then recorded on a distributed ledger.

References

1. M. Abadi, A. Chu, I. Goodfellow, H.B. McMahan, I. Mironov, K. Talwar, L. Zhang, Deep learning with differential privacy, in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, CCS '16, New York, NY* (ACM, New York, 2016), pp. 308–318
2. Y. Aono, T. Hayashi, L. Trieu Phong, L. Wang, Scalable and secure logistic regression via homomorphic encryption, in *Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy, CODASPY '16, New York, NY* (ACM, New York, 2016), pp. 142–144
3. Y. Aono, T. Hayashi, L. Trieu Phong, L. Wang, Privacy-preserving logistic regression with distributed data sources via homomorphic encryption. IEICE Trans. Inf. Syst. **99**(8), 2079–2089 (2016)
4. A. Ben-David, N. Nisan, B. Pinkas, Fairplaymp: a system for secure multi-party computation, in *Proceedings of the 15th ACM Conference on Computer and Communications Security, CCS '08, New York, NY* (ACM, New York, 2008), pp. 257–266
5. A. Blum, C. Dwork, F. McSherry, K. Nissim, Practical privacy: The sulq framework, in *Proceedings of the Twenty-fourth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS '05, New York, NY* (ACM, New York, 2005), pp. 128–138
6. R. Bost, R.A. Popa, S. Tu, S. Goldwasser, Machine learning classification over encrypted data, in *Network and Distributed System Security Symposium* (2014)
7. R. Canetti, Security and composition of multiparty cryptographic protocols. J. Cryptol. **13**(1), 143–202 (2000)
8. M.d. Cock, R. Dowsley, A.C. Nascimento, S.C. Newman, Fast, privacy preserving linear regression over distributed datasets based on pre-distributed data, in *Proceedings of the 8th ACM Workshop on Artificial Intelligence and Security, AISec '15, New York, NY* (ACM, New York, 2015), pp. 3–14
9. C. Cortes, V. Vapnik, Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)
10. M. De Cock, R. Dowsley, C. Horst, R. Katti, A. Nascimento, W. Poon, S. Truex, Efficient and private scoring of decision trees, support vector machines and logistic regression models based on pre-computation. IEEE Trans. Depend. Sec. Comput. **16**(2), 217–230 (2019)
11. R. Detrano, A. Janosi, W. Steinbrunn, M. Pfisterer, J.-J. Schmid, S. Sandhu, K. H. Guppy, S. Lee, V. Froelicher, International application of a new probability algorithm for the diagnosis of coronary artery disease. Am. J. Cardiol. **64**(5), 304–310 (1989)
12. D. Dua, C. Graff, UCI machine learning repository (Irvine, CA, 2019). http://archive.ics.uci.edu/ml
13. C. Dwork, A firm foundation for private data analysis. Commun. ACM **54**(1), 86–95 (2011)

14. O. Goldreich, *Foundations of Cryptography: Volume 2, Basic Applications* (Cambridge University Press, Cambridge, 2009)
15. F.-J. González-Serrano, Á. Navia-Vázquez, A. Amor-Martín, Training support vector machines with privacy-protected data. Pattern Recogn. **72**, 93–107 (2017)
16. T. Graepel, K. Lauter, M. Naehrig, Ml confidential: Machine learning on encrypted data, in *Information Security and Cryptology – ICISC 2012*, ed. by T. Kwon, M.-K. Lee, D. Kwon (Springer, Berlin, 2013), pp. 1–21
17. J. Hsu, M. Gaboardi, A. Haeberlen, S. Khanna, A. Narayan, B.C. Pierce, A. Roth, Differential privacy: an economic method for choosing epsilon, in *Proceedings of the 2014 IEEE 27th Computer Security Foundations Symposium, CSF '14, Washington, DC* (IEEE Computer Society, Washington, 2014), pp. 398–410
18. H. Huang, B. Zhao, H. Zhao, Z. Zhuang, Z. Wang, X. Yao, X. Wang, H. Jin, X. Fu, A cross-platform consumer behavior analysis of large-scale mobile shopping data. In *Proceedings of the 2018 World Wide Web Conference, WWW '18, Republic and Canton of Geneva* (International World Wide Web Conferences Steering Committee, Geneva, 2018), pp. 1785–1794
19. J. Katz, Y. Lindell, *Introduction to Modern Cryptography*. CRC Cryptography and Network Security Series (CRC Press, Boca Raton, 2014)
20. J. Lee, C. Clifton, How much is enough? choosing ϵ for differential privacy, in *Proceedings of the 14th International Conference on Information Security, ISC'11* (Springer, Berlin, 2011), pp. 325–340
21. H. Li, L. Zhu, M. Shen, F. Gao, X. Tao, S. Liu, Blockchain-based data preservation system for medical data. J. Med. Syst. **42**(8), 141 (2018)
22. Y. Lindell, B. Pinkas, An efficient protocol for secure two-party computation in the presence of malicious adversaries, in *Proceedings of the 26th Annual International Conference on Advances in Cryptology, EUROCRYPT '07* (Springer, Berlin, 2007), pp. 52–78
23. X. Liu, R. Lu, J. Ma, L. Chen, B. Qin, Privacy-preserving patient-centric clinical decision support system on naive Bayesian classification. IEEE J. Biomed. Health Inform. **20**(2), 655–668 (2016)
24. F.D. McSherry, Privacy integrated queries: an extensible platform for privacy-preserving data analysis, in *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data, SIGMOD '09, New York, NY* (ACM, New York, 2009), pp. 19–30
25. M.A. Pathak, S. Rane, B. Raj, Multiparty differential privacy via aggregation of locally trained classifiers, in *Proceedings of the 23rd International Conference on Neural Information Processing Systems – Volume 2, NIPS'10* (Curran Associates Inc., Red Hook, 2010), pp. 1876–1884
26. J. Platt, Sequetial minimal optimization: a fast algorithm for training support vector machines, in *Technical Report MST-TR-98-14* (Microsoft Research, 1998)
27. Y. Rahulamathavan, R.C.W. Phan, S. Veluru, K. Cumanan, M. Rajarajan, Privacy-preserving multi-class support vector machine for outsourcing the data classification in cloud. IEEE Trans. Depend. Sec. Comput. **11**(5), 467–479 (2014)
28. M. Shen, Y. Deng, L. Zhu, X. Du, N. Guizani, Privacy-preserving image retrieval for medical IoT systems: A blockchain-based approach. IEEE Netw. **33**(5), 27–33 (2019)
29. M. Shen, J. Duan, L. Zhu, J. Zhang, X. Du, M. Guizani. Blockchain-based incentives for secure and collaborative data sharing in multiple clouds. IEEE J. Sel. Areas Commun. **38**(6), 1229–1241 (2020)
30. M. Shen, H. Liu, L. Zhu, K. Xu, H. Yu, X. Du, M. Guizani, Blockchain-assisted secure device authentication for cross-domain industrial IoT. IEEE J. Sel. Areas Commun. **38**(5), 942–954 (2020)
31. M. Shen, B. Ma, L. Zhu, R. Mijumbi, X. Du, J. Hu, Cloud-based approximate constrained shortest distance queries over encrypted graphs with privacy protection. IEEE Trans. Inf. Forensics Secur. **13**(4), 940–953 (2018)
32. M. Shen, B. Ma, L. Zhu, X. Du, K. Xu, Secure phrase search for intelligent processing of encrypted data in cloud-based IoT. IEEE Internet Things J. **6**, 1 (2019)

33. M. Shen, X. Tang, L. Zhu, X. Du, M. Guizani, Privacy-preserving support vector machine training over blockchain-based encrypted IoT data in smart cities. IEEE internet Things J. **6**(5), 7702–7712 (2019)
34. M. Shen, J. Zhang, L. Zhu, K. Xu, X. Tang, Secure SVM training over vertically-partitioned datasets using consortium blockchain for vehicular social networks. IEEE Trans. Veh. Technol. **69**(6), 5773–5783 (2020)
35. J. Soria-Comas, J. Domingo-Ferrer, D. Sánchez, S. Martínez, Enhancing data utility in differential privacy via microaggregation-based k-anonymity. VLDB J. **23**(5), 771–794 (2014)
36. M. Upmanyu, A.M. Namboodiri, K. Srinathan, C.V. Jawahar, Efficient privacy preserving k-means clustering, in *Intelligence and Security Informatics*, ed. by H. Chen, M. Chau, S.-h. Li, S. Urs, S. Srinivasa, G.A. Wang (Springer, Berlin, 2010), pp. 154–166
37. J. Vaidya, B. Shafiq, W. Fan, D. Mehmood, D. Lorenzi, A random decision tree framework for privacy-preserving data mining. IEEE Trans. Depend. Sec. Comput. **11**(5), 399–411 (2014)
38. H. Wang, F. Wang, J. Liu, C. Lin, K. Xu, C. Wang, Accelerating peer-to-peer file sharing with social relations. IEEE J. Sel. Areas Commun. **31**(9), 66–74 (2013)
39. Q. Wang, S. Hu, M. Du, J. Wang, K. Ren, Learning privately: privacy-preserving canonical correlation analysis for cross-media retrieval, in *IEEE INFOCOM 2017 – IEEE Conference on Computer Communications, May* (2017), pp. 1–9
40. K. Xu, H. Wang, J. Liu, S. Lin, L. Xu, Pushing server bandwidth consumption to the limit: modeling and analysis of peer-assisted VoD. IEEE Trans. Netw. Serv. Manag. **11**(4), 472–485 (2014)
41. A.C.C. Yao, How to generate and exchange secrets, in *27th Annual Symposium on Foundations of Computer Science (SFCS 1986), October* (1986), pp. 162–167
42. E. Yilmaz, H. Ferhatosmanoglu, E. Ayday, R.C. Aksoy, Privacy-preserving aggregate queries for optimal location selection. IEEE Trans. Depend. Sec. Comput. **16**(2), 329–343 (2019)
43. L. Yu, L. Liu, C. Pu, Dynamic differential location privacy with personalized error bounds, in *Network and Distributed System Security Symposium* (2017)
44. Y. Zhang, R. Yu, M. Nekovee, Y. Liu, S. Xie, S. Gjessing, Cognitive machine-to-machine communications: visions and potentials for the smart grid. IEEE Netw. **26**(3), 6–13 (2012)
45. T. Zhu, G. Li, W. Zhou, P.S. Yu, Differentially private data publishing and analysis: a survey. IEEE Trans. Knowl. Data Eng. **29**(8), 1619–1638 (2017)
46. L. Zhu, X. Tang, M. Shen, X. Du, M. Guizani, Privacy-preserving DDoS attack detection using cross-domain traffic in software defined networks. IEEE J. Sel. Areas Commun. **36**(3), 628–643 (2018)

Chapter 5
Secure Heterogeneous Data Sharing Using Blockchain

Big data play an extremely important role in different industries. Limited by the data sources between different departments or organizations, data collected by them differ greatly in attributes. These heterogeneous datasets form a complementary relationship with each other, so data sharing between organizations is necessary. Machine learning (ML) methods are widely used in big data processing, thus it is necessary to train machine learning models based on heterogeneous data. In the process of collaborative training machine learning based on heterogeneous data, the current scheme has many challenges, including efficiency, security, and availability in real situations. In this chapter, we propose a secure SVM training mechanism based on the consortium blockchain and threshold homomorphic encryption algorithm. By introducing the consortium blockchain, a decentralized data sharing platform can be constructed, and a secure SVM classifier algorithm can be built based on threshold homomorphic encryption. In order to ensure the efficiency of the model training process, most of the training work is performed locally based on plain text and only necessary shared data need to be encrypted.

5.1 Privacy Preservation of Heterogeneous Data

The development of cloud computing and edge computing has led to a proliferation of data, such as the large amount of data generated everyday in vehicular social networks, which can be used to optimize the security, convenience and entertainment of applications in vehicular social networks [2, 8, 27, 48]. Effective data analysis methods need to be used in such scenarios, among which machine learning and deep learning are particularly important [5, 13, 15, 44, 45]. Among the commonly used machine learning methods, support vector machine (SVM) model has significant advantages in performance and robustness, so it has a wide range of applications [22, 40, 43].

M. Shen et al., *Blockchain: Empowering Secure Data Sharing*,
https://doi.org/10.1007/978-981-15-5939-6_5

Take vehicular social network as an example. There are different entities in vehicle networks, such as vehicle manufacturer, vehicle management agency and vehicle social network application service provider. These entities have different data sources, and the differences in the data sources cause the data to complement each other in terms of attributes [1, 38]. We call the scenario as data heterogeneous data. However, for a single organization, its dataset cannot cover the multi-dimension, which has great limitations in the use process. Especially in the training process of SVM classifier, the classification effect of the final model is highly correlated with the quality of the data set, so it is difficult for a single organization to train an ideal classifier through its own data. Therefore, it is necessary to share heterogeneous data among multiple institutions. Through data sharing, a dataset covering multiple attributes can be combined to improve the effectiveness of the classifier. From another perspective, the dataset obtained after the fusion of these heterogeneous data can be vertically partitioned into sub-data sets provided by each unit according to the attributes.

However, in the process of data sharing, data privacy is facing serious challenges. First of all, the heterogeneous data to be shared contains users' privacy information. With the increasing attention of the government and individuals to users' privacy issues, more and more regulations restrict the sharing of users' data by enterprises. As a result, direct data sharing is subject to increasingly stringent regulations. In addition, for the data owners, the high value of heterogeneous data is mainly reflected in the privacy of the data, that is, the data is only owned by itself, or a small number of institutions. So if the data is shared directly, it becomes less private and less valuable and data owners are unwilling to reduce the value of their data [20].

For a long time, privacy disclosure issues raised in diverse scenarios has been highly concerned [19, 25, 29, 31, 33, 50]. Among those scenarios, many researches pay attention to train a machine learning classifier securely over both horizontally and vertically partitioned datasets. Many existing solutions adopt secure multi-party computation (SMC) to prevent privacy disclosure. Firstly, in those schemes, how to balance security and efficiency issues still faces big challenges [28, 29, 41, 50]. Then, one or more aided servers are essential with the assumption that they are trusted or semi-trusted during the training process. Obviously, in a real-world scenario, it is impractical to provide such aided servers for the participants. To deal with the two challenges of applying the privacy protection scheme to real-world scenarios, we propose an efficient and secure SVM classifier training scheme based on consortium blockchain where no third party is introduced [17, 24, 30, 36].

In this chapter, we propose a security SVM training mechanism based on consortium blockchain for multi-source heterogeneous data sharing scenario, which solves the above two problems. First, because the differential privacy protection scheme introduces noise to the training results and the training process is not secure, we adopt the scheme based on homomorphic encryption [16]. Different from other privacy training schemes based on homomorphic encryption, in our scheme, datasets do not need to be shared, and all the training work is based on plain text datasets, avoiding the need for training based on private datasets in traditional schemes, so the efficiency is greatly improved. We use homomorphic encryption to protect the necessary data that needs to be shared between multiple participants. In order to further reduce the data sharing frequency in the training process of SVM model, we adopted the training algorithm based on stochastic gradient descent. In one iteration, each participant only needs to share data twice.The result is a balance between security and efficiency [7, 18, 26, 34, 42].

Secondly, in order to improve the applicability of the scheme in real scenarios, we avoid introducing trusted third parties and reduce data leakage caused by trusted third parties. Through the threshold homomorphic encryption scheme, the decryption of the secret data to be protected needs to be jointly decrypted by participants exceeding the threshold value [6]. No organization can infer the clear text data from the secret data. At the same time, if the traditional P2P data transmission scheme is adopted, multiple participants need to establish connections, maintain connections and transmit data with all other participants in the process of data sharing.Such an approach is not desirable in terms of efficiency and safety. We introduce block chain to establish a decentralized data sharing platform for sharing secret data. When each participant shares data, they simply upload the data to the data sharing platform. The access control and permission mechanism of the consortium blockchain fully ensures the unknowability of the external data and the openness and transparency of the internal data [14, 32, 35, 49, 51].

In this chapter, we propose a SVM training scheme that contributes more secure and efficient heterogeneous data sharing. First, an open, reliable and transparent data sharing platform was built based on blockchain technology. The operation of the platform does not rely on trusted third parties. The data on the platform is visible to members in the blockchain and not visible to the outside. After that, most of the training work was completed locally by each participant based on clear text data. We introduce threshold homomorphic encryption scheme to ensure a data privacy protection scheme in a decentralized environment. All data that needs to be shared can be fully protected by this scheme and maintain its homomorphic property. Our scheme guarantees a controllable degree of privacy protection by setting the size of the threshold. A large number of experiments based on real datasets prove the feasibility and efficiency of the scheme.

5.2 Secure Machine Learning over Heterogeneous Data

Existing researches on privacy preserving machine learning are involved in many ML methods including traditional methods such as linear regression [3], support vector machine [39, 46], naive Bayes classifier [21], logic regression [12, 47] and so on. Deep learning [37] is also focused in the last few years.

Many researches preserve the privacy of machine learning process with the help of one or more third party [9, 11, 24–26]. It is a hard work to find such entity to serve as a third party, thus these methods are unpractical when applied to real-world in this aspect.

Table 5.1 Notations

Notations	Description
D^A	The dataset of participant A
d^A	The dimension of dataset D^A
x_i^A	The i-th data instance of dataset D
y_i	Number

5.3 Preliminaries

Consider a dataset D is combined with several participants who have its own dataset D^p $p \in A, B, \dots N$, where x_i^p represent the i-th instance in D^p, and y_i is shared as a data label between all related i-th instance x_i^X. When training a SVM classifier, we define w as the model parameters, Δ_t as gradient in the t iteration, λ as the learning rate. Meanwhile, we assume that $[[m]]$ as the encryption of message m under Paillier. Table 5.1 shows the notations used in this paper.

5.4 System Model

We divide our system into three components based on their relationship with the data. As shown in Fig. 5.1, they are data device (DD), data provider (DP) and blockchain service platform (BSP).

- Data Device: Refer to devices capable of generating data, including sensors, mobile devices, and so on. Because the data directly collected from these devices contains high-value information, these data are collected, processed, and then used for data analysis.

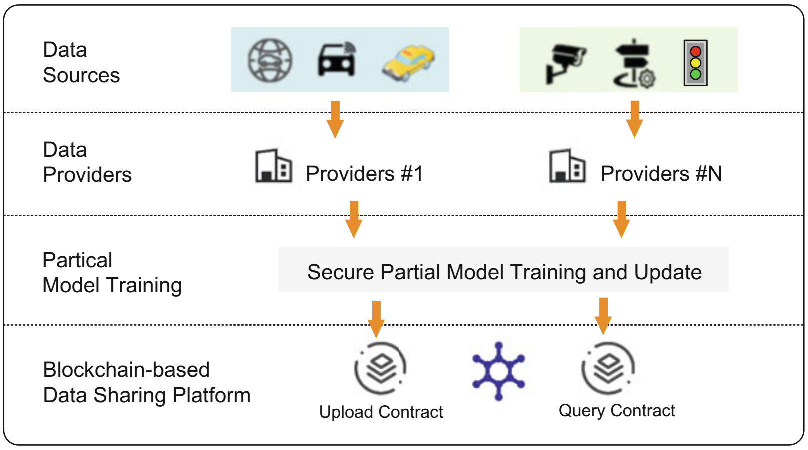


Fig. 5.1 Overview of secure SVM training scheme over heterogeneous data

- Data Provider: The equipment that generates the data is collected, stored, and used by different parties. These participants are called participants and act as data providers in our solution. Due to the different equipment, the collected data is different, and due to the different data processing methods, the available data after processing has different attributes and complement each other. In addition to serving as data providers, these participants also act as model trainers to train machine learning models in collaboration. According to the scheme in this paper, most of the training work is done locally in participant.
- Blockchain Service Platform: This is a service platform that runs on the consortium blockchain. On one hand, it provides a transparent data sharing platform distributed in participant, allowing participant to retrieve all the data recorded in the BSP. At the same time, no one captured the data recorded on the BSP for changes. On the other hand, BSP has strong security protections, making data outside of participant invisible to entities. In addition, communication data between the BSP and participant is also encrypted, preventing data leakage.

5.4.1 Threat Model

In the scheme, there is only one role of the data provider. We treat participants honest but curious when it comes to the security model, that is, all participants are curious about the data of other participants, but they will execute the scheme according to the rules. In addition, due to the large number of interactions between participant and BSP, potential threats in the interaction process are also considered.

- Known Ciphertext Model. BSP is a common and transparent data sharing platform for all participants. The data shared by each participant is visible to other participants. These data include the dense intermediate value and the decrypted calculation results.
- Known Background Model. We assume that multiple participants can conspire and collaborate to analyze shared data. Compared with the above threat model, this model can obtain more information.

5.4.2 Design Goals

Under the above system model and threat model, we established the following three system design goals to meet the system's requirements for security, accuracy and performance.

- Data privacy is fully protected. Under the two threat models, during the entire training process, the privacy of the original data and the shared intermediate value will not be leaked, and the participants cannot infer valuable information from the shared data. Second, the data in the data sharing platform is guaranteed to be invisible to the outside world.

- High accuracy of training results. Generally speaking, the introduction of privacy protection schemes may introduce noise into the calculation process and cause inaccurate calculation results. Our design goal is to obtain a classifier that is not significantly different from conventional training conditions.
- Low training overhead. Similarly, the introduction of privacy protection schemes will increase training overhead. These overheads are mainly caused by additional computing operations such as encryption and decryption, and additional communication overhead. Therefore, our solution needs to ensure low training overhead while ensuring security.

5.5 Secure SVM Training Scheme over Heterogeneous Datasets

In this section, in order to clearly introduce the work of each participant in the training process, we assume that three participants participate in the SVM model training. The respective training sets are complementary in attributes. As shown in Fig. 5.2, the entire training process mainly includes three parts: local training, gradient update judgment, and model update. In these three steps, two data sharing and one decryption operation are involved. Finally, after multiple iterations, each participant gets its own partial model and uploads it to the blockchain to form a complete model together.

5.5.1 System Initialization

The data privacy protection method of this solution is based on a threshold homomorphic encryption algorithm. Before training the model, a pair of public and private keys needs to be generated for each participant. The public key is the same and the private key is different. Through the secret sharing scheme combined with the existing threshold key management scheme, such a key pair is negotiated and distributed. In addition, the three participants join the consortium blockchain data sharing platform as nodes, and they need to pass identity authentication before joining. Finally, all participants need to initialize the model parameters and preprocess the data set, including unified labeling and sample order.

