Edgence: A Blockchain-enabled Edge-computing Platform for Intelligent IoT-based dApps

Jinliang Xu[1], Shangguang Wang[1,*], Ao Zhou[1], Fangchun Yang[1]

[1]State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China
* The corresponding author is Shangguang Wang, email: sgwang@bupt.edu.cn.

***Abstract*—Nowadays scalable IoT management is a bottleneck of IoT development due to the geographically dispersed distribution, fragmented ownerships, and ever-growing population of IoT devices. To intelligently manage massive decentralized applications (dApps) in IoT usecases, Edgence (EDGe + intelligENCE) is proposed to use edge clouds to access IoT devices and users, and then use its in-built blockchain to realize self-governing and self-supervision of the edge clouds. Edgence proposes to use masternode technology to introduce IoT devices and users into a closed blockchain system, which can extend the range of blockchain to IoT-based dApps. Further, masternodes do good to scalability by raising the TPS (transactions per second) of the blockchain network. To support various dApps, a three-tier validation is proposed, namely script validation, smartcontract validation, and masternode validation. To avoid energy consumption resulted by blockchain consensus, Edgence proposes a random but verifiable way to elect a masternode to generate each new block. The potential of the tailored Edgence is shown by examples of decentralized crowdsourcing and AI training.**

Keywords*—*mobile edge computing, IoT, crowd-intelligence, blockchain, dApp

I. Introduction

Ever since the first electronic computer was born, decentralization and centralization of computer networks have been interacting and promoting each other, and jointly accelerating the evolution of information technology [1]. Supercomputer era means centralization at a high level, which helped to facilitate enterprise server producers like IBM and GE. Then decentralization boomed as personal computer era came, which produced companies like MicroSoft, Apple, and the concepts of Internet and grid computing. After that, cloud computing era brings new centralization trend of computation and network resources, where Google, Amazon and VMware play the best. While today, as concepts like Internet of Things (IoT), edge computing, peer-to-peer (P2P) communication and blockchain appear and develop fast, no one can deny that we are on the very verge of another wave of decentralization.

Decentralization is naturally integral to the development of IoT: 1) scalable IoT management is a bottleneck of IoT development due to the geographically dispersed IoT devices, their fragmented ownerships, and ever-growing population [2]; 2) as multimedia data is transferred from one edge devices to another in IoT usecases, if IoT management is centralized and its server serves as an intermediary, the data would go through the core network and result in long latency because IoT devices and users are geographically distributed. So centralization management is not fit for time-sensitive IoT applications [3], [4]; 3) centralization can also cause high operation cost, and the users in return will be charged high fees to cover the cost [5]; 4) what is more, if a centralized platform has the full copy of the IoT data, it will raises possibility of leakage of private data or company-sensitive data [6], [7]; 5) the ownership of IoT network is fragmented in that no one can deploy all infrastructures of the whole IoT network, and many personals and organizations will build their own IoT subnetworks for private or public use [2]. As they will not trust each other, a mandatory centralized management does not work here.

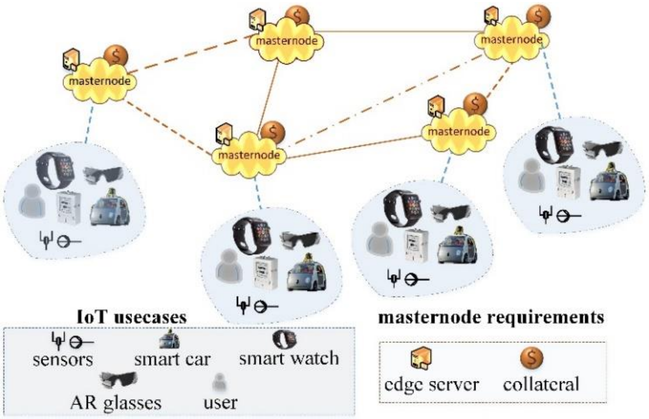

Fig. 1 Decentralized Edgence platform running on edge clouds of mobile edge computing. Edgence units edge clouds provided by personals, and then IoT devices and users can obtain services from the whole network from nearby edge cloud. Edgence platform are made of many masternodes. Each masternode is deployed on one edge cloud, and it hosts a blockchain node and a fixed amount of fund as collateral.

However, every coin has two sides, and dencentralization of IoT is not easy to realize. The key is the trust among all participants. In the centralized management, the management center (e.g., a platform or company, and the person or organization behind it) serves as the third party guarantee for all participants. While decentralized IoT platform abandons the management center, the challenge is how to build mutual trust among participants in a decentralized way [8]. Blockchain technology makes it possible by implementing self-governing and self-supervision of a closed payment system like Bitcoin. And blockchain smartcontracts can be utilized to implement some IoT applications. But it is still not enough for real IoT usecases because existing blockchain technology cannot connect virtual reality of blockchain to the real world.

In this work, Edgence (EDGe + intelligENCE, Fig. 1) is proposed to serve as a blockchain-enabled edge-computing platform to intelligently manage massive decentralized applications (dApps) in IoT usecases[1]. To extend the range of blockchain to IoT-based dApps, Edgence adopts masternode technology to connect to a closed blockchain-based system to the real world. A masternode contains a full node of the blockchain and a