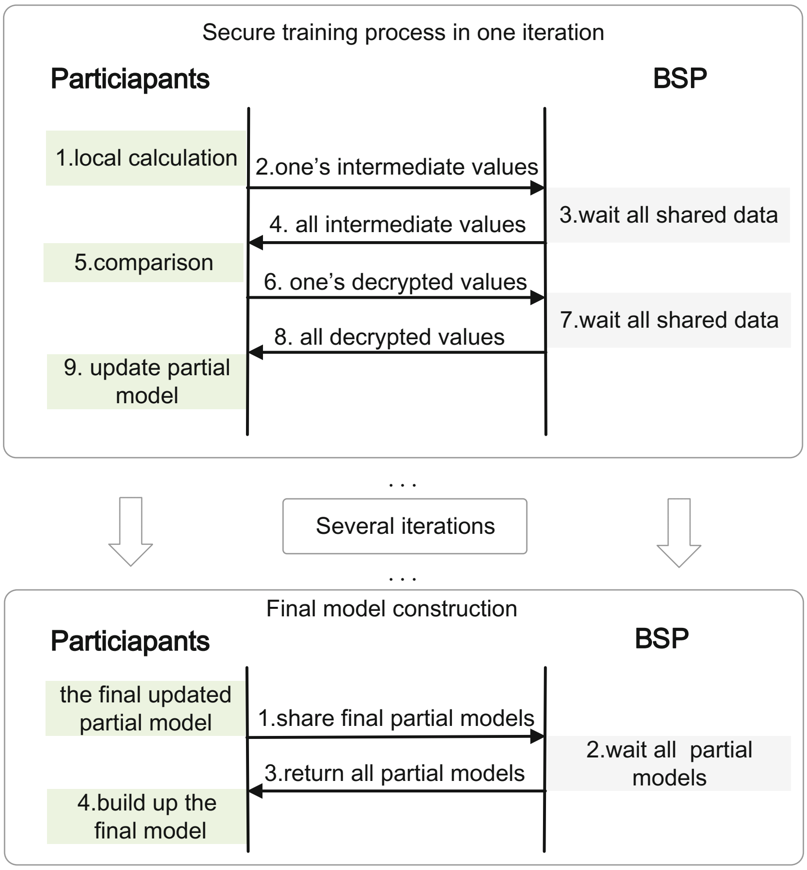

Secure training process in one iteration
Particiapants
BSP
1.local calculation
2.one's intermediate values
3.wait all shared data
4. all intermediate values
5.comparison
6. one's decrypted values
7.wait all shared data
8. all decrypted values
9. update partial model
Several iterations
Final model construction
Particiapants
BSP
the final updated partial model
1.share final partial models
2.wait all partial models
3.return all partial models
4.build up the final model

Fig. 5.2 Workflow of secure training over heterogeneous datasets

5.5.2 *Local Training Process*

In order to ensure the efficient training of the model, this solution puts most of the work locally on three participants. During one iteration, all training work can be done locally before the gradient update judgment. This section will introduce how each participant can be trained locally based on its own heterogeneous data.

SVM optimization algorithm based on stochastic gradient descent (SGD) is easy to perform. SVM based on stochastic gradient descent can be expressed in the following form:

$$f(w) = \frac{1}{2}w^T w + C\sum_{i=1}^{m} max(0, 1 - y_i w^T x_i) \tag{5.1}$$

```
Algorithm 1 SVM based on SGD
Require: Training set $D$, learning rate $\lambda$, maxIters $T$.
Ensure: Trained model $w^*$.
1: for $t = 1$ to $T$ do
2:     Select $i_t$ from $D$ randomly.
3:     Update $\Delta_{t+1}$ by Eq. (5.2).
4:     Update $w_{t+1}$ by Eq. (5.3).
5: end for
6: return $w^*$.
```

The right part of the equation is the hinge-loss function, where C is the misclassification penalty and we take $\frac{1}{m}$ as its value.

At each iteration, we use Eq. (5.2) to calculate the gradient.

$$\Delta_t = \lambda w_t - I\left[(wx_i < 1)\right] x_i y_i \tag{5.2}$$

If $I\left|(wx_i < 1)\right|$ is true which means $(wx_i < 1)$, $I\left[wx_i < 1)\right] = 1$; Otherwise, $I\left[(wx_i < 1)\right] = 0$.

Then we can update the w by Eq. (5.3).

$$w_{t+1} = w_t - \lambda \Delta_t \tag{5.3}$$

Through one iteration of the training process over several heterogeneous datasets, only when calculating I, data exchange between multiple participants is required. The rest of the training operations are performed locally. We represent wx_i by a in the following sections.

$$I = \begin{cases} 1 & y_i\left(w^A x_i^A + w^B x_i^B + w^C x_i^C\right) < 1 \\ 0 & otherwise \end{cases} \tag{5.4}$$

The compete algorithm is described in Algorithm 2.

5.5.3 *Privacy-Preserving Gradient Update Judge*

The three participants need to share the calculated median value to the BSP during the training model. This solution treats the shared data with a threshold homomorphic encryption scheme to ensure data security and ensure that the gradient can be calculated correctly. To judge how to update the gradients, here we use additive homomorphic encryption to construct Eqs. (5.5), (5.6) and (5.7).

Algorithm 2 Partial model training process

```
Require: Training set $D^A$, $D^B$, $D^C$, learning rate $\lambda$, maxIters $T$.
Ensure: Trained model $w^*$.
 1: All participants perform the following operations simultaneously. Take participant $A$ to describe in detail.
 2: for $t = 1$ to $T$ do
 3:     Select $i_t$ randomly.
 4:     Calculate $y \sum_{i=1}^{d^A} w_i x_i$.
 5:     Cooperate with other participants to judge how to update gradient by Eq. (5.5).
 6:     Update $\Delta_{t+1}$ by Eq. (5.2).
 7:     Update $w_{t+1}$ by Eq. (5.3).
 8: end for
 9: Get several partial model parameters and combine them.
10: return $w^*$.
```

$$[[a]] = [[\sum_{i=1}^{n} a^i]] = \prod_{i=1}^{n} [[a^i]] \tag{5.5}$$

$$[[r_2]] = [[\sum_{i=1}^{n} r_2^i]] = \prod_{i=1}^{n} [[r_2^i]] \tag{5.6}$$

$$\begin{aligned} [[ar_1 + r_2]] = [[ar_1]][[r_2]] &= [[\sum_{i=1}^{r_1} a]][[r_2]] \\ &= \prod_{i=1}^{r_1} [[a]][[r_2]] = [[a]]^{r_1}[[r_2]] \end{aligned} \tag{5.7}$$

In order to determine the update method of the gradient, the method adopted in this solution is to compare the encrypted calculation result with the constant 1. In Algorithm 3, the security comparison algorithm in three participant scenarios is introduced in detail.

It is obvious that for integer a, if $(ar_1{+}r_2) > (r_1 + r_3)$, we can derive that $a > 1$, otherwise $a{<}1$.

Algorithm 3 Privacy-preserving gradient update judge

```
Input A: $[[a^i]]$ from participant $i$.
Input B: $r_1^i$, $[[r_2^i]]$, $r_3^i$ from participant $i$.
Ensure: $a>1$ or $a<1$.
1: Each participant $i$ picks three positive integers $r_1^i$, $r_2^i$, $r_3^i$, where $\left|r_3^i - r_2^i\right| < r_1^i$, and encrypts $r_2^i$ to get $[[r_2^i]]$.
2: Each participant $i$ uploads $[[a^i]]$, $r_1^i$, $[[r_2^i]]$, $r_3^i$.
3: Each participant $i$ downloads all the other participants' $[[a^i]]$, $r_1^i$, $[[r_2^i]]$, $r_3^i$.
4: Each participant $i$ calculates $[[a]]$, $[[r_2]]$ by Eq. (5.5) and Eq. (5.6), and calculates $r_1$ and $r_3$ where $r_1 = \sum_{i=1}^n r_1^i$ and $r_3 = \sum_{i=1}^n r_3^i$.
5: Each participant $i$ calculates $[[ar_1 + r_2]]$ by Eq. (5.7).
6: Each participant $i$ decrypts $[[ar_1 + r_2]]$ by sub-private key $SK^i$ and uploads it to BSP.
7: Each participant $i$ downloads all other decrypted values from participants to recover $(ar_1 + r_2)$, and compares $(ar_1 + r_2)$ with $(r_1 + r_3)$.
8: If $(ar_1 + r_2)>(r_1 + r_3)$, $a>1$; Else $a<1$.
9: return $a>1$ or $a<1$.
```

5.5.4 Data Sharing on BSP

Participant relies on BSPs to securely calculate intermediate values. BSP simplifies complex point-to-point communication between participants. Participant completes data on-chain and data query by calling smart contracts. During the iteration process, each participant uploads data twice: calculating the intermediate value (IV) and the decrypted value (DV), respectively. These two data are also read twice.

1. The Format of IVs

   Iteration Round: When multiple data providers train the model collaboratively, some data needs to be exchanged in each iteration. Therefore, in order to represent the data exchanged in each round and to distinguish it from other rounds of data, a field is required to indicate the training round. *Iteration Round* is maintained by smart contracts.

   DP ID: A field that identifies the owner of the data. When a node calls a contract to upload data, its address will be automatically recorded in this field.

   Training Intermediate Value: The intermediate value of the encrypted state during model training. The values provided by each participant will be summed and compared to the magnitude of 1 in the encrypted state.

   r_1*:* An unencrypted random positive integer which is used to compare.

   r_2*:* An encrypted random positive integer which is used to compare.

Table 5.2 Attributes of IV

Attributes	Details
IR	Current training round
DP ID	A unique identifier of a DP
TIV	It is encrypted by DP
r_1	A random positive integer (unencrypted)
r_2	A random positive integer (encrypted)
r_3	A random positive integer (unencrypted)
RPI	It is generated randomly between 1 and m

Table 5.3 Attributes of DV

Attributes	Details
IR	Current training round
DP ID	A unique identifier of a DP
DV	Partial decrypted values from each participant

r_3: An unencrypted random positive integer which is used to compare.

Random Positive Integer: It is generated randomly by each participant and its value is between 1 and m, the sum of which determines the data instances selected in the next iteration (Table 5.2).

2. The Format of DVs

   Iteration Round: Similar function described in IVs.

   DP ID: Similar function described in IVs.

   Decrypted Value: Each participant decrypts the result obtained based on his own private key. By combining all these values, each participant can obtain the final decryption result (Table 5.3).

5.6 Security Analysis

Goldreich described in detail the current widely accepted security definition in the paper [10], which builds an ideal model by establishing a trusted third party T to establish secure communication with other participants. Since the ideal model is in an absolutely secure state, if the model constructed under the real protocol is not different from the ideal model, then we consider the real protocol to be secure. An attacker in an ideal model can perform the same attack as in a real model.

The definition of computing security for a secure multiparty computing protocol is given below.

Definition 5.1 The multi-party computation protocol with n participants under the cryptography model is considered to be computation security, if for any attacker A, there exists a corresponding simulator S in the ideal model interacting with A, and satisfying the following conditions:

(1) The running time of S is the polynomial of A's running time.
(2) For any input set, the n+1 outputs produced by the multi-party computation protocol are computationally indistinguishable from the n+1 outputs produced by the ideal model.

We conducted a security analysis based on the above idea. Thus we acquire the information which an attacker can get from the ideal model and the real protocol. Then we compare them and prove they are indistinguishable.

In our scheme, n participants are involved to share their encrypted intermediate values to calculate F: $F([[a]]^1, \dots, [[a]]^n, 1, r_1^1, [[r_2^1]], r_3^1, \dots, r_1^n, [[r_2^n]], r_3^n)$. Assume that the attacker has corrupted a set of participants $A = P_{i1}, \dots, P_{i|A|}$. Then all the data the attacker obtained in the ideal model is the output F of the participants and the input:

$$\left([[a]]^{i1}, \dots, [[a]]^{i|A|}, 1, r_1^{i1}, [[r_2^{i1}]], r_3^{i1}, \dots, r_1^{i|A|}, [[r_2^{i|A|}]], r_3^{i|A|}\right).$$

We construct a simulator S that simulates all the data the attacker gets in the real model based on the data the attacker obtained in the ideal model. Firstly, we analyze the information that the attacker can get in the real protocol.

Input Phase Since all the participants share their encrypted input:

$$\left([[a]]^1, \dots, [[a]]^n, 1, r_1^1, [[r_2^1]], r_3^1, \dots, r_1^n, [[r_2^n]], r_3^n\right)$$

The attacker is able to get all of them. Especially for the corrupted participants, the attacker also gets $a^{i1}, \dots, a^{i|A|}, r_2^1, \dots, r_2^{|A|}$.

Computation Phase At each step of the calculation phase, the attacker obtains data $[[x + y]]$ based on $[[x]]$ and $[[y]]$.

Output Phase In the output phase, the attacker gets the result:

$$F([[a]]^1, \dots, [[a]]^n, 1, r_1^1, [[r_2^1]], r_3^1, \dots, r_1^n, [[r_2^n]], r_3^n).$$

Then we construct the simulator S of the polynomial time. S takes $a^{i1}, \dots, a^{i|A|}$ and $r_2^1, \dots, r_2^n$ and F as the input. The following step S0 simulates the information calculated based on the input.

Step S0 S generates the encrypted data $[[a^{i1}]], \dots, [[a^{i|A|}]], [[r_2^{i1}]], \dots, [[r_2^{i|A|}]]$ and $[[F]]$ based on $a^{i1}, \dots, a^{i|A|}, r_2^1, \dots, r_2^n$ and F. Then, S can simulate the calculations based on those encrypted data such as $[[a^{i1} + a^{i2}]]$ and $[[r_2^{i1} + r_2^{i2}]]$.

Step S1 After step S0, we can simulate part of the calculated intermediate values which are defined as $[[a^{j1}]], \dots, [[a^{j|r|}]], [[r_2^{j1}]], \dots, [[r_2^{jr}]]$. Then for the remaining intermediate values that cannot be directly simulated by $S0$, S simulates them by selecting the random numbers to generate the corresponding ciphertext. According to the threshold cryptosystem's security, these simulations are successful.

Step S2 Based on steps S0 and S1, we can simulate all the values calculated in the computation phase.

Step S3 F is one of S's input, so S can easily get a simulation of F.

From the above simulation process, the information obtained by the attacker from the ideal model and the information obtained from the real model are computationally indistinguishable. You can prove the security of the solution.

5.7 Experiments

In this section, we carry out a large number of experiments to verify the training of our scheme on different datasets. We will evaluate the results from the aspects of accuracy, training efficiency and scalability. At the same time, a certain theoretical analysis has been carried out on the evaluation results.

5.7.1 Experimental Settings

We simulate several participants in our experiments using our secure SVM algorithm to train the model. First, the model training program and data upload program are implemented by java and go respectively. Finally, the Java program was runned on a PC (3.60 GHz AMD Ryzen 5 2600X six-core processor and 32 GB RAM) to simulate the entire training process. The Go program runs on the consortium blockchain (structure 1.3) in the form of consortium code in a virtual machine with 4 GB of memory.

Because the operations in the threshold Paillier scheme are in integer space, before model training, we first map floating point numbers in the data set to integer space. By using integers to represent floating-point numbers, the problem of unavoidable floating-point operations during SVM training is solved. According to the international standard IEEE 754, any binary floating point number D can be expressed as $D = (-1)^s \times M \times 2^E$, s represents a sign bit, M represents a significant number, and E represents an exponent bit.Before model training, we also need to set the parameters related to the threshold Paillier, including the key N is set to 256 bits, and the threshold t is set to the number of participants. To evaluate the performance of our solution on a real dataset, we used a dataset under real conditions. To evaluate the performance of our scheme, we use the real-world datasets Breast Cancer Wisconsin Data Set (BCWD) [23] and Australian Credit Approval Data Set (ACAD) [4]. The detail information about the two datasets is list in Table 5.4. For these two data sets, we divide them vertically into three heterogeneous sub-data sets

Table 5.4 Statistics of datasets

Datasets name	Instances number	Attributes number
BCWD	699	9
ACAD	690	14

according to their attributes. Each data set has an average number of attribute values and is retained by each participant.

The model training parameters are set as follows: the maximum number of iterations for model training is 1500, the learning rate for model training is 0.00095, and the number of samples selected from the training data set per iteration is 1.

5.7.2 *Accuracy*

In terms of model training accuracy, we use two criteria: accuracy P and recall R to evaluate the proposed secure SVM training scheme. Where P is calculated by the formula $P = t_p/(f_p + t_p)$ and is calculated by the equation $R = t_p/(f_n + t_p)$, T_p in the formula represents the number of positive examples of correct classification, f_p represents the number of negative examples of correct classification, and f_n is a positive example of incorrect number classification.

Through this experiment, we plan to prove that after introducing the privacy protection scheme, the accuracy of the classifier trained by our proposed SVM training scheme will not be significantly reduced. To prove this inference, we have done it through a comparative experiment. The classification results of the SVM classifier (PP-SVM) under our proposed security scheme are compared with the conventional SVM classifier (SVM) without privacy protection (Table 5.5).

Through the experimental results, we find that compared with the SVM classifiers trained under normal conditions, there is no obvious loss of accuracy in the accuracy of the classifiers obtained by our proposed secure SVM method. This result validates our inference. In theory, because the threshold Paillier scheme does not cause loss of accuracy to the accuracy of the calculation, it finally guarantees a high accuracy of the classifier.

Table 5.5 Performance of classifier accuracy

	Precision		Recall	
Dataset	PP-SVM	SVM	PP-SVM	SVM
BCWD	0.9160%	0.9122%	0.9958%	0.9958%
ACAD	0.8870%	0.9176%	0.8198%	0.8146%

5.7.3 *Efficiency*

In this section, we will evaluate the efficiency of our proposed scheme from time overhead and scalability.

After theoretical analysis and experimental verification, the total training time in our scheme is mainly composed of two parts: calculation time overhead and communication time overhead. The calculation time overhead is mainly composed of local training time (LTT) and gradient update judgment time (GUJT). At the same time, the communication time overhead mainly consists of calling smart contracts to upload and query data time. In order to obtain the total time cost and composition during the training process, we conducted related experiments.

In the experiment, we simulated three participants and trained part of the model on their own data set during each round of iteration. In this process, there is no encryption and decryption operation, this time-consuming part is named LTT. Secondly, in order to update the gradient and parameters of the model, each round requires a secure gradient update operation. In this process, encryption and decryption operations are designed to eliminate communication time. This part of the time is called GUJT.

The detailed time cost information is shown in Fig. 5.3. The results of the time cost statistics are shown in Tables 5.6, 5.7, and 5.8. Experimental results show that model training takes very little time. And because a large number of encryption and decryption operations are avoided, the communication time in the total time accounts for a large proportion, and the calculation time overhead is not large.

Compared with the calculation overhead, the communication overhead is too high, but the consortium blockchain as a data sharing platform requires consensus

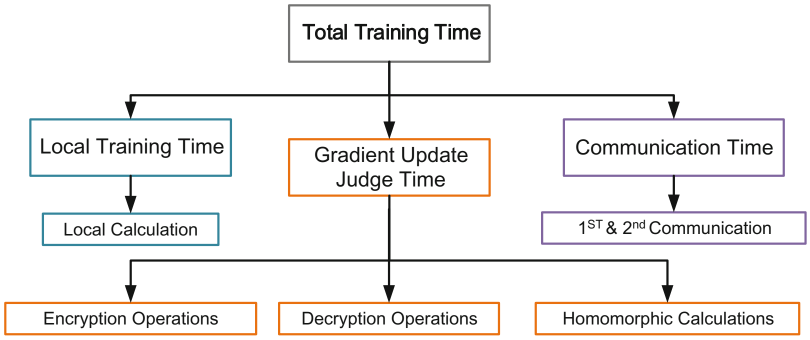


Fig. 5.3 Time calculation

Table 5.6 Performance of classifier efficiency

	Total time overhead
Dataset	Total time
BCWD	384,649 ms
ACAD	393,261 ms

Table 5.7 Performance of calculation time

	Calculation time overhead	
Dataset	Local training time	Gradient update judge time
BCWD	65 ms	8064 ms
ACAD	58 ms	8247 ms

Table 5.8 Performance of communication time

	Communication time overhead	
Dataset	First communication time	Second communication time
BCWD	186,034 ms	190,486 ms
ACAD	189,332 ms	195,624 ms

between the nodes, so such time overhead cannot be avoided, which is a sacrifice of security.

5.7.4 Scalability Evaluation

As the number of participants increases, more experiments will be performed to evaluate the scalability of the scheme. In these experiments, the dataset is split vertically into three to five parts, and each participant holds one of them. The results are shown in Figs. 5.4 and 5.5.

From Fig. 5.4, we can conclude that the number of participants does not affect the classifier accuracy. However, Fig. 5.5 shows that increasing the number of participants has a negative impact on computational time overhead. Theoretically, experiments performed with four participants take longer to encrypt and decrypt in the gradient update decision process, compared to experiments performed with three participants. However, the total communication time has not increased significantly.

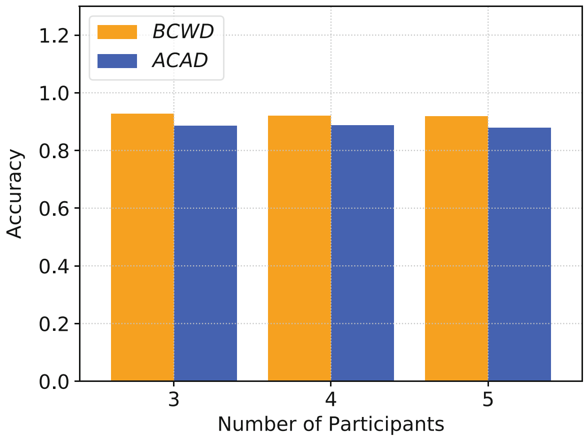


Fig. 5.4 Accuracy with different numbers of participants

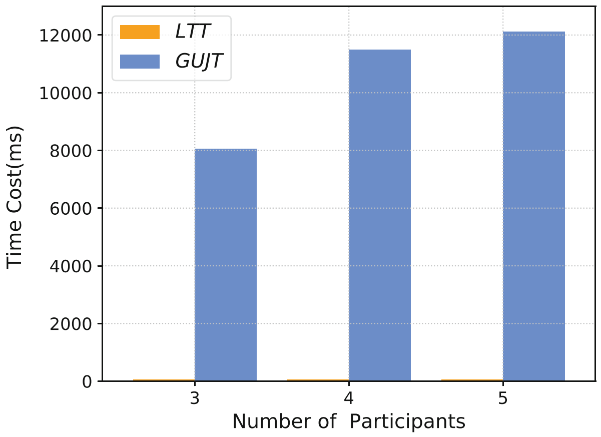


Fig. 5.5 Time consumption of dataset BCWD with different numbers of participants

The way to define airtime in a round of iteration is the time interval between when data is first uploaded to the BSP and when participant last queries the data. On the other hand, each participant performs almost the same amount of work before uploading the data, so all participants upload data and receive the returned data almost simultaneously. Increasing the number of participants from 3 to 5 does not change the communication time much. Communication time mainly depends on the number of iterations fixed at 1500 in all experiments.

5.8 Summary

In this section, we propose an effective and secure SVM training scheme that helps multiple data providers train SVM classifiers on vertically partitioned datasets. The target of this chapter is to combine consortium blockchain technology and threshold Paillier to create a decentralized and secure SVM training platform. To achieve high performance, most training operations are performed locally on raw data, so there are only a few intermediate values that need to be shared across platforms. Extensive experiments have been performed, and the results show that this scheme can train accurate SVM classifiers at lower time cost.

References

1. R. Bost, R.A. Popa, S. Tu, S. Goldwasser, Machine learning classification over encrypted data, in *NDSS*, vol. 4324 (2015), pp. 4325
2. N. Cheng, F. Lyu, J. Chen, W. Xu, H. Zhou, S. Zhang, X.S. Shen, Big data driven vehicular networks. IEEE Netw. **32**(6), 160–167 (2018)

3. M.d. Cock, R. Dowsley, A.C. Nascimento, S.C. Newman, Fast, privacy preserving linear regression over distributed datasets based on pre-distributed data, in *Proceedings of the 8th ACM Workshop on Artificial Intelligence and Security*, pp. 3–14 (ACM, New York, 2015)
4. D. Dua, C. Graff, UCI machine learning repository (2017). http://archive.ics.uci.edu/ml
5. Z.M. Fadlullah, F. Tang, B. Mao, N. Kato, O. Akashi, T. Inoue, K. Mizutani, State-of-the-art deep learning: evolving machine intelligence toward tomorrow's intelligent network traffic control systems. IEEE Commun. Surv. Tutorials **19**(4), 2432–2455 (2017). Fourthquarter
6. P.-A. Fouque, G. Poupard, J. Stern, Sharing decryption in the context of voting or lotteries, in *International Conference on Financial Cryptography* (Springer, Berlin, 2000), pp. 90–104
7. K. Gai, Y. Wu, L. Zhu, M. Qiu, M. Shen, Privacy-preserving energy trading using consortium blockchain in smart grid. IEEE Trans. Ind. Inf. **15**(6), 3548–3558 (2019)
8. F. Gao, L. Zhu, M. Shen, K. Sharif, Z. Wan, K. Ren, A blockchain-based privacy-preserving payment mechanism for vehicle-to-grid networks. IEEE Netw. **32**(6), 184–192 (2018)
9. A. Gascón, P. Schoppmann, B. Balle, M. Raykova, J. Doerner, S. Zahur, D. Evans, Secure linear regression on vertically partitioned datasets. IACR Cryptol. **2016**, 892 (2016). ePrint Archive
10. O. Goldreich, *Foundations of Cryptography: Basic Tools* (Cambridge University Press, Cambridge, 2001)
11. F.-J. González-Serrano, Á. Navia-Vázquez, A. Amor-Martín, Training support vector machines with privacy-protected data. Pattern Recogn. **72**, 93–107 (2017)
12. T. Graepel, K. Lauter, M. Naehrig, Ml confidential: machine learning on encrypted data, in *Information Security and Cryptology – ICISC 2012*, ed. by T. Kwon, M.-K. Lee, D. Kwon (Springer, Berlin, 2013), pp. 1–21
13. M. Hao, H. Li, X. Luo, G. Xu, H. Yang, S. Liu, Efficient and privacy-enhanced federated learning for industrial artificial intelligence [J]. IEEE Trans. Ind. Inf. (2019)
14. W. Jiang, H. Li, G. Xu, M. Wen, G. Dong, X. Lin, PTAS: privacy-preserving thin-client authentication scheme in blockchain-based PKI. Future Gener. Comput. Syst. **96**, 185–195 (2019)
15. N. Kato, Z.M. Fadlullah, B. Mao, F. Tang, O. Akashi, T. Inoue, K. Mizutani, The deep learning vision for heterogeneous network traffic control: proposal, challenges, and future perspective. IEEE Wirel. Commun. **24**(3), 146–153 (2017)
16. J. Katz, Y. Lindell, *Introduction to Modern Cryptography* (Chapman and Hall/CRC, Boca Raton, 2014)
17. H. Li, Y. Yang, Y. Dai, S. Yu, Y. Xiang, Achieving secure and efficient dynamic searchable symmetric encryption over medical cloud data. IEEE Trans. Cloud Comput. **8**(2), 484–494 (2020)
18. Y. Li, L. Zhu, M. Shen, F. Gao, B. Zheng, X. Du, S. Liu, S. Yin, Cloudshare: towards a cost-efficient and privacy-preserving alliance cloud using permissioned blockchains, in *International Conference on Mobile Networks and Management* (Springer, Cham, 2017), pp. 339–352
19. H. Li, D. Liu, Y. Dai, T. H. Luan, S. Yu, Personalized search over encrypted data with efficient and secure updates in mobile clouds. IEEE Trans. Emerg. Top. Comput. **6**(1), 97–109 (2018)
20. H. Li, L. Zhu, M. Shen, F. Gao, X. Tao, S. Liu, Blockchain-based data preservation system for medical data. J. Med. Syst. **42**(8), 141 (2018)
21. X. Liu, R. Lu, J. Ma, L. Chen, B. Qin, Privacy-preserving patient-centric clinical decision support system on naive Bayesian classification. IEEE J. Biomed. Health Inf. **20**(2), 655–668 (2016)
22. L. Lv, Y. Zhang, Y. Li, K. Xu, D. Wang, W. Wang, M. Li, X. Cao, Q. Liang, Communication-aware container placement and reassignment in large-scale Internet data centers. IEEE J. Sel. Areas Commun. **37**(3), 540–555 (2019)
23. O.L. Mangasarian, W.H. Wolberg, Cancer diagnosis via linear programming. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 1990

24. P. Mohassel, P. Rindal, ABY 3: a mixed protocol framework for machine learning, in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 35–52 (ACM, New York, 2018)
25. P. Mohassel, Y. Zhang, Secureml: a system for scalable privacy-preserving machine learning, in *2017 IEEE Symposium on Security and Privacy (SP)* (IEEE, Piscataway, 2017), pp. 19–38
26. V. Nikolaenko, U. Weinsberg, S. Ioannidis, M. Joye, D. Boneh, N. Taft, Privacy-preserving ridge regression on hundreds of millions of records, in *2013 IEEE Symposium on Security and Privacy* (IEEE, Piscataway, 2013), pp. 334–348
27. Z. Ning, F. Xia, N. Ullah, X. Kong, X. Hu, Vehicular social networks: enabling smart mobility. IEEE Commun. Mag. **55**(5), 16–55 (2017)
28. H. Ren, H. Li, Y. Dai, K. Yang, X. Lin, Querying in internet of things with privacy preserving: challenges, solutions and opportunities. IEEE Netw. **32**(6), 144–151 (2018)
29. M. Shen, B. Ma, L. Zhu, R. Mijumbi, X. Du, J. Hu, Cloud-based approximate constrained shortest distance queries over encrypted graphs with privacy protection. IEEE Trans. Inf. Forensics Secur. **13**(4), 940–953 (2018)
30. M. Shen, M. Wei, L. Zhu, M. Wang, Classification of encrypted traffic with second-order Markov chains and application attribute bigrams. IEEE Trans. Inf. Forensics Secur. **12**(8), 1830–1843 (2017)
31. M. Shen, G. Cheng, L. Zhu, X. Du, J. Hu, Content-based multi-source encrypted image retrieval in clouds with privacy preservation. Future Gener. Comput. Syst. **109**, 621–632 (2018)
32. M. Shen, Y. Deng, L. Zhu, X. Du, N. Guizani, Privacy-preserving image retrieval for medical IoT systems: a blockchain-based approach. IEEE Netw. **33**(5), 27–33 (2019)
33. M. Shen, B. Ma, L. Zhu, X. Du, K. Xu, Secure phrase search for intelligent processing of encrypted data in cloud-based IoT. IEEE Internet Things J. **6**(2), 1998–2008 (2019)
34. M. Shen, X. Tang, L. Zhu, X. Du, M. Guizani, Privacy-preserving support vector machine training over blockchain-based encrypted IoT data in smart cities. IEEE Internet Things J. **6**, 7702 (2019)
35. M. Shen, J. Duan, L. Zhu, J. Zhang, X. Du, M. Guizani, Blockchain-based incentives for secure and collaborative data sharing in multiple clouds. IEEE J. Sel. Areas Commun. **38**(6), 1229–1241 (2020)
36. M. Shen, H. Liu, L. Zhu, K. Xu, H. Yu, X. Du, M. Guizani, Blockchain-assisted secure device authentication for cross-domain industrial IoT. IEEE J. Sel. Areas Commun. **38**(5), 942–954 (2020)
37. R. Shokri, V. Shmatikov, Privacy-preserving deep learning, in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security* (ACM, New York, 2015), pp. 1310–1321
38. D.X. Song, D. Wagner, A. Perrig, Practical techniques for searches on encrypted data, in *IEEE S&P* (2000), pp. 44–55
39. J. Vaidya, H. Yu, X. Jiang, Privacy-preserving SVM classification. Knowl. Inf. Syst. **14**(2), 161–178 (2008)
40. A.M. Vegni, V. Loscri, A survey on vehicular social networks. IEEE Commun. Surv. Tutorials **17**(4), 2397–2419 (2015)
41. K. Xu, H. Yue, L. Guo, Y. Guo, Y. Fang, Privacy-preserving machine learning algorithms for big data systems, in *2015 IEEE 35th International Conference on Distributed Computing Systems* (IEEE, Piscataway, 2015), pp. 318–327
42. G. Xu, H. Li, Y. Dai, K. Yang, X. Lin, Enabling efficient and geometric range query with access control over encrypted spatial data. IEEE Trans. Inf. Forensics Secur. **14**(4), 870–885 (2019)
43. G. Xu, H. Li, S. Liu, M. Wen, R. Lu, Efficient and privacy-preserving truth discovery in mobile crowd sensing systems. IEEE Trans. Veh. Technol. **68**(4), 3854–3865 (2019)
44. G. Xu, H. Li, H. Ren, K. Yang, R.H. Deng, Data security issues in deep learning: attacks, countermeasures and opportunities. IEEE Commun. Mag. **57**(11), 116 (2019)
45. G. Xu, H. Li, S. Liu, K. Yang, X. Lin, Verifynet: secure and verifiable federated learning. IEEE Trans. Inf. Forensics Secur. **15**(1), 911–926 (2020)

46. H. Yu, J. Vaidya, X. Jiang, Privacy-preserving SVM classification on vertically partitioned data, in *Pacific-Asia Conference on Knowledge Discovery and Data Mining* (Springer, Berlin, 2006), pp. 647–656
47. J. Zhang, Z. Zhang, X. Xiao, Y. Yang, M. Winslett, Functional mechanism: regression analysis under differential privacy. Proc. VLDB Endowment **5**(11), 1364–1375 (2012)
48. Y. Zhang, J. Jiang, K. Xu, X. Nie, M.J. Reed, H. Wang, G. Yao, M. Zhang, K. Chen, BDS: a centralized near-optimal overlay network for inter-datacenter data replication, in *Proceedings of the Thirteenth EuroSys Conference* (ACM, New York, 2018), p. 10
49. B. Zheng, L. Zhu, M. Shen, X. Du, J. Yang, F. Gao, Y. Li, C. Zhang, S. Liu, S. Yin, Malicious bitcoin transaction tracing using incidence relation clustering, in *International Conference on Mobile Networks and Management* (Springer, Cham, 2017), pp. 313–323
50. B.-K. Zheng, L.-H. Zhu, M. Shen, F. Gao, C. Zhang, Y.-D. Li, J. Yang, Scalable and privacy-preserving data sharing based on blockchain. J. Comput. Sci. Technol. **33**(3), 557–567 (2018)
51. L. Zhu, B. Zheng, M. Shen, S. Yu, F. Gao, H. Li, K. Shi, K. Gai, Research on the security of blockchain data: a survey (2018). Preprint. arXiv:1812.02009

Chapter 6
Secure Data Retrieval Using Blockchain

Medical IoT devices are gradually being widely used, and acquisition requirements such as image type and quantity are also changing. Retrieving medical images can help diagnose the condition and improve treatment timeliness. However, medical images contain a large amount of sensitive patient information, which can easily cause privacy leakage. At the same time, existing research cannot protect the sensitive information of medical images or achieve data sharing [15]. Therefore, it is of practical significance to study the realization of medical image data sharing and retrieval under privacy protection. This chapter proposes a medical image retrieval system based on emerging blockchain technology that can protect image privacy and give a system model and a threat model. Later, we not only capture specific feature vectors based on medical images but also design the transaction structure accordingly, so as to achieve the purpose of privacy protection [6, 21]. Finally, the effectiveness and safety are demonstrated through theories and experiments [23].

6.1 Overview

Medical image retrieval, as an important diagnostic method for doctors to judge the patient's condition is of considerable significance to the entire medical field. However, today's hospitals are isolated information islands. There are both technical barriers and trust crises in sharing data between hospitals.

Existing solutions using cloud services as a medium for data transmission and sharing [13, 20], but the cloud service itself as a "semi-honest" third party, there is a risk of leaking data privacy, and the frequency of interaction with users is high, so the process is usually cumbersome [8, 14, 24].

The emergence of blockchain has brought a new dawn to solve trusted data sharing [17, 19]. The decentralized point-to-point structure of the blockchain can

M. Shen et al., *Blockchain: Empowering Secure Data Sharing*,
https://doi.org/10.1007/978-981-15-5939-6_6

solve the problem of untrustworthy third parties. At the same time, the open distributed ledger also ensures that the security of the data cannot be falsified.

Our solution wants to solve the problem is that the user uploads a medical image, and the system platform can help him retrieve the image with the highest similarity to the image he provides. The platform search for similar conditions and treatment options to achieve the purpose of assistive care. Our data sources should be as wide as possible, and the image providers are multi-party hospitals, not single ones, thus ensuring reference value. On this basis, to ensure the security of images provided by hospitals and the security of user-uploaded images.

The retrieval function is implemented on the blockchain. The scheme is mainly carried out in two steps. One is to establish a full node similar to the function of "miner" node, as the medical image retrieval service node of the system. The node has all the medical image information on the chain and is a trusted third party. The service node is used to establish an index table for the image library. When the user makes an access request, the service node searches the image information with the highest similarity through the index table and returns it to the user, thereby ensuring data security and ensuring retrieval efficiency. The second is to write a smart contract running on the chain, which can be automatically executed when the user makes an access request and provides retrieval for the user.

6.2 System Model

The system model designed by this scheme is shown in the following figure. There are four main roles, namely hospital nodes, image retrieval service node, regulatory agencies, and data users. The system diagram is shown in Fig. 6.1.

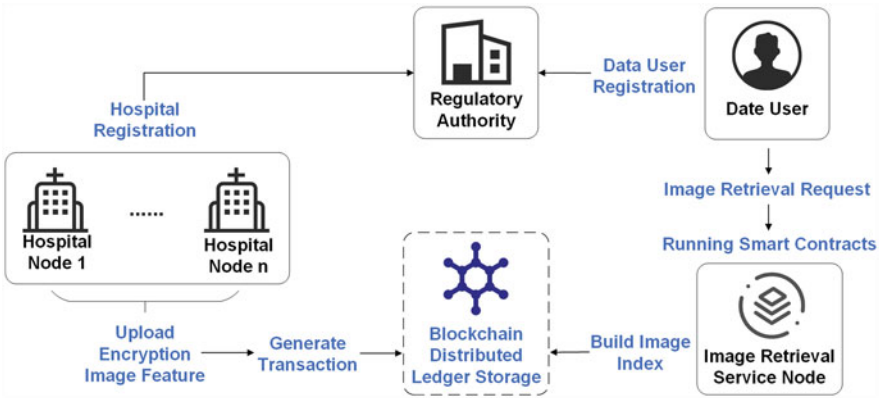


Fig. 6.1 Blockchain based medical image retrieval system

As shown in Fig. 6.1, the block represents four character entities. In this ecosystem of blockchain image retrieval, all hospitals and users are registered in the regulatory agencies and joined the alliance chain at first. After the hospital nodes encrypt the features of the diagnostic images of different types of patients, it is packaged and uploaded to the alliance chain for distributed storage together with the corresponding medical treatment information. The image service node synchronizes and updates, and acquires all existing encrypted image feature information from the chain and downloads it to a local database for storage and index. A smart contract is deployed on the image retrieval service node to process the data user's access request, and the user is searched for the image with the highest similarity, thereby helping the user to judge the condition and achieve the auxiliary treatment effect.

- Hospital Node

The hospital node is the provider of image data, extracting the features of the patient diagnosis image collected by the IoT device. It encrypts the feature using the hospital's private key, then generates a transaction and uploads it to the alliance chain. The input of the transaction is the encrypted image feature, and the output is a public address that is convenient for the image retrieval node to download. The design of the scheme is based on the alliance blockchain, so it is necessary to submit identity information to the regulatory authority for registration.

- Image Retrieval Service Node

The image retrieval service node is a full node on the alliance chain platform and performs the task of image retrieval on the platform. In this system, the transaction medical data uploaded by each hospital is output to a public address. The image retrieval service node obtains the encryption features of all medical images on the chain by accessing the public address, and downloads them into a local database, then indexes all the image information to improve the retrieval efficiency. The image retrieval service node automatically updates the index every so often. On this image retrieval service node, run a piece of smart contract code. When the user proposes an image data retrieval request, the code is automatically executed, and the encrypted image feature most similar to the image provided by the user is found through indexing and secure multi-party calculation.

- Regulatory Agencies

Regulatory agencies play a role in maintaining the smooth operation of the entire blockchain platform. They can prevent malicious nodes from tampering with the data on the chain while preventing malicious users from forging transactions or attacking smart contracts. In addition, regulatory agencies are responsible for verifying the legality of the authentication information of the hospital and third-party users. When hospital nodes and data users join the alliance chain platform, they need to register an account with regulatory agencies.

- Data Users

When data users want to use the platform for retrieval, they submit the encryption features of their images to the smart contract. After identity authentication, the image retrieval service node performs image retrieval based on content similarity on the image features in the database and returns similar image features, image numbers, and diagnostic treatment information associated with the image are queried for the smart contract. After receiving the search result, the smart contract verifies the integrity of the result and confirms that the verification result is correct, then returns it to the user for reference. If the data user additionally wishes to obtain the original image of the retrieved result, it is done by paying the hospital a certain amount of bitcoin.

6.2.1 Threat Model

- Privacy Disclosure

Privacy leaks caused by members without access rights. They are members who have not been registered and certified by a regulatory agency and legally joined to the sharing platform. They may be information thefts, malicious attack nodes, unauthorized hospital nodes or user nodes. During the whole process of system operation, members without access rights may threaten the privacy of image information on the platform. For example, when the hospital node uploads the encrypted image feature set to the alliance chain, the image retrieval service node obtains image data from the chain, or when the user makes access for a request to retrieve. At the same time, they may also steal private information from the image directly from the set of encrypted image features stored on the alliance chain. Image feature encryption is usually used to prevent this type of attack. First, using features to represent images can save storage space and does not leak original images. Second, encrypting features can better protect data safety.

- Data Tampering

In the process of uploading image information by the hospital node, although the alliance chain platform can ensure that the uploaded image data will not be deleted or tampered with, there may be a potential attacker tampering with the image data that has not been uploaded to the blockchain, or the smart contract is tampered with to achieve some of the personal business objectives of the individual (the image retrieval service node in this design is a trusted third-party node). Defense of this type of attack usually takes the form of a federated chain or a private chain. All members need to register on the chain and provide their own authentication information to ensure that the image providers that are connected to the chain to operate the data are safe and reliable.

- Forgery Attacks

If the user needs to obtain the original information of the image, the user needs to pay a certain amount of bitcoin. But how to set access rights to ensure that users do not spread or falsify data or even illegally sell the data, is still a problem that needs to be resolved.

6.2.2 Design Goals

The design goal of our proposed privacy-based blockchain medical image retrieval program mainly includes the following three aspects:

- Image Feature Privacy Protection

The privacy protection of image features is a significant indicator in the encrypted image retrieval service. Since the hospital node wants to upload the image features to the blockchain, there may be some nodes that do not have access rights try to acquire and analyze the content information of the image from the chain, and perform illegal transactions. Therefore, the design of this scheme should ensure that the image service retrieval node on the blockchain is a trusted third party. The person without access rights cannot analyze the privacy information of the image content, the image feature privacy information, and the similarity from the encrypted image feature or transaction information.

- Image Retrieval Accuracy

Image retrieval accuracy is an important indicator to ensure the basic functions of the search platform. Image retrieval on the system platform is mainly done by the image retrieval service node. The information is stored in the local database by acquiring and synchronizing the image feature data on the chain and indexing the data. The process of retrieving is performed by smart contracts, so the retrieval accuracy depends on the service node on the one hand and the smart contract on the other. We should ensure that the effect of encrypted search and plaintext search in this scheme is within the same level.

- Efficiency of The Program

The efficiency of the system scheme includes image feature encryption efficiency, the efficiency of encryption feature storage to the chain, efficiency of image retrieval service node downloading and updating data, system platform processing transaction efficiency, retrieval index establishment efficiency, and encrypted image feature retrieval efficiency. The efficiency of image feature retrieval under encryption should be at the same level as that under plaintext, ensuring the credibility and validity of the scheme.

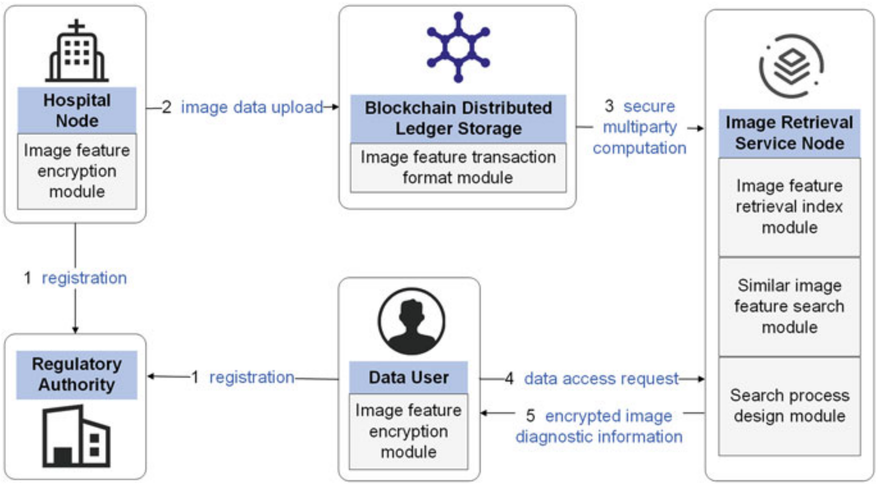

Hospital Node
Image feature encryption module
2 image data upload
Blockchain Distributed Ledger Storage
Image feature transaction format module
3 secure multiparty computation
Image Retrieval Service Node
Image feature retrieval index module
Similar image feature search module
Search process design module
1 registration
Regulatory Authority
1 registration
Data User
Image feature encryption module
4 data access request
5 encrypted image diagnostic information

Fig. 6.2 System architecture and workflow diagram

6.2.3 System Architecture

This section mainly introduces the system architecture of the solution and explains the operation and connection between the modules. The design of this scheme mainly includes five key modules. The system architecture diagram of each module is shown in Fig. 6.2. The modules are briefly introduced as follows.

- Image Feature Encryption Module

Hospital nodes and data consumer nodes use the image feature encryption module. When the medical node uploads the image of the patient, the image feature needs to be extracted, and then the feature is encrypted by the method used by the encryption module. Then packaged and uploaded to the chain to realize the security protection of the data. When the data user makes a query request to the system, it needs to extract and encrypt the image features in the same way, and submit it to the image service retrieval node for a search query.

- Image Feature Transaction Structure Module

Drawing on the transaction structure of data in the traditional Bitcoin trading system, this scheme redesigns the transaction structure of the image retrieval process. In the system, the input side of the transaction is the encrypted image feature information packaged and uploaded by the hospital node, and the output is a public address, which is convenient for the image service retrieval node to download. More importantly, it is necessary to design that the encrypted feature information of the image and the diagnostic information corresponding to the image are stored in the transaction of the block, so the transaction structure on the alliance chain platform can satisfy the image retrieval demand.

- Image Feature Retrieval Index Module

Considering the huge number of image features uploaded by hospital nodes in real application scenarios and the long time of each retrieval, a retrieval index module is designed to facilitate the execution of smart contracts and speed up the efficiency of the query process.

- Similar Image Feature Retrieval Module

This module is a content-based image retrieval method and a secure multi-party computing encryption design [18, 27]. In this system, multi-party hospital nodes are participating in data sharing. Therefore, it is considered to encrypt the image features by means of secure multi-party computation [4, 22]. Then the method of calculating the feature similarity by Euclidean distance is improved for the content-based image retrieval in the traditional mode, in order to comply with this design.

- Retrieval Process Design Module

Based on the image feature retrieval index module, the search process design module is designed to respond to the user's data retrieval access request using a smart contract. The user issues an image retrieval request to provide an encryption feature of the image to be queried, and the smart contract uploads the user's query request to the image retrieval service node. The node searches for images that are closest to the image feature to be queried according to the index, and the result of the query will be packaged with the corresponding diagnostic information returned to the smart contract together. The smart contract verifies the integrity of the search result then finally feeds back to the user, helping the data user assist in diagnosing the disease condition.

The workflow of the system design is described as follows:

First, the system scheme is established on the alliance chain. So the hospital nodes and the data user nodes must first register the identity authentication information on the regulatory authority node. In order to ensure the security in the whole retrieval process, and ensure that the unregistered node will not Malicious attacks and take potential threats to the system.

Second, the hospital node initiates the image feature encryption module to package the image features of the patients and upload them to the alliance chain. The image feature transaction structure module should be designed before the system runs. The most significant change in the transaction structure is mainly the input, output, and storage content compared to the bitcoin system. In the second step of the process, the packaged image feature data are output to a public address, then the transaction generated.

Then, the image retrieval service node downloads the encrypted image features uploaded by the hospitals from the output public address of the transaction, stores them in the local database, indexes the data based on the method of secure multi-party calculation, and starts the image feature retrieval index module.

The next step, when the data user needs to access the image on the data sharing platform, the user activates the image feature encryption module to process the image to be retrieved and submits the encrypted image feature to the smart contract. The smart contract transmits the information to the image retrieve service node. At this point, the image retrieval service node initiates the retrieval process design module to retrieve for the user. The retrieval is based on the image similarity to do the content-based image retrieval. Find the most matching image information for the user's image based on the image feature set and the index generated by the index module.

Following that, the feedback result of the image retrieval service node is first verified integrity by the smart contract. Then the result of the verification is returned to the user, including the hospital information, the number corresponding to the similar encrypted image features, and the diagnosis result corresponding to the images.

6.3 Design of Trading Format

6.3.1 Image Feature Encryption

The system is an image retrieval platform based on privacy protection. Therefore, the feature encryption of data is essential to protect data security. Each hospital node needs to extract the features of the image before uploading the data to the blockchain.

The hospital node first extracts the features of the image. This scheme adopts MPEG−7, which is called "Multimedia Content Description Interface" and is used to describe various features of images. Standard features include color, texture, and shape descriptors. In this system, the color structure histogram, edge histogram descriptor, and region shape descriptors are mainly extracted. The color histogram contains the color and structure information of the image, and the feature extraction adds the color and structure information of the pixels in the 8∗8 panes to the descriptor. Edge histograms can further improve the accuracy of retrieval when used in conjunction with other image features such as color features. The region shape descriptor takes all the pixels that make up a shape within a frame, while the descriptor is robust to small deformations of the boundary.

The process of encrypting features is as follows. Assume the eigenvector of an image $e = \{m_1, m_2, \dots, m_l\}$. Among them l is the dimension of the image feature e. The feature is encrypted in a secure multi-party calculation, the sum of e and e^2 are processed first.

First, select two prime numbers a and b of the same length so that $a - 1$ can be divisible by b. In the computation of this feature encryption, the parameter a is exposed to all hospital nodes, data consumers, and image retrieval service node. Select the random number h and calculate the h_1 and h_1 according to Eqs. (6.1) and (6.2).

Second, each hospital node p_i randomly selects a number c_i and calculate R_j in corresponding each dimension m_j of feature e in Eq. (6.3).

Third, according to Eq. (6.4), the ciphertext em_j corresponding to each dimension m_j is obtained, and the em_j and ${em_j}^2$ can be calculated [10].

$$h_1 = h^{(a-1)/b} \quad mod \quad a \tag{6.1}$$

$$h_2 = {h_1}^a \quad mod \quad a^2 \tag{6.2}$$

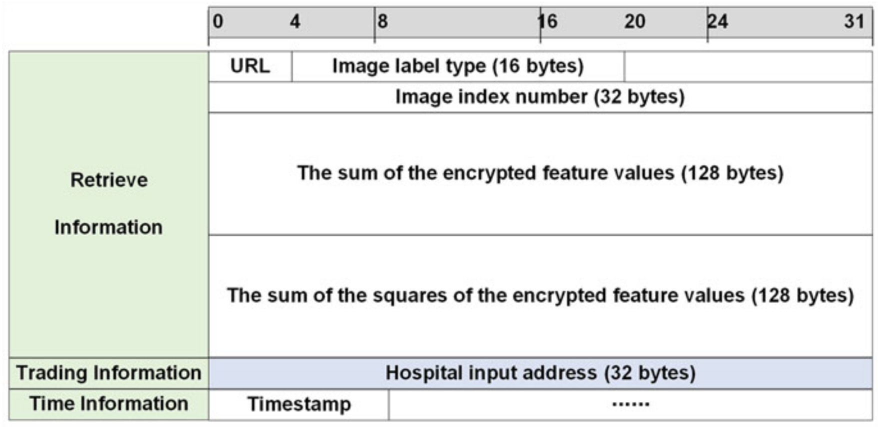
$$R_j = \left({h_2}^{c_{i+1}} / {h_1}^{c_{i-1}}\right)^{c_i} \tag{6.3}$$

$$em_j = \left(1 + m_j a\right) R_j \quad mod \quad a^2 \tag{6.4}$$

6.3.2 Image Feature Transaction Structure

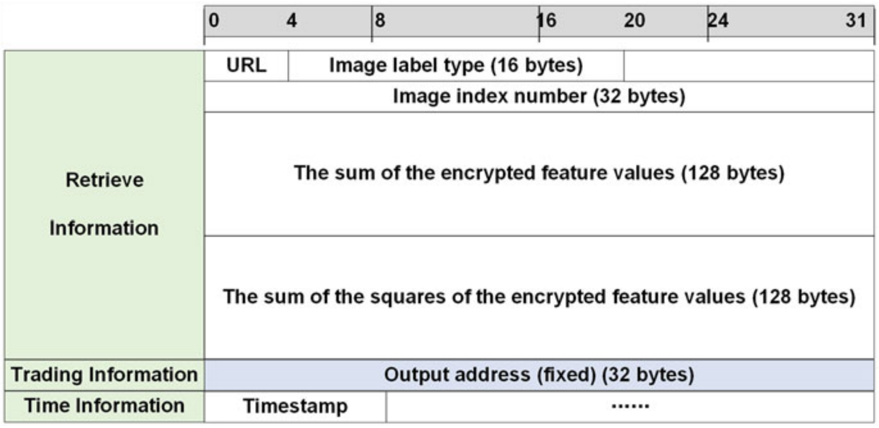
Based on the particularity of sharing data, the scheme redesigned the structure of the hospital nodes packaging and uploading data to the alliance chain, which facilitates the collection of data and retrieval process by the image service node. The design of the transaction structure is shown in Figs. 6.3 and 6.4.

The content stored in this trading system is divided into three parts: retrieval information, transaction information, and time information. A transaction refers to the process in which a hospital node packs and uploads image features and related information to the alliance chain for storage and outputs to a fixed address. The transaction consists of two parts, input, and output. The search and output information of the input and output are in the same format, except for the transaction



	0 4	8 16 20	24 31
Retrieve Information	URL	Image label type (16 bytes)	
	Image index number (32 bytes)		
	The sum of the encrypted feature values (128 bytes)		
	The sum of the squares of the encrypted feature values (128 bytes)		
Trading Information	Hospital input address (32 bytes)		
Time Information	Timestamp	······	

Fig. 6.3 Image feature input transaction



	0 4 8 16 20 24 31		
Retrieve Information	URL	Image label type (16 bytes)	
	Image index number (32 bytes)		
	The sum of the encrypted feature values (128 bytes)		
	The sum of the squares of the encrypted feature values (128 bytes)		
Trading Information	Output address (fixed) (32 bytes)		
Time Information	Timestamp	......	

Fig. 6.4 Image feature output transaction

information part. The input party is the hospital node, and the output party is a fixed transaction address in the design of the system, which is convenient for the image service retrieval node to download.

In the retrieval information part, the original image URL occupies 4 bytes of memory, which is convenient for the user to verify when needed. Assign 16 bytes to the image type label, where the label represents the patient's disease type, and the image is sorted in advance for easy retrieval. The image index number is the number of the image, which occupies 32 bytes. Considering that the number of images is gradually increased, large storage space is allocated. The sum of the encrypted feature values and the square of the encrypted feature values respectively occupy 128 bytes of storage space, because the feature encryption method adopted by the system is a secure multi-party calculation. When content-based image retrieval is performed, it is necessary to use these two values to find similar images and match the related images.

In the transaction information section, the input address of the 32-byte hospital node is input, and the output is a fixed output address of 32 bytes. In the time information section, a timestamp is kept to record the time of each transaction.

6.4 Feature Extraction

6.4.1 Image Feature Retrieval Index Library

The image retrieval service node stores the downloaded image feature set in a local database, and indexes the data to improve the retrieval efficiency of the encrypted image. The way to the index is to create an index for each image dataset uploaded by each hospital node. Because the use of secure multi-party computation in the

Table 6.1 Encrypted image retrieval index table

Hospital node number	Image index number	$\sum_{i=1}^{l} em_i$	$\sum_{i=1}^{l} {em_i}^2$
HN-1	IMNUM-1	HN-1-IMNUM-1-SUM1	HN-1-IMNUM-1-SUM1
HN-2	IMNUM-2	HN-2-IMNUM-2-SUM2	HN-2-IMNUM-2-SUM2
…	…	…	…
HN-n	IMNUM-n	HN-n-IMNUM-n-SUMn	HN-n-IMNUM-n-SUMn

process of feature encryption involves the em_j and ${em_j}^2$, the calculation $em = \{em_1, em_2, \ldots, em_l\}$ and $em = \left\{{em_1}^2, {em_2}^2, \ldots, {em_l}^2\right\}$, that are $\sum_{i=1}^{l} em_i$ and $\sum_{i=1}^{l} {em_i}^2$, so we index the sum of the features and the squares of the features, simplify the indexing process, and speed up the search. The index Table 6.1 created is shown below.

6.4.2 *Similar Image Feature Retrieval*

In this system approach, the task of the image retrieval service node is to retrieve images of features most similar to the images provided by the user to assist the user for the diagnostic information reference. The academic community usually uses the similarity between image features to estimate the similarity of images. The higher the similarity, the smaller the distance between features. The images are searched using Euclidean distance and the results are sorted according to the similarity of the images. Assuming the feature vectors of the two images are $E = \{m_1, m_2, \ldots, m_l\}$, $F = \{n_1, n_2, \ldots, n_l\}$ the Euclidean distance between them is as shown in Eq. (6.5) below.

$$\begin{aligned} EucDis &= \sqrt{(m_1 - n_1)^2 + (m_2 - n_2)^2 + \ldots + (m_l - n_l)^2} \\ &= \sqrt{\sum_{i=1}^{l} {m_i}^2 + \sum_{i=1}^{l} {n_i}^2 - \sum_{i=1}^{l} 2m_i n_i} \end{aligned} \tag{6.5}$$

Through this formula, the similarity between the two images can be calculated. Therefore, in the index table, we design the inclusion of the sum of the encrypted image features in the table to improve the calculation speed.

6.5 Secure Data Retrieval

6.5.1 Image Retrieval Process Design

When a user makes an access request, he submits the encryption features of an image to the smart contract. The smart contract receives the request and verifies the user's identity. After the verification, the information submitted by the user is uploaded to the image retrieval service node. The retrieval service node starts the image similarity search process, finds the image feature information that most matches the user, and returns it to the smart contract, which verifies the integrity of the retrieval results and ensures that the data is accurate before returning to the user.

In the process design, we introduce the smart contracts for two aspects: one is not to let the user directly connecting with image retrieval service node, to protect the user privacy, the second is in the scheme design of image retrieval service node is a reliable third party, but with the expansion of the volume, it needs to have a regulatory role in ensuring the quality of service node, so consider adopting smart contracts to verify the integrity of the results.

6.5.2 Image Retrieval Service Update

The image retrieval service node may add, delete, or update the indexed image feature nodes. Whenever the number of images submitted by the hospital node increases to a certain level, the retrieval service node needs to update the index table. In the system design, the scheme can support dynamic operations of adding, deleting, and updating on the retrieval service node.

- Add

When the number of images uploaded by the hospital nodes to the public address on the chain is increasing, the image retrieval service node will download all the image feature information from the public address every time a fixed time, and add the new image data to the search library.

- Delete

If the image retrieval service node finds that the downloaded image has duplicate uploaded image information, it will delete the downloaded image, delete the encrypted image features from the encrypted image database, and their corresponding numbers and related records in the index table.

- Update

Due to the need for data privacy protection, the hospital node may want to change the parameters used to encrypt the image features after uploading a batch of images. Then the hospital node re-encrypts the batch of images. At this point, the hospital nodes will re-encrypt the image information after re-encryption and upload it to the chain. When the image retrieval service node receives the images, it checks the image number and finds the same number of images, which will overwrite the previous version with the latest version.

6.6 Implementation and Evaluation

6.6.1 Experimental Setup

In the field of medical imaging, MedPix1 is widely used as an open-source online database. The database covers medical images, teaching cases, and textual metadata for different clinical topics, including cases from more than 12,000 patients, and a total of nearly 59,000 medical images. The audience includes doctors, medical professionals, medical researchers, and medical students. The medical image database is classified according to types such as disease location (organ system), pathological category, patient file, image type, and label, which facilitates the extraction of different types of data. We selected 10 label classifications from the data set, each with 25 images, and a total of 250 medical images were used as query targets to conduct experiments and evaluate the system solution. The total number of uploaded images we set is 10,000.

The experimental platform uses Win7 operating system and Core (TM) i5-4200U CPU. The experimental environment uses Hy-perledger fabric, which is a good carrier for realizing the alliance chain in the blockchain. Because the mining process is not required, the transaction speed is breakneck. At present, it can reach a throughput of at least 1000/s times. The contract controls the transaction logic in the form of a chain code. The programming language used is Go.

For the third party, it will extract the EHD edge histogram features of each image to be queried [25]. Assume that the extraction data provider extracts a medical image with a gray level of 8 and pixels of 640*480. The third party encrypts the features of the image, according to ImgFeaEnc. Assume that the values of prime numbers p, q, h ($a - 1$ can be b divide) are selected to encrypt the image features.

The third party sends to the smart contract according to the feature set EQ of the image to be queried and the verification identity ID. After the smart contract verifies the legality of the transaction, it is uploaded to the image retrieval service for retrieval. The image retrieval service will query similar images according to the established index table ϕ.

After a third party submits the retrieval request, the image retrieval service will calculate the similarity based on the euclidean distance using the image to be queried by the third party and the image feature set in the index database. The images in the index database are derived from the image features in the transaction information parsed from the public address by the retrieval service. The retrieval service extracts the third and fourth items from the index table ϕ and sequentially calculates the similarity with the third-party image to be queried. After traversing the entire index database, the top K encrypted images with the highest similarity to the query image are obtained, and then the retrieval results are returned to the third party through a smart contract.

6.6.2 Time Complexity

The time complexity [5, 12] targets the complexity of medical image feature encryption time, image retrieval service indexing time, and similarity retrieval time for encrypted medical images [16]. For medical image feature encryption time, this article uses a secure multi-party calculation to encrypt it. Assuming that the dimension of a medical image feature is n, the time complexity of encrypting the feature is O(n), if there are m n-dimensional, The time complexity of encryption is O(mn). The image retrieval service node will automatically download image feature information from the public address after a certain period, and establish an index database to improve retrieval efficiency. The complexity of establishing retrieval time has a linear relationship with the number of image features. When the number of features is m, the complexity of establishing the retrieval time is O(m).

We analyze the statistics of the upload time of encrypted features and the plaintext feature when the number of images uploaded to the chain is from 5000 to 10,000, as shown in Fig. 6.5. The upload time range of both schemes is from 80 s to 180 s, and the upload time increases with the number of images. For the encryption

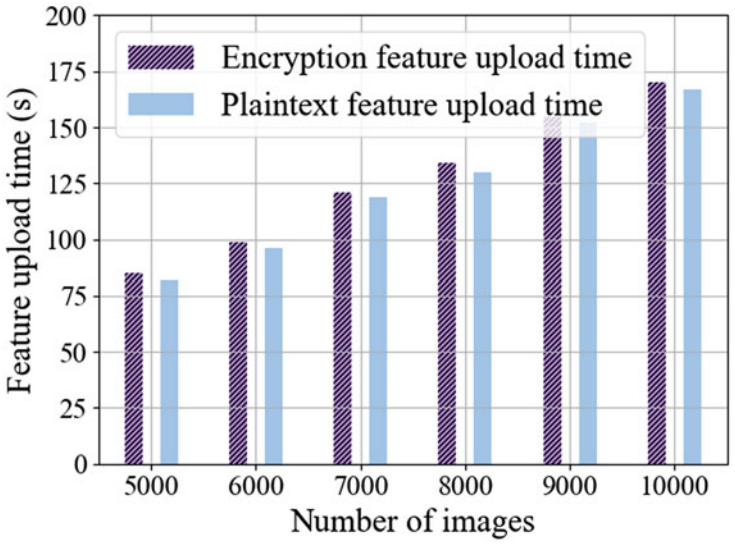


Fig. 6.5 Feature upload time

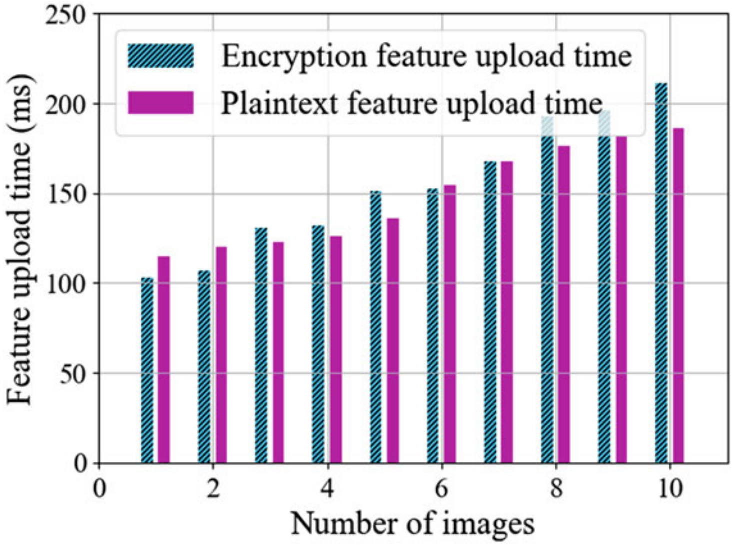


Fig. 6.6 Encryption feature and plaintext feature upload time

feature upload time and the plaintext feature upload time, when the number of uploaded images is more than 6000, there is a relatively obvious increase in the upload time, and the upload time has increased by about 19%. The time overhead brought by the encrypted on-chain over the plaintext on-chain does not exceed 5 s, which is within the acceptable range.

At the same time, we also performed a set of control experiments in this scheme to test the relationship between the number of images uploaded in a transaction and the time of uploading on a hospital node. As shown in 6.6, the on-chain time for processing a transaction ranges from 100 ms to 200 ms. The time when the plaintext feature is uploaded and the time when the feature is encrypted with the number of images are consistent. When the number of images in a transaction is less than 3, the processing time of the chain fluctuates greatly, due to the characteristics of different images. When the number of images is greater than 7, the time of the chaining of encrypted feature images depends on the number of images. When it is larger than 7 sheets, there is a clear improvement.

The encryption feature upload time and the plaintext feature upload time are listed separately, as shown in Fig. 6.6. Both the encryption feature upload time and the plaintext feature upload time increase with the number of images. Considering the upper limit of transaction throughput, the maximum number of images uploaded in one transaction is 10. Moreover the on-chain time of plaintext and ciphertext is the same. It shows that the overhead of the ciphertext of this scheme is not large. Under the same retrieval accuracy, the use of ciphertext can more effectively protect the privacy and security of image data.

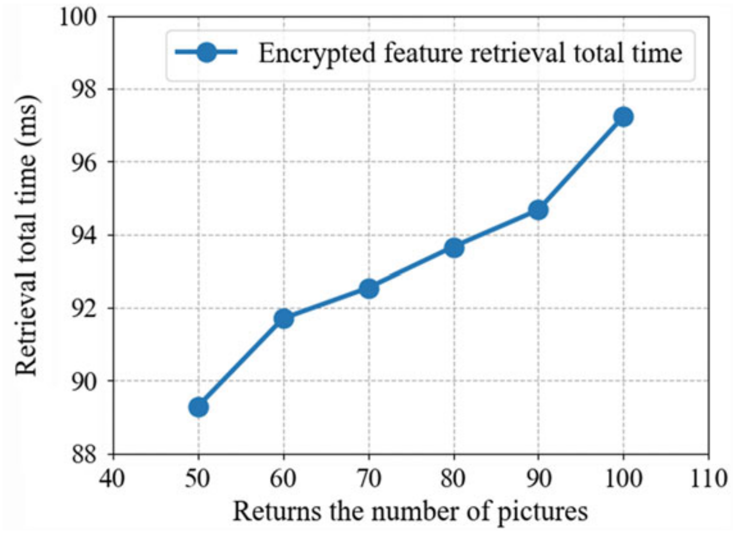


Fig. 6.7 Encrypted feature retrieval time

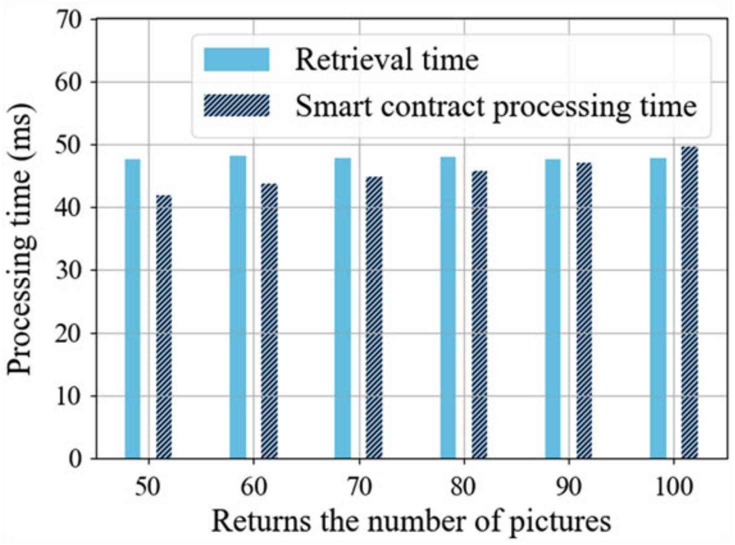


Fig. 6.8 Retrieval time and smart contract runtime

A good search solution should have a short search time. Figures 6.7 and 6.8 respectively show that when the number of images returned to the user after the image retrieval service node retrieves ranges from 50 to 100, from the client Time trend of the retrieval request to the retrieval node to return the retrieval result. Here we set the total number of retrieved images to 10,000. The total running time in the figure refers to the time from when the user initiates an image retrieval request transaction to the smart contract calling the retrieval node for similarity retrieval of the image, and then returns the retrieval result to the smart contract for data integrity verification, and finally returns the result to the user total time. That is, the sum of the retrieval time of the Fig. 6.8 and the processing time of the corresponding Fig. 6.7.

When the number of retrieved images increases from 50 to 100, the total running time increases from 89 ms to 97 ms, and the time overhead is minimal, which illustrates the efficiency and feasibility of our retrieval scheme. The retrieval time in Fig. 6.8 refers to the running time of the retrieval service node, searching for similar images from the index database, and then performing similarity calculation. It can be seen that the retrieval time increases with the number of images, and the smart contract processing time is unchanged. Figure 6.8 is the node retrieval process time and the smart contract processing request time. Processing time refers to the sum of the two periods when the user submits a retrieval access request to the smart contract, and the smart contract feeds back the processing result to the user. The processing time increased slightly with the increase in the number of images. The average processing time was 95 ms, and the processing time was relatively stable.

We compare the scheme with the similar scheme [18] in the case of ciphertext and plaintext, and analyze the retrieval time under the same number of image retrievals as shown in Fig. 6.9. The comparison of retrieval time between the total number of retrieved images from 2000 to 10,000 was analyzed. It is found that the total time consumed by the encryption feature scheme is basically the same as the total time consumed by the plaintext scheme, and the retrieval time of the similar

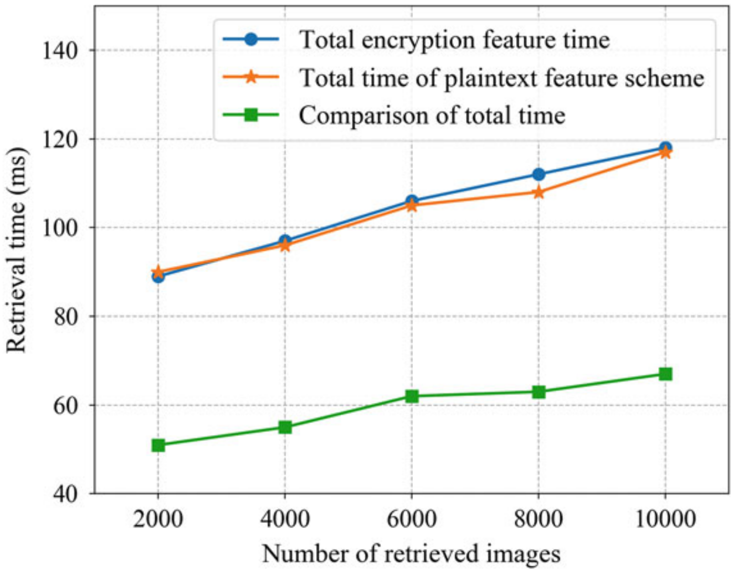


Fig. 6.9 Comparison of total search time

scheme is basically about half of the scheme. Combining the above Figs. 6.7 and 6.8, the total retrieval time of our scheme is the communication overhead of the node processing the retrieval process plus the smart contract processing the user request and feedback, and the proportion of communication overhead and retrieval processing time is also 50% each. Therefore, compared with the comparison scheme, this scheme only has communication overhead. In the comparison scheme, the author ignores the communication overhead on the cloud.

6.6.3 *Storage Complexity*

The storage complexity [2, 26, 28] increases with the number of image features, and the indexing time of the image retrieval service node also increases, because the storage space occupied by the image is increasing. Assuming that the number of medical image features is m, the storage complexity of the system scheme can be expressed as O (m).

This section compares the encryption image feature storage complexity, retrieval index storage complexity, and plaintext image feature storage complexity of the scheme. The data obtained are shown in Table 6.2. The following table lists the increase in storage space when the number of image processing increases, and also calculates the storage overhead caused by encrypted image features [3, 30]. Experiments were performed on changes in the number of images from 2000 to 10,000 encrypted image features, plaintext image features, and retrieval index time. It can be seen from the table that as the number of images in the image set increases, the storage space required for the encrypted image set, the encrypted image feature set, and the retrieval index table will also increase. Encryption increases the storage space of the image. In this solution, the storage index of 10,000 images requires approximately 284 KB of space, and the encryption feature of 10,000 images requires approximately 112 MB of space. Therefore, it can be seen that the storage space required for the retrieval index is the smallest, which is more helpful to improve retrieval efficiency. It can be seen that under the same number of images, the storage space occupied by encrypted image features is larger than the space

Table 6.2 Image retrieval storage space table

Number of images (photos)	Encrypted image features (KB)	Plaintext image features (KB)	Search index (KB)
2000	21,870	21,285	42
4000	47,454	35,843	89
6000	71,152	70,761	165
8000	91,951	87,820	216
10,000	115,423	90,648	284

Table 6.3 Comparison of retrieval accuracy of various schemes

	The proposed method				Cheng's method [18]			
	Encrypted		Plaintext		Encrypted		Plaintext	
Images returned	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
20	36.96	6.56	36.96	6.56	34.80	6.96	34.80	6.96
40	30.94	10.94	30.94	10.94	29.85	11.94	29.85	11.94
60	30.40	16.40	30.40	16.40	29.00	17.40	29.00	17.40
80	27.92	19.92	27.92	19.92	27.40	21.92	27.40	21.92
100	27.64	25.64	27.64	25.64	26.64	26.64	26.64	26.64

occupied by plaintext image. However, it is basically on order of magnitude, and the increase in storage space does not exceed 5%, which is an acceptable range.

6.6.4 *Retrieval Accuracy*

Retrieval accuracy [7, 9, 29] and recall rate [1, 11] are two very important evaluation indicators in the field of image retrieval. As shown in the image retrieval accuracy Eq. (6.6), TP stands for accurate classification, and judges the number of similar images of the same type as FP. FP stands for the number of similar images that were originally misidentified as two different types of images. As shown in the image retrieval recall in Eq. (6.7), FN represents the number of images that should be determined to be similar but misidentified to be of different types. We will get the accuracy comparison of encrypted search, plaintext search and Cheng's search scheme [18] from Table 6.3. The results are shown below. From the experimental results, the retrieval accuracy and recall rate in the case of plaintext and ciphertext are the same, and the retrieval accuracy rate and recall rate of the comparison scheme is not much different.

$$P = \frac{TP}{TP + FP} \tag{6.6}$$

$$R = \frac{TP}{TP + FN} \tag{6.7}$$

6.7 Summary

This chapter introduces the design and implementation of a retrieval platform based on privacy protection for federated chain encryption. The system model and threat model are first defined, and the design goals of the solution are described. In the solution design part, the system architecture is first introduced, the workflow of the role composition and distribution is explained, followed by the design of several detailed modules. Image feature encryption is a privacy protection issue that solves the problem of uploading images from hospital nodes.

The image feature retrieval index library is designed to speed up the retrieval process and improve retrieval efficiency. This scheme the image retrieval method based on content similarity, adopts smart contracts to simplify and constrain the retrieval process, and provides a synchronous update of retrieval service to ensure real-time data update. The specific introduction and implementation of the above program are realized.

References

1. N.S. Admile, R.R. Dhawan, Content based image retrieval using feature extracted from dot diffusion block truncation coding, in *Proceeding of 2016 International Conference on Communication and Electronics Systems (ICCES)* (IEEE, New York, 2016), pp. 1–6
2. S. Chandra, W.W. Hsu, Lossless medical image compression in a block-based storage system, in *Proceeding of 2014 Data Compression Conference* (IEEE, New York, 2014), pp. 400–400
3. H.G. Do, W.K. Ng, Blockchain-based system for secure data storage with private keyword search, in *Proceeding of 2017 IEEE World Congress on Services (SERVICES)* (IEEE, New York, 2017), pp. 90–93
4. X. Du, M. Guizani, Y. Xiao, H.-H. Chen, Transactions papers a routing-driven elliptic curve cryptography based key management scheme for heterogeneous sensor networks. IEEE Trans. Wirel. Commun. **8**(3), 1223–1229 (2009)
5. H. Fang, R. Li, L. Wang, X. Li, A full-rate and low-detection complexity cooperative transmission scheme based on distributed space-time-frequency code design, in *Proceeding of 2013 IEEE 4th International Conference on Software Engineering and Service Science* (IEEE, New York, 2013), pp. 977–980
6. F. Gao, L. Zhu, M. Shen, K. Sharif, Z. Wan, K. Ren, A blockchain-based privacy-preserving payment mechanism for vehicle-to-grid networks. IEEE Netw. **32**(6), 184–192 (2018)
7. R.K. Grace, R. Manimegalai, S.S. Kumar, Medical image retrieval system in grid using hadoop framework, in *Proceeding of 2014 International Conference on Computational Science and Computational Intelligence*, vol. 1 (IEEE, New York, 2014), pp. 144–148
8. Z. Guan, J. Li, L. Wu, Y. Zhang, J. Wu, X. Du, Achieving efficient and secure data acquisition for cloud-supported internet of things in smart grid. IEEE Internet Things J. **4**(6), 1934–1944 (2017)
9. A.K. Jain, J.-E. Lee, R. Jin, N. Gregg, Content-based image retrieval: an application to tattoo images, in *2009 16th IEEE International Conference on Image Processing (ICIP)* (IEEE, New York, 2009), pp. 2745–2748
10. T. Jung, X.-Y. Li, M. Wan, Collusion-tolerable privacy-preserving sum and product calculation without secure channel. IEEE Trans. Dependable Secure Comput. **12**(1), 45–57 (2014)
11. B. Jyothi, Y. MadhaveeLatha, P.K. Mohan, Region based texture descriptor for content based medical image retrieval using second order moments, in *2015 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)* (IEEE, New York, 2015), pp. 1–4

12. K.S. Kumar, D. Malathi, A novel method to find time complexity of an algorithm by using control flow graph, in *Proceeding of 2017 International Conference on Technical Advancements in Computers and Communications (ICTACC)* (IEEE, New York, 2017), pp. 66–68
13. X. Li, Q. Xue, M.C. Chuah, Casheirs: cloud assisted scalable hierarchical encrypted based image retrieval system, in *IEEE INFOCOM 2017-IEEE Conference on Computer Communications* (IEEE, New York, 2017), pp. 1–9
14. Y. Li, L. Zhu, M. Shen, F. Gao, B. Zheng, X. Du, S. Liu, S. Yin, Cloudshare: towards a cost-efficient and privacy-preserving alliance cloud using permissioned blockchains, in *International Conference on Mobile Networks and Management* (Springer, Berlin, 2017), pp. 339–352
15. H. Li, L. Zhu, M. Shen, F. Gao, X. Tao, S. Liu, Blockchain-based data preservation system for medical data. J. Med. Syst. **42**(8), 141 (2018)
16. L. Lin, G. Zames, et al., Time complexity and model complexity of fast identification of continuous-time lti systems. IEEE Trans. Autom. Control **44**(10), 1814–1828 (1999)
17. O. Novo, Blockchain meets IoT: an architecture for scalable access management in IoT. IEEE Internet Things J. **5**(2), 1184–1195 (2018)
18. M. Shen, G. Cheng, L. Zhu, X. Du, J. Hu, Content-based multi-source encrypted image retrieval in clouds with privacy preservation. Futur. Gener. Comput. Syst. (2018)
19. M. Shen, J. Duan, L. Zhu, J. Zhang, X. Du, M. Guizani, Blockchain-based incentives for secure and collaborative data sharing in multiple clouds. IEEE J. Sel. Areas Commun. **38**(6), 1229–1241 (2020)
20. M. Shen, H. Liu, L. Zhu, K. Xu, H. Yu, X. Du, M. Guizani, Blockchain-assisted secure device authentication for cross-domain industrial IoT. IEEE J. Sel. Areas Commun. **38**(5), 942–954 (2020)
21. M. Shen, B. Ma, L. Zhu, X. Du, K. Xu, Secure phrase search for intelligent processing of encrypted data in cloud-based IoT. IEEE Internet Things J. **6**(2), 1998–2008 (2018)
22. M. Shen, B. Ma, L. Zhu, R. Mijumbi, X. Du, J. Hu, Cloud-based approximate constrained shortest distance queries over encrypted graphs with privacy protection. IEEE Trans. Inf. Forensics Secur. **13**(4), 940–953 (2018)
23. M. Shen, Y. Deng, L. Zhu, X. Du, N. Guizani, Privacy-preserving image retrieval for medical IoT systems: a blockchain-based approach. IEEE Netw. **33**(5), 27–33 (2019)
24. M. Shen, B. Ma, L. Zhu, X. Du, K. Xu, Secure phrase search for intelligent processing of encrypted data in cloud-based IoT. IEEE Internet Things J. **6**(2), 1998–2008 (2019)
25. V. Vida, H. Mehrpouyan, High rate and low complexity space-time block codes for $2\times$ 2 mimo systems. IEEE Commun. Lett. **20**(6), 1227–1230 (2016)
26. H. Wang, D. Yang, N. Duan, Y. Guo, L. Zhang, Medusa: blockchain powered log storage system, in *2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS)* (IEEE, New York, 2018), pp. 518–521
27. Z. Xia, Y. Zhu, X. Sun, Z. Qin, K. Ren, Towards privacy-preserving content-based image retrieval in cloud computing. IEEE Trans. Cloud Comput. **6**(1), 276–286 (2015)
28. Y. Xu, Section-blockchain: a storage reduced blockchain protocol, the foundation of an autotrophic decentralized storage architecture, in *2018 23rd International Conference on Engineering of Complex Computer Systems (ICECCS)* (IEEE, New York, 2018), pp. 115–125
29. X. Zhang, W. Liu, M. Dundar, S. Badve, S. Zhang, Towards large-scale histopathological image analysis: Hashing-based image retrieval. IEEE Trans. Med. Imaging **34**(2), 496–506 (2014)
30. Q. Zheng, Y. Li, P. Chen, X. Dong, An innovative IPFS-based storage model for blockchain, in *Proceeding of 2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI)* (IEEE, New York, 2018), pp. 704–708

Chapter 7
Data Sharing Incentives with Blockchain

With the development of big data, various industries have put forward demands for data storage and data sharing. In the medical industry, various medical institutions will apply data sharing to medical diagnosis, thus giving rise to third party institutions that can perform pathological diagnosis through machine learning. The current data sharing schemes among multiple participants are constrained by efficiency and security problems, which could be solved by blockchain. No participant in a data-sharing collaboration tends to make an effort without compensation. Existing studies on medical data sharing have rarely concerned about the reasonable incentive mechanism for multi-party cooperation. In this chapter, we aim to achieve trusted data sharing by establishing fair and cooperative patterns, exploring the application of Shapley value in distributing revenues among the data sharing participants. After introducing the design goal and theory of Shapley value used in our method in Sects. 7.1 and 7.2, we derive the topological relationships among the participants and gradually develop the computational process of Shapley value revenue distribution in Sects. 7.3 and 7.4, respectively. In Sect. 7.5.1, we discuss the laws of revenue distribution and the rationality in incentives. Lastly, a summary is drawn in Sect. 7.6.

7.1 Overview

Big data and machine learning technologies have been widely used in the medical field [13, 25], in which a large amount of medical data resources can be transformed into valuable knowledge that helps scientific decision-making [18]. Some artificial intelligence (AI) companies have been able to use medical data (such as medical images) to build predictive models and alliance with medical institutions to provide diagnostic services to patients [7, 22]. Thus, they pay much attention to the problem of medical resource allocation, which depends heavily on the proper collection, management, and utilization of health information [10, 27].

M. Shen et al., *Blockchain: Empowering Secure Data Sharing*,

Generally, as the information collection may be distributed independently in multiple organizations [17, 21], trustworthy data sharing is particularly significant in improving the quality and efficiency of services. When obtaining enemy intelligence, practical intelligence can only be obtained if a considerable price is given [15, 16]. In the same way, for the data owner to provide real and reliable data, a particular incentive mechanism is also needed. The behavior of data sharing and service provision should be profit-driven. However, there is no such research on the incentive mechanisms of data sharing based on blockchain [16, 17].

The existing data sharing schemes in peer-to-peer network and cloud environment can protect data security through encryption [19] and data recovery [4], but cannot completely avoid the risk of a single point of failure and potential attacks for the centralized storage. Azaria et.al [1] proposed MedRec, a decentralized record management system for electronic medical records using blockchain technology, providing a secure access to medical data and comprehensive, immutable log. Xia et.al [3] proposed a blockchain-based data sharing framework that ensures sensitive data sharing by employing the secure cryptographic techniques. They also presented MedShare, a blockchain-based system that solves the problem of medical data sharing among medical big data custodians in a trust-less environment.

The advantages of blockchain in the area of medical information is remarkable, which are especially embodied in the following aspects:

- The distributed architecture makes medical data accesible. Users have the opportunity to hold more shared data from different medical institutions through the blockchain system.
- The features of "de-centralization", "de-trusting" and "tamper-resistant" can solve the problems of information security and data integrity.
- Once uploaded, the medical data is stored on the blockchain permanently, and any medical service provider with corresponding authority can get read directly, leaving out the process of sending repeated data by data owners.
- It is able to promote data transactions and bilateral agreements of data owners and service providers by the application of digital currency via blockchain.

Practically, an efficient incentive mechanism must be exploited to encourage more stakeholders of data sharing to take part in the coalition and to provide reliable data actively. In general transactions, the data owners might be paid all at once, and share their data that is required. The smart contract of Blockchain can provide a reliable platform where revenue distribution is dynamic, and each participant is paid continually as new revenues flow into the coalition from patients. This form of revenue distribution seems more reasonable, as it can encourage participants to improve the quality of services and to provide reliability of data for more profits.

In this chapter, we aim to achieve trusted data sharing by establishing fair and cooperative patterns, exploring the application of Shapley value in distributing revenues among the data sharing participants. We propose a basic business model for medical data sharing and describe a viable trading process based on the Blockchain system. Then we construct a cooperation model for intelligent healthcare service. To the best of our knowledge, it is the first attempt to explore the incentive mechanism of Shapley value revenue distribution in the coalition of medical data sharing. Considering the Blockchain scenario, the incentive schemes we presented not only distribute revenues to data owners and third parties providing services but also treat miners in the Blockchain as the indispensable participants rewarded in the cooperation.

The remainder of this chapter is organized as follows: Sect. 7.2 introduces the theory of Shapley value used in our method. Then, the data sharing business model and cooperation model are represented in Sect. 7.3. Section 7.4 presents an incentive mechanism with revenue distribution among data owners and miners. We discuss the laws of revenue distribution and rationality of the designed models in Sect. 7.5.1. Lastly, a summary is drawn in Sect. 7.6.

7.2 Introduction to Shapley Value Theory

In data sharing, reasonable revenue distribution becomes the part of incentive mechanism to ensure reliable collaboration. Game theory provides a variety of solutions for the revenue distribution, such as the core, the nucleolus, and the Shapley value. The Shapley value, proposed by Lloyd Shapley [14, 23], serves as an appropriate mechanism for a set of players to share revenues generated by the corresponding coalition. In practice, the revenue changes dynamically along with a set of players joining a coalition in a particular order.

The Shapley value is most accompanied by collaboration and sharing scenarios. Some solutions based on Shapley value are suitable for other industries. A lot of efforts have been made to explore the cooperative solution of Internet service providers by using Shapley value by Ma et al. [11]. Cai and Pooch [2] applies Shapley values in wireless mobile ad hoc networks to encourage collaborative work and reward service providers based on contributions. Zheng et al. [26] use the improved Shapley value method to solve the problem of closed-loop supply chain allocation in the third-party reclaim mode.

Shapley values embody the concept of hard work and, like a miner, effectively motivate the active participation of all members [5]. Therefore, when constructing our model, we leverage Shapley value and consider fairness in the distribution to avoid equalitarianism and consume less computing resources.

We consider a set of participants denoted as $\mathscr{N}$. $N = |\mathscr{N}|$ denotes the number of the participants in this set. We call any nonempty subset $S \subseteq \mathscr{N}$ a coalition of the participants. A coalition can generate revenues by sharing medical data and providing medical analysis services through internal cooperation. For each coalition S, we denote $v(\mathscr{N})$ as the worth function, which measures the total revenue produced by the service which all members of this coalition S play a part in. To be specific, $v(\mathscr{N})$ defines as the revenue produced by the whole set of participants

$\mathcal{N}$. Let $SP_i(S)$ denote the income earned by the participant i in coalition S, then we have:

$$v(\mathcal{N}) = \sum_{i \in S} SP_i(S) \tag{7.1}$$

Thus we can measure the contribution of each participant to a coalition through the worth function v, which can be defined as follows.

Definition 1 The marginal contribution of participant i to a coalition $S \subseteq \mathcal{N} \backslash \{i\}$ is defined as $\Delta_i(v, S) = v(S \cup \{i\}) - v(S)$ [28].

In this case, the Shapley value can be computed as the Eq. (7.2) shows, where Π is a set of all $N!$ orderings of $\mathcal{N}$ and $S(\pi, i)$ is the set of players preceding i in the ordering π.

$$\varphi_i(\mathcal{N}, v) = \frac{1}{N!} \sum_{\pi \in \Pi} \Delta_i(v, S(\pi, i)) \quad \forall i \in \mathcal{N} \tag{7.2}$$

Thus, the Shapley value of a player i can be interpreted as the marginal contribution $\Delta_i(v, S)$ where S is the set of players in $\mathcal{N}$ preceding i in a uniformly distributed random orderings. This definition expresses that the numerical change of revenue Δ_i depends on the orderings. However, the Shapley value only depends on the values $\{v(S) : S \subseteq \mathcal{N}\}$. We may simplify the computing process as we only take the revenues of all coalitions into account.

$$\varphi_i(\mathcal{N}, v) = \sum_{s \in S_i} w(|s|)[v(s \cup \{i\}) - v(s)] \quad \forall i \in \mathcal{N} \tag{7.3}$$

where $w(|s|) = \frac{(|\mathcal{N}|-|s|-1)!|s|!}{|\mathcal{N}|!}$ can be considered as the weighting factor and S_i denotes a collection of all subsets of N excluding the player i. Note that the Shapley value defined by Eqs. (7.2) and (7.3) satisfies the property of effectiveness:

$$\sum_{i \in \mathcal{N}} \varphi_i(\mathcal{N}, v) = v(\mathcal{N}) \tag{7.4}$$

The Shapley value mechanism reasonably measures the contribution of each player through computing the increment of revenues as it takes part in a coalition. In this section, we focus on calculating the Shapley value revenue distribution for data sharing participants and analyzing its effect on incenting cooperation.

7.3 Data Sharing Cooperation Model

In this section, We introduce a business model to describe the application of blockchain in medical services and design a cooperation model with miners participating.

7.3.1 Business Model

The current research shows that blockchain technology has a great advantage in the field of e-health for its property of decentralization, tamper proof, and openness. In the case of data sharing, data owners, such as hospitals, publish their medical data via blockchain transactions [9]. The legal third-party research institutions access to the available data to train their respective data analysis models and provide diagnostic services for revenues. In this business scenario, the content providers and receivers have incentives to do the above. We describe the business model in detail, as shown in Fig. 7.1.

Figure 7.1 describes a shared business model based on the blockchain. It explains how to realize data sharing and revenue distribution through information interaction and capital flow. The data owner holds various types of medical data. When a user requests a third party to obtain a pathology report, the third party submits a medical data request to the data owner, and the data owner shares their data on the blockchain network. In general, medical data is encrypted before uploading to protect privacy. After the third party obtains the data and provides services for the users who have already paid, the platform treats the fees paid by the users as revenues and distributes them to the data owners who provide the relevant data and the third parties who provide the services.

We use smart contract to control revenue distribution automatically. The smart contract can be used as a trusted intermediary to perform payment functions and manage capital flows. In the business model, participants continue to get paid as new

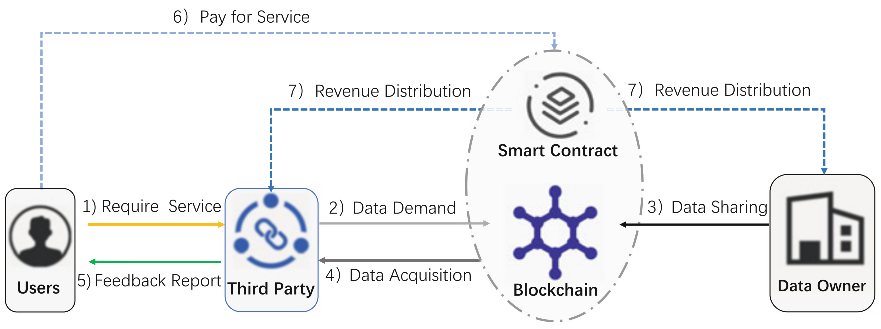


Fig. 7.1 Business model in data sharing

revenue flows into the alliance. This form of revenue distribution seems reasonable because it encourages participants to be held accountable for the outcome of the collaboration.

7.3.2 Cooperation Model

Through the blockchain sharing platform solution, medical data can be securely and efficiently shared. Data owners share their medical data through a blockchain network. After getting the data from the data owner, authorized third-party research institutions train their diagnostic models by applying advanced computing technologies such as big data analysis, machine learning, and data mining to obtain their diagnostic models and obtain benefits by providing diagnostic services.

In a blockchain-based data sharing system, data owners and third parties act as senders and receivers of medical data, respectively. Besides, the miners' recorded data on the blockchain plays an irreplaceable role and contributed to the cooperation. Therefore, we should also design incentive mechanisms to encourage miners to participate in cooperation to promote information exchange.

We categorize participants as three types: third parties, miners and data owners. The participants are defined as $\mathscr{N} = \mathscr{T} \cup \mathscr{M} \cup \mathscr{H}$, where $\mathscr{T} = \{T_1, \cdots, T_{|\mathscr{T}|}\}$ denotes a set of third parties, $\mathscr{M} = \{M_1, \cdots, M_{|\mathscr{M}|}\}$ denotes a set of miners, and $\mathscr{H} = \{H_1, \cdots, H_{|\mathscr{H}|}\}$ denotes a set of data owners. We denote $\mathscr{R} = \{1, \cdots, R\}$ as the set of services provided by $\mathscr{T}$. The elements in $\mathscr{R}$ represent the sequence number of services and R is the number of service types. The set of services provided by T_i is denoted by $R_i \subseteq \mathscr{R}$. Similarly, we denote $\mathscr{D} = \{1, \cdots, D\}$ as a set of different types of medical data. The elements in $\mathscr{D}$ are corresponding to those in $\mathscr{R}$, thus we have $\mathscr{D} = \mathscr{R}$, and the number of data types D satisfies $D = R$. The types a data owner H_j shares are in $D_j \subseteq \mathscr{D}$. When T_i provides the service $r \in R_i$ and H_j shares the data of type $d \in D_j$ that satisfies $d = r$, it means that H_j shares type d data to T_i. There may be more than one type of data shared by H_j matching the request of T_i. Such relationship is represented by a connection, which goes through a group of miners.

Figure 7.2 shows a cooperation model where $|\mathscr{T}| = 3$, $|\mathscr{H}| = 4$, $|\mathscr{M}| = m$ and $\mathscr{R} = \mathscr{D} = \{a, b, c\}$. All the miners in group $\mathscr{M}$ act the same functionally and only one works at a time. The data owners in Fig. 7.2 contain a total of three types of data: a, b and c. However, the type of data contained by each owner is different, and the type of data required from users is different. For example, the service requirement SP_2 aims to two kinds of data: b and c. Thus T_2 receives the data from H_2, H_3, and H_4, which contains these two kinds of data and joins the alliance of data sharing, and each sharing is processed by a miner M.

The revenue of the system consists of multiple payments. We define the payment SP_i^r as shown in Eq. (7.5).

$$SP_i^r = \alpha_r x_i^r \quad \forall T_i \in \mathscr{T} \tag{7.5}$$

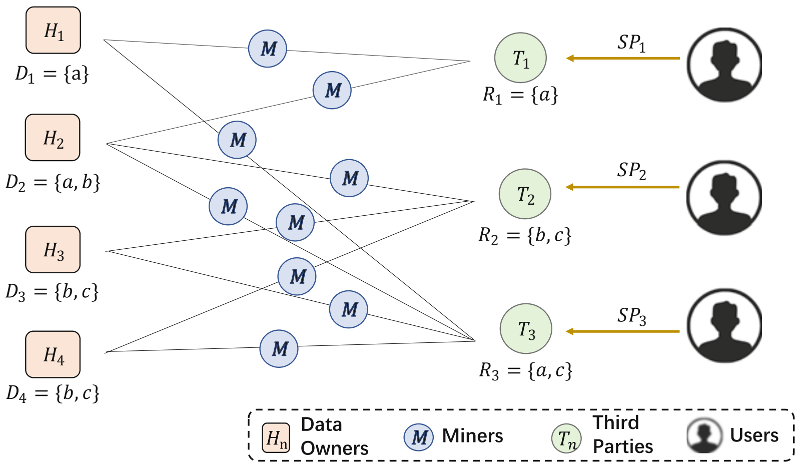


Fig. 7.2 Cooperation model based on blockchain

where α_r is the charge for each service r and x_i^r denotes the number of times that third party T_i provides service r in a period time (equals zero if T_i does not provide service r during the period or $r \notin R_i$). The payment SP_i^r is the periodical charge produced by T_i providing service r. If service r is out of the business of T_i, we have $SP_i^r = 0$ for $x_i^r = 0$. We assume that all third parties charge the same amount for the same service so that they compete fairly in business.

7.4 Incentive Mechanism and Revenue Distribution

In this section, we explore the distribution of Shapley value gains among data sharing participants, taking into account the contribution differences of data owners and the importance of the roles in the cooperative alliance, which also includes specific rules regarding the distribution of miners' revenue.

7.4.1 Revenue Distribution Among Data Owners

In terms of the quality of service, the performance of prediction models can be evaluated by these commonly used metrics: Precision, Recall and F1 score [19, 20]. However, there is often taken F1 score, which quantifies the tradeoff between Precision(P) and Recall(R), as the overriding evaluation parameter ($F1 = 2\frac{P*R}{P+R}$). In the scenario where the third party provide only one service, we infer the contribution of each data owner to the revenue from the F1 score of the data-trained predictive model.

We assume that the data owner $H_i \in \mathscr{H}$ can provide data that satisfies the service r, i.e., $D_i = \{r\}$. The total revenue belonging to the group of data owners is represented by $v(\mathscr{H})$. Assuming that all the data owners have positive impact on the alliance. We mainly target the arbitrary data owners in $\mathscr{H}$ and observe the F1 score of the prediction model trained on the data provided by them.

We leverage the equation of $\Delta_i(F1, s) = F1(s \cup \{i\}) - F1(s)$ to quantify the contribution of the F1-score of each participant, where $F1(s)$ is the F1 score of the prediction model trained by the coalition s from data owners. It is similar to the marginal contribution $\Delta_i(v, S)$ defined in the Eq. (7.2).

For any regular system $(\mathscr{H}, v)$, we calculate the Shapley value combining with F1 score as shown in Eq. (7.6).

$$\varphi_i(\mathscr{H}, v) = \sum_{s \in S_i} w(|s|) \frac{\Delta_i(F1, s)}{F1(\mathscr{H})} v(\mathscr{H}) \quad \forall i \in \mathscr{H}, \tag{7.6}$$

where $S_i = \{s | s \subseteq \mathscr{H}/\{i\}\}$ represents the set of all data owner subsets of $\mathscr{H}$, excluding participant i. Since we use a unary method to divide $F1(\mathscr{H})$ in Eq. (7.6), the attributes represented in the Eq. (7.4) are satisfied, i.e., $\sum_{i \in \mathscr{H}} \varphi_i(\mathscr{H}, v) = v(\mathscr{H})$.

Theorem 1 shows how to distribute revenue for data owners through the Shapley value. We define φ_{H_i} as the Shapley value allocated to the data owner H_i.

Theorem 1 (Shapley Value for Data Owners) *We assume that the data owner $\mathscr{H}$ shares data for the same service. We define $S^i_{\mathscr{H}} = \{\mathscr{H}' | \mathscr{H}' \subseteq \mathscr{H}/\{H_i\}\}$ as a set of all subsets of $\mathscr{H}$, excluding the data owner H_i. Based on the marginal contribution to the F1-score of the prediction model, the Shapley value revenue of each data owner is defined as the Eq. (7.7).*

$$\varphi_{H_i}(\mathscr{H}, v) = \varphi^i_H(\mathscr{H}) v(\mathscr{H}) \quad \forall H_i \in \mathscr{H}, \tag{7.7}$$

where the normalized Shapley values $\{\varphi^1_H, \cdots, \varphi^{|\mathscr{H}|}_H\}$ are represented as Eq. (7.8) shows.

$$\varphi^i_H(\mathscr{H}) = \sum_{\mathscr{H}' \in S^i_{\mathscr{H}}} w(|\mathscr{H}'|) \frac{\Delta_{H_i}(F1, \mathscr{H}')}{F1(\mathscr{H})} \quad \forall H_i \in \mathscr{H}. \tag{7.8}$$

The normalized Shapley value φ^i_H can be considered as the percentage of total revenue $v(\mathscr{H})$ received by the data owner H_i. It is essentially an accumulation of the marginal contributions $\Delta_{H_i}(F1, \mathscr{H}')$.

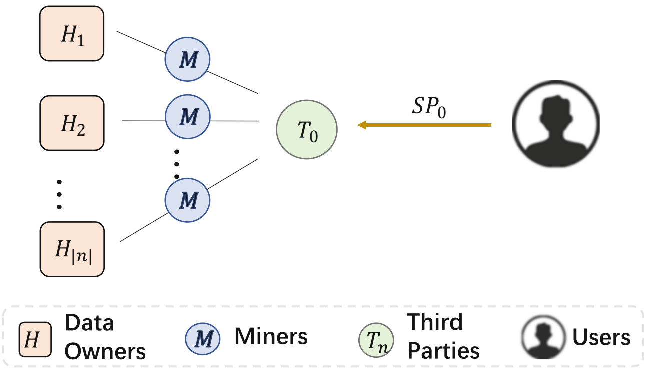


Fig. 7.3 Simple cooperation model

7.4.2 Revenue Distribution in Cooperation Model

Before calculating the revenue distribution in the multi-role cooperation model, we first build a simple model that provides a single service, as shown in Fig. 7.3. The simple model contains three groups of participants, but only one service requirement. The data owner and a third party form a complete data flow in which all connections go through the miner.

Under the simple model, the coalition $\mathscr{N} = \mathscr{H} \cup \mathscr{M} \cup \mathscr{T}$ provides service r and $|\mathscr{T}| = 1$. The revenue comes from the service generated by the only third party T_0, i.e., $v(\mathscr{N}) = \alpha_r x_0^r$. We define φ_{H_i}, φ_{M_k} and φ_{T_0} as the Shapley value revenues distributed to H_i, M_k and T_0 respectively.

Theorem 2 (Shapley Value For Simple Model) *The coalition of $\mathscr{T}$ containing a single third party T_0, a set of miners $\mathscr{M}$, and a set of data owners, $\mathscr{H}$. For the data owners, we define $S_H = \{\mathscr{H}'|\mathscr{H}' \subseteq \mathscr{H}, \mathscr{H}' \neq \emptyset\}$ as a set of all non-empty subsets of $\mathscr{H}$ and $S_H^i = \{\mathscr{H}'|\mathscr{H}' \subseteq \mathscr{H}/\{H_i\}\}$ as a set of all subsets of $\mathscr{H}$, excluding H_i. For the miners, we define $S_M = \{\mathscr{M}'|\mathscr{M}' \subseteq \mathscr{M}, \mathscr{M}' \neq \emptyset\}$ as a set of all non-empty subsets of $\mathscr{M}$, and P_k as the probability of miner M_k mining a block. Assuming all data owners share the data of the service type d, which satisfies $d = r$. The Shapley value revenue for each participant is defined in Eq. (7.9).*

$$\begin{aligned}
&\varphi_{H_i}(\mathscr{H}, \mathscr{M}, \mathscr{T}) = \varphi_H^i(\mathscr{H}, \mathscr{M}, \mathscr{T})v(\mathscr{N}) && \forall H_i \in \mathscr{H},\\
&\varphi_{M_k}(\mathscr{H}, \mathscr{M}, \mathscr{T}) = P_k|\mathscr{M}|\varphi_M(\mathscr{H}, \mathscr{M}, \mathscr{T})v(\mathscr{N}) && \forall M_k \in \mathscr{M}, \qquad (7.9)\\
&\varphi_{T_0}(\mathscr{H}, \mathscr{M}, \mathscr{T}) = \varphi_T(\mathscr{H}, \mathscr{M}, \mathscr{T})v(\mathscr{N}),
\end{aligned}$$

where the normalized Shapley values φ_H^i, φ_M^k and φ_T are represented as follows.

$$\begin{aligned}
\varphi_H^i(\mathscr{H},\mathscr{M},\mathscr{T}) &= \sum_{\mathscr{H}'\in S_H^i}\sum_{m=1}^{|\mathscr{M}|}\binom{|\mathscr{M}|}{m} w(|\mathscr{H}'|+m+1)\frac{\Delta_{H_i}(F1,\mathscr{H}')}{F1(\mathscr{H})},\\
\varphi_M(\mathscr{H},\mathscr{M},\mathscr{T}) &= \sum_{\mathscr{H}'\in S_H} w(|\mathscr{H}'|+1)\frac{F1(\mathscr{H}')}{F1(\mathscr{H})},\\
\varphi_T(\mathscr{H},\mathscr{M},\mathscr{T}) &= \sum_{\mathscr{H}'\in S_H}\sum_{m=1}^{|\mathscr{M}|}\binom{|\mathscr{M}|}{m} w(|\mathscr{H}'|+m)\frac{F1(\mathscr{H}')}{F1(\mathscr{H})}.
\end{aligned} \tag{7.10}$$

We consider $P_k|\mathscr{M}|$ as the "Expectation" of the weight of miner M_k of the φ_{M_k}. Then, we derive the normalized Shapley value from Eq. (7.6). Although third parties and miners will not cause changes in F1 values, cooperation is necessary. Calculating the score of F1 is only meaningful when there are three types of participants in the alliance, otherwise the value of F1 is 0. Therefore, we remove all zero-value marginal contributions and derive the above Eq. (7.10).

We aggregate the results of all included simple models to distribute the total revenue in the cooperation model as shown in Theorem 3, where the φ_{H_i}, φ_{M_k} and φ_{T_j} denote the Shapley value revenues distributed to H_i, M_k and T_j respectively. The revenue of the cooperation model is denoted as Eq. (7.11).

$$v(\mathscr{N}) = \sum_{r=1}^{|\mathscr{R}|}\sum_{j=1}^{|\mathscr{T}|} SP_j^r. \tag{7.11}$$

Theorem 3 (Shapley Value For Cooperation Model) *We assume that in a set of cooperative alliances, the data owner $\mathscr{H}$, miner $\mathscr{M}$, and third party $\mathscr{T}$ are included. The third party T_j is responsible for providing a set of services $R_j \subseteq \mathscr{R}$, and the data owner H_i shares the medical data $D_i \subseteq \mathscr{D}$ required by the service. The data owner H_i connects to the third-party T_j only if $R_j \cap D_i \neq \emptyset$ and all connections pass through the miner group. The Shapley value of each participant is expressed as the Eq. (7.12).*

$$\begin{aligned}
\varphi_{H_i}(\mathscr{H},\mathscr{M},\mathscr{T}) &= \sum_{r\in D_i}\sum_{j=1}^{|\mathscr{T}|}\varphi_H^i(\mathscr{H}_r,\mathscr{M},\{T_j\})SP_j^r,\\
\varphi_{M_k}(\mathscr{H},\mathscr{M},\mathscr{T}) &= P_k|\mathscr{M}|\sum_{r\in\mathscr{D}}\sum_{j=1}^{|\mathscr{T}|}\varphi_M(\mathscr{H}_r,\mathscr{M},\{T_j\})SP_j^r,\\
\varphi_{T_j}(\mathscr{H},\mathscr{M},\mathscr{T}) &= \sum_{r\in R_j}\varphi_T(\mathscr{H},\mathscr{M},\{T_j\})SP_j^r,
\end{aligned} \tag{7.12}$$

where φ_H^i, φ_M and φ_T are defined in Theorem 2. The revenue SP_j^r generated by the third party T_j is supposed to be proportionally allocated to the miners and paid to the data owner who provides data of type d, where $d = r$.

7.4.3 Mining Revenue

In this section, we focus on the revenue distribution mechanism among miners who use different consensus in the blockchain. Assuming that the data shared on the blockchain is large enough, the revenue allocated to each miner can approach the ideal value in practice. In the following, we discuss several of the most common consensus mechanisms.

The consensus is a mechanism that achieves Blockchain system reliability in the presence of several faulty processes [12]. The users distributed in the network do not need to trust the other side of transactions, as well as a central authority. As long as the public protocols of blockchain are widely acknowledged, all participants will comply with predetermined rules voluntarily and honestly.

Under the consensus, all the responsible nodes determine the validity of each record and keep the same ledger. The functions of consensus are ensuring the authenticity and consistency of the blockchain system. Some consensuses develop mining mechanisms to incent participant nodes to keep the records legal, which generates rewards, such as PoW(Proof of Work), PoS(Proof of Stake), etc [24]. Here, mining is broadly defined as a behavior of striving for recording right.

- Proof of Work(PoW)

  Under the PoW mechanism, miners compete for record blocks through a large number of calculations [8]. $Hashrate$ is a unit of $hash/s$, which is used to calculate the capacity of miners. In our method, we define c_k as the hashrate of miner M_k. Assuming that the proportion of each miner's revenue is equal to the probability of mining a block. Therefore, the revenue ratio can be expressed as the formula (7.13).

$$P_k = c_k / \sum_{i=1}^{|\mathscr{M}|} c_i \quad \forall M_k \in \mathscr{M}. \tag{7.13}$$

- Proof of Stake(PoS)

  The PoS mechanism is also used to record blocks. Under this rule, miners holding more shares are more likely to get rights. We assume that a miner M_k holds w_k tokens, and the coinage on this point is a_k, so the share of this miner is $w_k a_k$. However, the coinage will be reset when the miner wins a competition. Assuming the interest rate is I, when a miner mines the current block, he will get w_k tokens, and a_k days of coinage will be paid $w_k a_k I/365$ tokens. For every miner, the growth of coinage is synchronous. Therefore, the miner's share increase is proportional to the number of their tokens. The formula (7.14) is used to infer the revenue ratio of M_k for each miner.

$$P_k = w_k / \sum_{i=1}^{|\mathscr{M}|} w_i \quad \forall M_k \in \mathscr{M}. \tag{7.14}$$

- Practical Byzantine Fault Tolerance(PBFT)

  The PBFT algorithm is used to solve the Byzantine problem, which is also a consensus mechanism and is applicable to the consortium blockchain [6]. Under PBFT algorithm, nodes take turns acting as leader nodes, sending current messages to all other nodes. After synchronizing messages between other nodes, new blocks will be generated. Assuming that all nodes have the same chance to become leader nodes, we consider that all normal nodes have equal pay, excluding wrong nodes and malicious nodes. Suppose that n blocks are generated in a cycle, we define $A_k \subseteq \{1, \cdots, n\}$ as the set of the sequence number of the round that makes node M_k a healthy node. We also define q_i as the number of normal nodes in the ith round, as shown in Eq. (7.15). Thus the revenue ratio of each node is defined as the Eq. (7.16).

$$q_i = \sum_{k=1}^{|\mathscr{M}|} 1_{\{i \in A_k\}} \quad \forall i \in \{1, \cdots, n\}, \tag{7.15}$$

$$P_k = \frac{1}{n} \sum_{i \in A_k} \frac{1}{q_i} \quad \forall M_k \in \mathscr{M}. \tag{7.16}$$

Our work is to explore the optimal revenue distribution scheme under various consensus mechanisms. Other consensus mechanisms not mentioned above can also be applied to the scenarios as well, which will be studied in the future.

7.5 Performance Evaluation

7.5.1 Incentive Effect

In this section, we discuss the laws of revenue distribution under the models developed in the previous section and explore the incentive effect and rationality of the incentive mechanism.

Based on the previous introduction, we mainly infer the contribution of each data owner to the revenue from the F1 score of the third-party prediction model. We explore the impact of the contribution of the data owner on the revenue distribution by fixing the role size of each cooperative group. We change the F1 score of a prediction model, and set a unit model that $|\mathscr{H}| = 2$, $|\mathscr{M}| = 5$ and $|\mathscr{T}| = 1$.

We define $\phi_{\mathscr{H}} = \sum_{H_i \in \mathscr{H}} \varphi_{H_i}$ as the aggregate Shapley value for the group of data owners,$\phi_{\mathscr{M}} = \sum_{M_k \in \mathscr{M}} \varphi_{M_k}$ as the aggregate Shapley value for the group of miners and $\phi_{\mathscr{T}} = \varphi_{T_0}$ as the aggregate Shapley value for the group of third party.

We first consider the situation that the two data owners contribute equally to F1-score. Figure 7.4 eliminates the differences between the two data owners by synchronize their F1-score ($F1(\{H_1\}) = F1(\{H_2\}) = F_0$). Figures 7.5, 7.6, and 7.7 illustrate how the revenue of $\phi_{\mathscr{H}}$, $\phi_{\mathscr{M}}$ and $\phi_{\mathscr{T}}$ change with the overall contribution

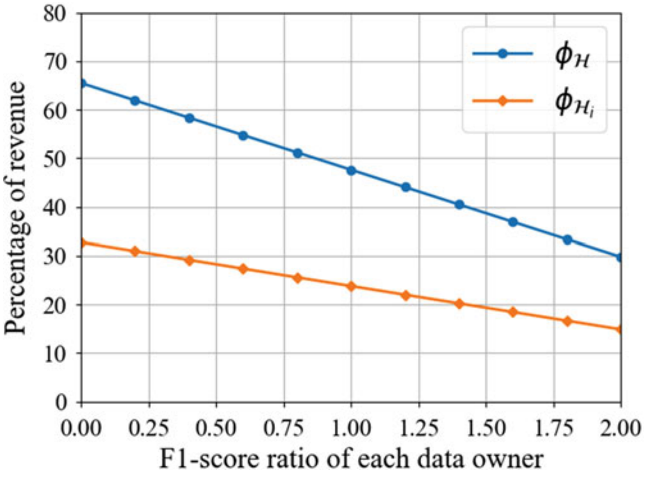


Fig. 7.4 The Shapley value revenue with different F1 score in data owners and unit data owner

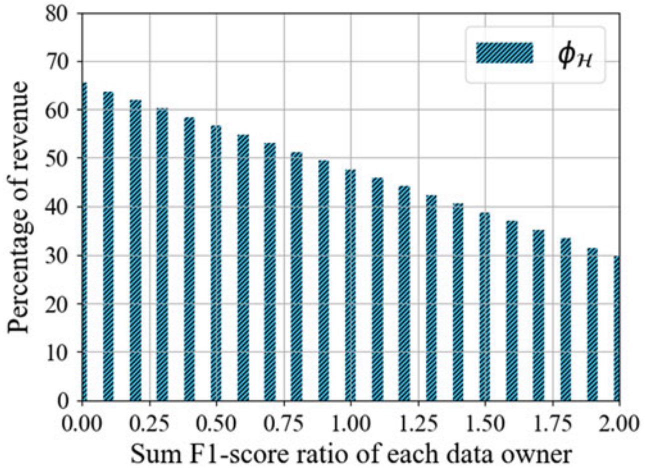


Fig. 7.5 The Shapley value revenue under the simple model with different F1 scores in data owners

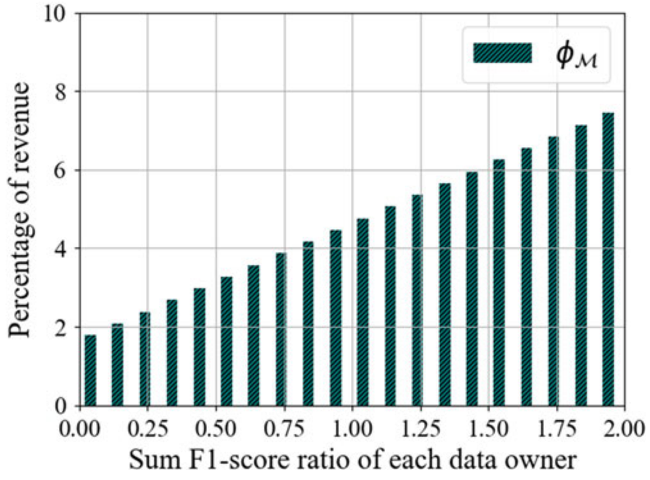


Fig. 7.6 The Shapley value revenue under the simple model with different F1 scores in miners

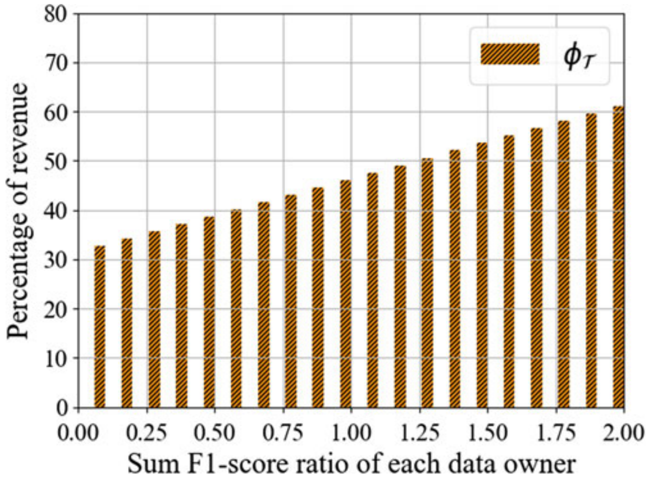


Fig. 7.7 The Shapley value revenue under the simple model with different F1 scores in third parties

of the shared data changing. We observe that the Shapley value of data owners decreases and that of miners and the third party increase when the F1-score ratio $F_0/F(\mathscr{H})$ increases.

As for a general situation that the F1-score of each data owner changes diversely, we find that the aggregate Shapley value revenues of each group are only related to the sum of F1-scores, i.e., $F1(\{H_1\}) + F1(\{H_2\})$, which is illustrated in Figs. 7.4 and 7.5. The observation is the same in the case of more data owners in the coalition.

The result of revenue distribution will not change when the sum of F1-score ratios of all subset of data owners remains unchanged.

We understand trends in revenue distribution in terms of the importance of cooperation. When F1 scores are low, collaboration dramatically improves the quality of service for data owners. Therefore, the involvement of each data owner has a significant impact on the system, which means that the data owner should receive more revenue.

The distribution of revenue under several roles follows the pattern in each unit model. In general, data owners should be encouraged to participate in more sharing patterns, encouraging them to increase their income by providing real, valid data, as shown in the Shapley value formula element of Theory 2.

7.5.2 Incentives Rationality

In order to verify the rationality of the incentive mechanism and the rationality of participants' willingness to join the cooperative alliance, we analyze the changes in the revenue distribution of the three roles by setting up experiments under the condition that the number of data owners changed. We fix the number of miners ($|\mathscr{M}| = 5$) and set all F1-scores to the same value. Miners can only benefit from participation. The results show in Fig. 7.8, which verify that when the number of data owners increases, so does the benefit to the third party, thus verifies the motivation of the third party to participate in the validation.

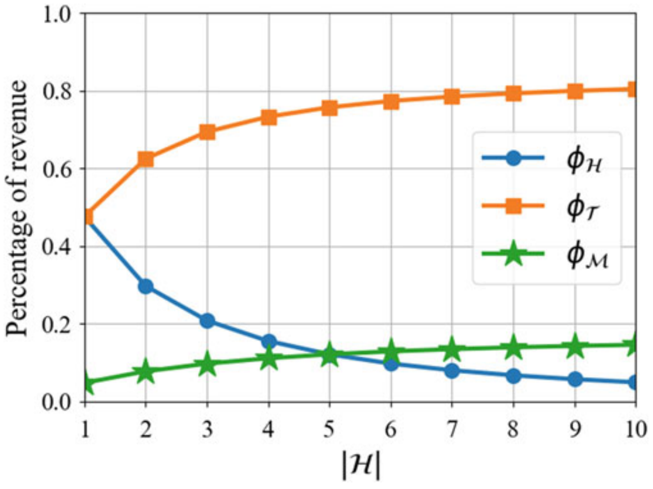


Fig. 7.8 The Shapley value revenue under the simple model with a variable size of the data owner group

The motivation for third parties to cooperate with more data owners is twofold. On the one hand, valuable data contributes to the accuracy of their prediction models, which may attract more customers. On the other hand, third parties can get more revenues from the coalition with more data owners, which can be studied in Fig. 7.8. It is necessary for miners to join in coalition for benefits. They are responsible for recording the shared data and making it credible. There is no doubt that the stakeholders, third parties are motivated to participate in a coalition with more miners for the consideration of information security.

The proposed model solves the problem of the permission of participants through the consortium blockchain and ensures that the shared data can be used safely and reasonably. We design an incentive mechanism suitable for the dynamic distribution of benefits among multiple clouds, which can encourage participants of the sharing consortium to provide reliable data and realize the collaborative sharing mechanism.

However, the scheme has the problem of transparent propagation, and the data shared in the platform is not anonymous or encrypted. We consider that some patients are eager to hide their inspection reports or other sensitive data in the medical data sharing situation. It is still a challenge to judge contributions when roles or data are unknown. Therefore, in the future work, we will consider and optimize the design of this aspect.

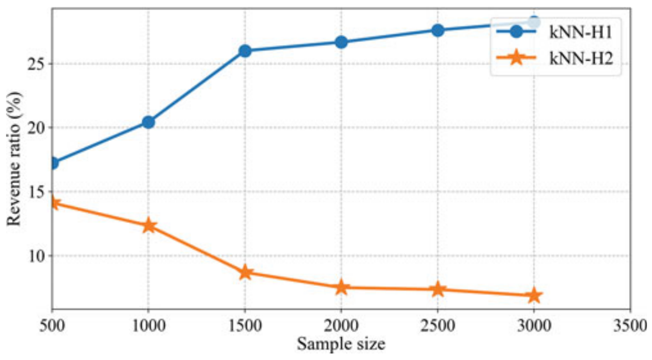
7.5.3 Data Effect on Participants

To explore the impact of medical data on the revenue among participants, we use several intelligent algorithms commonly used in third-party data processing agencies in healthcare scenarios to train prediction models for evaluation, including Support Vector Machines (SVM), Random Forest (RF) And K Nearest Neighbors (KNN). Among them, we use a data set on thyroid disease published by the Garavan Medical Institute in Sydney, Australia. We select 4427 data samples from the data set as the test set to measure the accuracy of the prediction model, and from the other 723 samples as the test set.

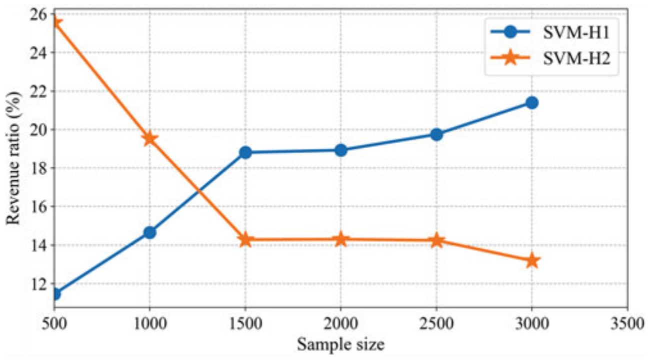
Figure 7.9 shows the relationship between the number of samples of the medical data shared by the data owners and the benefits obtained.

We take the case of the number of data owners $|H| = 2$ in a medical data sharing scene. Supposing there is a data owner set $H = \{H_1, H_2\}$. We select samples for all the training set samples for H_1 and H_2 to construct the training set. The data shared by the data owner H_2 contains 1000 samples, and the number of medical data samples divided by the data owner H_1 is 500, 1000, 1500, 2000, 2500, 3000, each of the values is simulated 100 times, and the revenue distribution results of the data owner under different sample sizes are calculated.

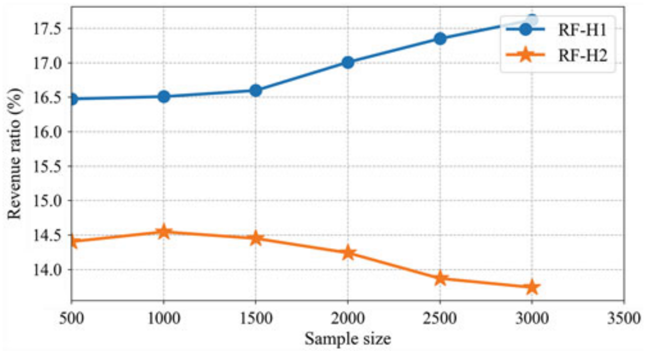
In this setting, we use three algorithms to build prediction models on the training set under various combinations and calculate the data owners' data when sharing different numbers of thyroid disease samples. The F1-index of the predictive model



(a)



(b)



(c)

Fig. 7.9 Relationship between revenue and sample size. (**a**) KNN. (**b**) SVM. (**c**) RF

obtained by standardized Shapley values and model learning on data shared in the alliance.

According to Fig. 7.9, it can be seen that under the same conditions, when the data owner H_1 increases the number of shared data samples, it increases its revenue and reduce the revenue of the data owner H_2. Goals are in line with expected results. When the data owner increases the number of shared data samples, the prediction model obtained by the learning algorithm has a better prediction effect. It can be considered that the data owner's contribution has increased, so he should get higher returns as compensation.

With other conditions unchanged, increasing the number of data samples shared by the data owner, H_1 increases their revenue, which is consistent with the design goals and expected effects of the incentive mechanism.

When the data owner increases the number of shared data samples, the prediction model obtained by the learning algorithm has a better prediction effect. It can be argued that the contribution of the data owner is improved and a higher revenue should be received as a reward. This proves that the incentive mechanism proposed in our method can motivate users with medical data to actively share more data.

7.6 Summary

In this section, we propose a blockchain-based Shapley value solution to stimulate cooperation in medical data sharing. We build a business model to describe the application of blockchain in medical services and design a cooperation model with miners participating. It is the first attempt to explore an incentive mechanism based on blockchain and Shapley values. Finally, we also prove that the solution can effectively motivate participants to share reliable data. Our method applies not only to medical data sharing but also to other data sharing scenarios. In this solution, there is no algorithm designed to support confidentiality for private data. In the future, we will leverage the encryption or other methods to build a more secure sharing model, and investigate other factors affecting contribution and distribution to achieve the evaluation of incentive mechanisms in practice effect.

References

1. A. Azaria, A. Ekblaw, T. Vieira, A. Lippman, Medrec using blockchain for medical data access and permission management, in *International Conference on Open and Big Data* (2016), pp. 25–30
2. J. Cai, U. Pooch, Allocate fair payoff for cooperation in wireless ad hoc networks using shapley value, in *Proceedings of 18th International Parallel and Distributed Processing Symposium, 2004* (IEEE, Piscataway, 2004), p. 219
3. F. Gao, L. Zhu, M. Shen, K. Sharif, Z. Wan, K. Ren, A blockchain-based privacy-preserving payment mechanism for vehicle-to-grid networks. IEEE Netw. **32**(6), 184–192 (2018)

4. R.A. Haraty, M. Zbib, M. Masud, Data damage assessment and recovery algorithm from malicious attacks in healthcare data sharing systems. Peer-to-Peer Netw. Appl. **9**(5), 812–823 (2016)
5. W. Hu, Y. Hou, L. Tian, Y. Li, A novel profit-allocation strategy for SDN enterprises. Enterp. Inf. Syst. **11**(1), 4–16 (2017)
6. W. Hu, Y. Hu, W. Yao, H. Li, A blockchain-based byzantine consensus algorithm for information authentication of the internet of vehicles. IEEE Access **7**, 139703–139711 (2019)
7. X. Huang, X. Du, Achieving big data privacy via hybrid cloud, in *2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)* (2014), pp. 512–517
8. G. Kumar, R. Saha, M.K. Rai, R. Thomas, T. Kim, Proof-of-work consensus approach in blockchain technology for cloud and fog computing using maximization-factorization statistics. IEEE Internet Things J. **6**(4), 6835–6842 (2019)
9. J. Li, Y. Zhang, X. Chen, Y. Xiang, Secure attribute-based data sharing for resource-limited users in cloud computing. Comput. Secur. **72**, 1–12 (2018)
10. Y. Li, L. Zhu, M. Shen, F. Gao, B. Zheng, X. Du, S. Liu, S. Yin, *CloudShare: Towards a Cost-Efficient and Privacy-Preserving Alliance Cloud Using Permissioned Blockchains* (2018), pp. 339–352
11. R.T. Ma, D.M. Chiu, J.C. Lui, V. Misra, D. Rubenstein, On cooperative settlement between content, transit, and eyeball internet service providers. IEEE/ACM Trans. Netw. **19**(3), 802–815 (2011)
12. S. Pahlajani, A. Kshirsagar, V. Pachghare, Survey on private blockchain consensus algorithms, in *Proceeding of 2019 1st International Conference on Innovations in Information and Communication Technology (ICIICT)* (2019), pp. 1–6
13. F. Rahman, M. Slepian, A. Mitra, A novel big-data processing framwork for healthcare applications: big-data-healthcare-in-a-box, in *IEEE International Conference on Big Data* (2017), pp. 3548–3555
14. A.E. Roth, L.S. Shapley, The Shapley value : essays in honor of Lloyd S. Shapley. Econ. J. **101**(406), 235–264 (1988)
15. M. Shen, Y. Deng, L. Zhu, X. Du, N. Guizani, Privacy-preserving image retrieval for medical IoT systems: a blockchain-based approach. IEEE Netw. **33**(5), 27–33 (2019)
16. M. Shen, J. Duan, L. Zhu, J. Zhang, X. Du, M. Guizani, Blockchain-based incentives for secure and collaborative data sharing in multiple clouds. IEEE J. Sel. Areas Commun. **38**(6), 1229–1241 (2020)
17. M. Shen, H. Liu, L. Zhu, K. Xu, H. Yu, X. Du, M. Guizani, Blockchain-assisted secure device authentication for cross-domain industrial IoT. IEEE J. Sel. Areas Commun. **38**(5), 942–954 (2020)
18. M. Shen, Y. Liu, L. Zhu, K. Xu, X. Du, N. Guizani, Optimizing feature selection for efficient encrypted traffic classification: a systematic approach. IEEE Netw. (2020). Accepted. https://doi.org/10.1109/MNET.011.1900366
19. M. Shen, B. Ma, L. Zhu, R. Mijumbi, X. Du, J. Hu, Cloud-based approximate constrained shortest distance queries over encrypted graphs with privacy protection. IEEE Trans. Inf. Forensics Secur. **13**(4), 940–953 (2018)
20. M. Shen, X. Tang, L. Zhu, X. Du, M. Guizani, Privacy-preserving support vector machine training over blockchain-based encrypted IoT data in smart cities. IEEE Internet Things J. **6**(5), 7702–7712 (2019)
21. M. Shen, J. Zhang, L. Zhu, K. Xu, X. Tang, Secure svm training over vertically-partitioned datasets using consortium blockchain for vehicular social networks. IEEE Trans. Veh. Technol. **69**(6), 5773–5783 (2020)
22. A.K. Shrestha, J. Vassileva, Blockchain-based research data sharing framework for incentivizing the data owners, in *Blockchain—ICBC 2018* ed. by S. Chen, H. Wang, L.-J. Zhang (Springer, Cham, 2018), pp. 259–266
23. E. Winter, Chapter 53 the shapley value. Handbook Game Theory Econ. Appl. **3**(02), 2025–2054 (2002)

24. F. Yang, W. Zhou, Q. Wu, R. Long, N.N. Xiong, M. Zhou, Delegated proof of stake with downgrade: a secure and efficient blockchain consensus algorithm with downgrade mechanism. IEEE Access **7**, 118541–118555 (2019)
25. C. Zhang, L. Zhu, X. Chang, R. Lu, PPDP: an efficient and privacy-preserving disease prediction scheme in cloud-based e-Healthcare system. Futur. Gener. Comput. Syst. **79**, 16–25 (2017)
26. Y. Zheng, S. Zhang, X. Chen, F. Liu, Application of modified shapley value in gains allocation of closed-loop supply chain under third-party reclaim. Energy Procedia **5**, 980–984 (2011)
27. B.-K. Zheng, L.-H. Zhu, M. Shen, F. Gao, C. Zhang, Y.-D. Li, J. Yang, Scalable and privacy-preserving data sharing based on blockchain. J. Comput. Sci. Technol. **33**(3), 557–567 (2018)
28. L. Zhu, H. Dong, M. Shen, K. Gai, An incentive mechanism using shapley value for blockchain-based medical data sharing, in *2019 IEEE 5th International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing, (HPSC) and IEEE International Conference on Intelligent Data and Security (IDS)*, pp. 113–118 (2019)

Appendix A
How to Develop Smart Contract

A smart contract is a piece of code deployed on a Shared, copied ledger that can maintain its status, control its assets and respond to incoming external information or assets. It can process information, receive, store and send value.

An intelligent contract program is more than just a computer program that can be executed automatically. It is more like a participant in the system. Think of it as a trustworthy person. As Fig. A.1, once a smart contract is established, it can be executed automatically without mediation, and no one can stop it from running.

A.1 History

In the 1990s, computer scientists engaged in research on digital contracts and currency, Nick Saab (Nick complained first proposed the "smart" contract, the claim its commitment to the existing contract law and business practices related to transfer to the Internet, can only make the stranger through the Internet can be achieved before offline business activities, and achieve the real full e-commerce. Despite its benefits, the idea of smart contracts has languished—mainly in the absence of blockchains that would allow it to work.

It wasn't until 2008 that the first cryptocurrency, bitcoin, was introduced, along with modern blockchain technology. Blockchain originally emerged as the underlying technology of bitcoin, and various blockchain forks led to great changes. Smart contracts were still not integrated into the bitcoin blockchain network in 2008, but there brought it to the surface 5 years later. Since then, various forms of smart contracts have emerged, among which there smart contracts are the most widely used.

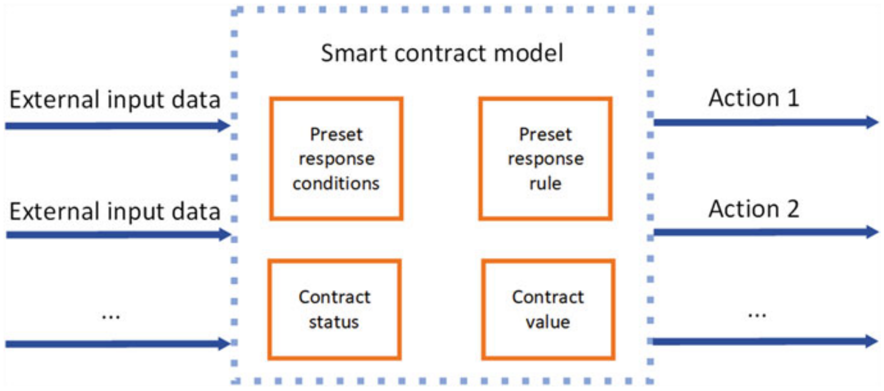


Fig. A.1 Smart contract model

A.2 Development Platform and Language

More than 40 platforms already support smart contracts. Many languages can support the development of smart contracts: solidity, C/C++, Golang, Java, Nodejs, etc.

A.2.1 Ethereum

Ethereum was one of the first platforms to introduce the concept of smart contracts into the blockchain and has received the greatest support from the developer community. It claims to have implemented Turing's complete intelligent contract platform. The contract code is executed by each miner in the Ethereum network on the EVM (Ethereum Virtual Machine). It is the most widely used platform for blockchain projects.

The platform writes smart contracts in solidity's language. Solidity achieves turning-like completeness quite well but compared with modern development languages such as C++, C#, and Python, solidity has no advantage. If writing smart contracts is not efficient, the implementation costs are high.

A.2.2 Enterprise Operation System

The EOS.IO based blockchain USES Web Assembly (WASM) to execute the application code provided by the developer. WASM is compiled from C/C++ programs using LLVM and Clang, which means that users need C/C++ development skills in deploying blockchain applications. Although EOS.IO can be developed using C, the EOS.IO C++ API is recommended. These APIs provide greater type safety and are easier to read.

To improve the speed of transactions, EOS has made a series of optimizations, including the use of dPoS as a consensus mechanism, parallel execution, staged execution, etc. EOS was considered a good alternative to Ethereum due to its high scalability, no transaction fees, and the use of C++ as a smart contract language. But EOS is still not widely used, and many of its advantages and disadvantages as a smart contract platform have yet to be tested.

A.2.3 HyperLedger Fabric

Hyperledger Fabric is one of the Hyperledger projects led by the Linux Foundation. It is the first distributed ledger platform for enterprise application scenarios, with contributions from many technologies and financial giants including IBM, Intel, Cisco, DAH, jp Morgan chase, R3, etc. It has received extensive attention and development in banking, supply chain, and other fields, and now has more than 200 enterprise members.

Hyperledger Fabric is a distributed book solution platform, the goal is to achieve a common permissions block Chain (Permissioned Chain) of the underlying infrastructure, in order to apply to different occasions, adopts the modular architecture to provide swappable and extensible components, including the consensus algorithm, encryption, security, digital assets, intelligent contracts, and identity authentication, etc.

Hyperledger Fabric itself is written in the Go language, so its smart contracts also support the Go language.

A.2.4 Quorum

Quorum is an enterprise-class blockchain platform launched by J. P. Morgan. It is an alliance chain that is based on Ethereum and offers additional services. The main difference between Quorum and Ethereum is that it provides Transaction and Contract privatization; Provides a variety of consensus mechanisms; It has network-based and node-based access management and higher performance than Ethereum.

Similarly, Quorum, which also writes smart contracts in solidity's language, retains the gas mechanism in Ethereum but reduces the price of gas to zero, which reduces the transaction cost to zero while taking advantage of the security features provided by gas restrictions. Also, by using constellation, the ability to send private transactions between two or more participants in a network is added, making it more suitable for enterprise users.

A.3 Ethereum Private Blockchain Setup

Take Ethereum, the most popular smart contract platform, as an example to show the process of smart contract execution (Fig. A.2).

A.3.1 Ethereum Wallet

Ethereum wallet has functions such as account creation, exchange of Ethereum COINS, deployment of smart contracts, transfer of Ethereum coins (ETH), backup of wallet, etc. These functions are operated through the interface or menu after launching the client program. The smart contract part requires the implementation to write the corresponding code and then release it through the client.

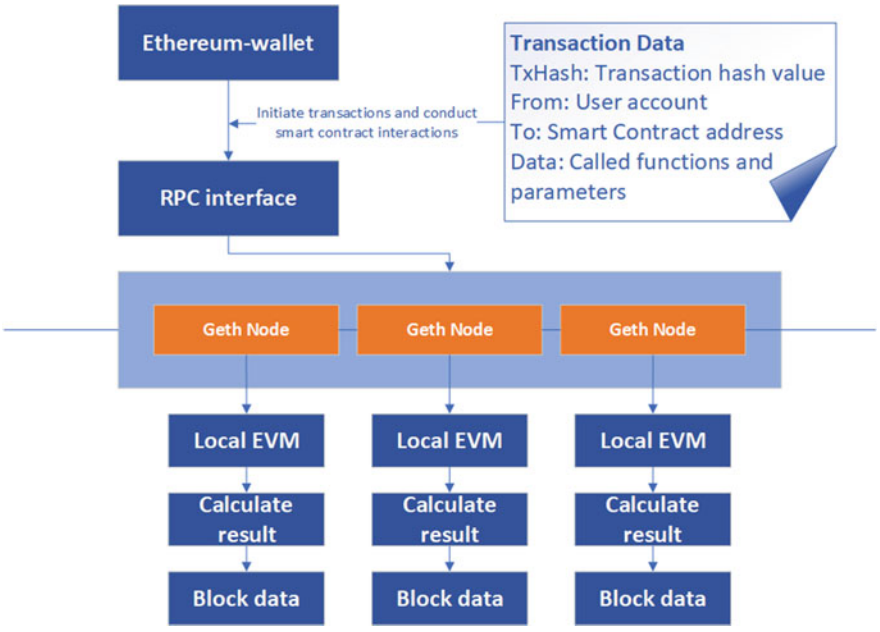


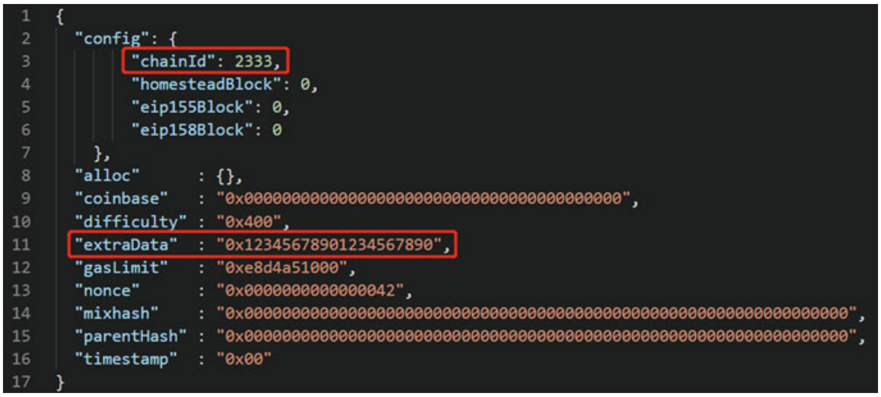
Fig. A.2 Ethereum smart contract execute process

A.3.2 Go Ethereum

Geth (Go Ethereum) is a client software written in Go language and implemented Ethereum protocol. Geth can manage the related API, call the API, dig and stop mining through command operation in the background.

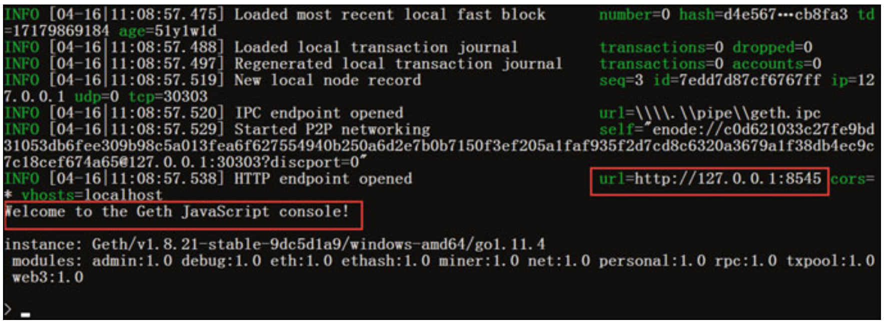
A.3.3 Setup Private Blockchain

- Config the foundation block: The foundation block of Ethereum is a manually configured genesis.json file that needs to be manually configured with chainId and extraDate (Fig. A.3).
- Create a storage directory: Place the genesis block file genesis. json in the parent directory. For example, if the installation directory is E:\Ethereum\geth, the JSON file will be placed in the 'Ethereum' directory. Create a new MyChain folder to store private chain block data.
- Generate the creation block based on gensis.json: Enter the following command in the terminal to generate the foundation block: $geth\ init\ genesis.json$ --$datad$ $ir"MyChain"$. When "Successfully wrote genesis state" appears, the creation block data is successfully generated.
- Start the Ethereum private chain node: Enter the following command to start the Ethereum client: $geth$ --$datadir$ $"MyChain"$ --$networkid$ 2333 --rpc --$rpccors$ $domain$ $"*"$ --$rpcapi\ eth, web3, personal$ --$nodiscover$ --$allow$-$insecure$-un $lock\ console$. $MyChain$ is the blockchain data storage directory, --$networkid$ is the id set in the previous creation block, --rpc and other parameters are used to realize the interaction between external programs and Ethereum, the API includes eth,web3, personal, --$nodiscover$ stands for single-node mode, --$allow$-$insecure$-$unlock$ stands for allowing account to unlock (insecure), and $console$ stands for launching Ethereum console (Fig. A.4).
- Generate Ethereum account: Enter $personal.newaccount()$ in the client to generate the Ethereum account, and enter the password to generate the Ethereum address.
- Check account balance: Enter $web3.eth.getBalance("your\ account\ address")$ or $web3.fromWei(eth.getBalance(eth.accounts[1]), 'ether')$ in the client to view the account balance.
- Unlock the account: Accounts need to be unlocked before mining and trading. In the client, type: $personal.unlockaccount("your\ account\ address",$ $"password")$ or $personal.unlockaccount(eth.accounts[0], "password")$ to unlock the account.
- Begin to dig: Using the following command, $miner.start()$, the initial mining will generate DAG data for Ethereum consensus, after which the mining will generate new blocks and rewards.



```json
{
    "config": {
        "chainId": 2333,
        "homesteadBlock": 0,
        "eip155Block": 0,
        "eip158Block": 0
    },
    "alloc"      : {},
    "coinbase"   : "0x0000000000000000000000000000000000000000",
    "difficulty" : "0x400",
    "extraData"  : "0x12345678901234567890",
    "gasLimit"   : "0xe8d4a51000",
    "nonce"      : "0x0000000000000042",
    "mixhash"    : "0x0000000000000000000000000000000000000000000000000000000000000000",
    "parentHash" : "0x0000000000000000000000000000000000000000000000000000000000000000",
    "timestamp"  : "0x00"
}
```

Fig. A.3 Genesis.json



```
INFO [04-16|11:08:57.475] Loaded most recent local fast block      number=0 hash=d4e567…cb8fa3 td
=17179869184 age=51y1w1d
INFO [04-16|11:08:57.488] Loaded local transaction journal         transactions=0 dropped=0
INFO [04-16|11:08:57.497] Regenerated local transaction journal    transactions=0 accounts=0
INFO [04-16|11:08:57.519] New local node record                    seq=3 id=7edd7d87cf6767ff ip=12
7.0.0.1 udp=0 tcp=30303
INFO [04-16|11:08:57.520] IPC endpoint opened                      url=\\\\.\\pipe\\geth.ipc
INFO [04-16|11:08:57.529] Started P2P networking                   self="enode://c0d621033c27fe9bd
31053db6fee309b98c5a013fea6f627554940b250a6d2e7b0b7150f3ef205a1faf935f2d7cd8c6320a3679a1f38db4ec9c
7c18cef674a65@127.0.0.1:30303?discport=0"
INFO [04-16|11:08:57.538] HTTP endpoint opened                     url=http://127.0.0.1:8545 cors=
* vhosts=localhost
Welcome to the Geth JavaScript console!

instance: Geth/v1.8.21-stable-9dc5d1a9/windows-amd64/go1.11.4
 modules: admin:1.0 debug:1.0 eth:1.0 ethash:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txpool:1.0
 web3:1.0

> _
```

Fig. A.4 Startup Ethereum client

- Stop to dig: When you stop digging, type *mine.stop*() to stop. Checking the balance again, you can see that the mining bonus has been transferred to your Ethereum account.

A.4 Ethereum Smart Contract Development

Solidity's syntax is close to Javascript and it is an object-oriented programming language. As a truly decentralized contract that runs on a blockchain network, it's a bit different:

- The Ethereum base is based on an account rather than UTXO, so there is a special address data type in solidity that locates the user, contract, and contract code.

- Since the embedded framework of the solidity language supports payments, there are keywords, such as “payable”, that enable payments to be implemented at the language level.
- Since you are using a blockchain network for data storage, you need to determine whether each variable is stored in memory or on a blockchain.
- Solfat’s anomaly mechanism is also quite different. In order to ensure the atomicity of contract execution, all execution would be canceled if an anomaly occurs.

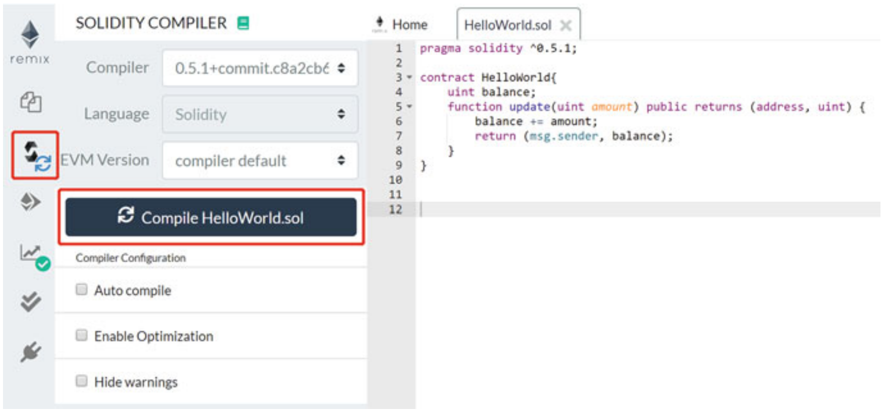
A.4.1 Development Tool

You can use geth to write and run smart contracts, or you can use remix-ide, a browser compiler that you don’t need to install. Remix-IDE url: http://remix.ethereum.org. You can also choose to install the remix-ide on your local computer. Download via url: https://github.com/ethereum/remix-ide or install via *npm install remix-ide -g*. Once the installation is complete, executing the remix-ide launches an 8080 port, opens the browser and enters *remix-ide* to open the remix compiler.

Remix provides a development environment for solidity smart contracts, which can be easily compiled, deployed, executed, and even faulted, suitable for beginners with little programming experience. However, version control, testing, and other development tools are not available. In fact, it is more convenient and professional for developers to compile and deploy DApp using Truffle3, Ganache, and web3.js.

A.4.2 Hello World

- As Fig. A.5. Create a new file in the menu on the left side of the ide and name it “helloworld.sol”. Enter the contract and compile. Contract contents: the function update receives a parameter of type unit, representing the amount transferred by others; Returns two parameters, one of type address, representing the address of the sender, and one of the final amount. The function of this function is to add up the amount and return it.
- After the contract is compiled, click “deploy” to deploy the contract.
- Run smart contracts: As shown in Fig. A.6, first check the current balance is 1024, then call the update function, and input the parameter is 2048. Click run to check the result in the last returned log, and the balance successfully becomes 3072.



Compiler: 0.5.1+commit.c8a2cb6
Language: Solidity
EVM Version: compiler default

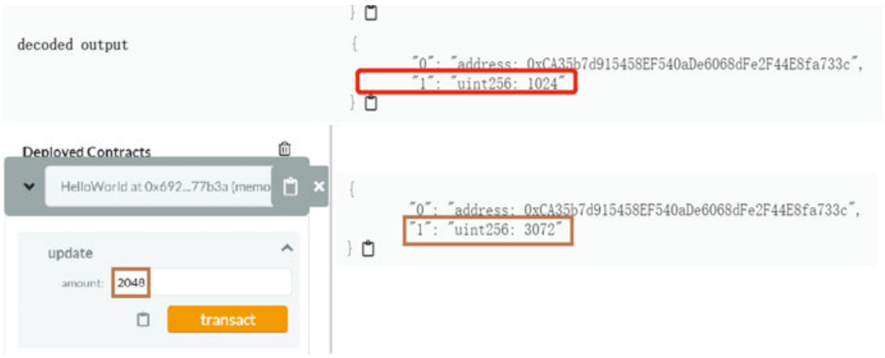
- [ ] Auto compile
- [ ] Enable Optimization
- [ ] Hide warnings

```
pragma solidity ^0.5.1;

contract HelloWorld{
    uint balance;
    function update(uint amount) public returns (address, uint) {
        balance += amount;
        return (msg.sender, balance);
    }
}


```

Fig. A.5 Compile HelloWorld



```
decoded output    {
                      "0": "address: 0xCA35b7d915458EF540aDe6068dFe2F44E8fa733c",
                      "1": "uint256: 1024"
                  }
```

amount: 2048

```
{
    "0": "address: 0xCA35b7d915458EF540aDe6068dFe2F44E8fa733c",
    "1": "uint256: 3072"
}
```

Fig. A.6 Run smart contract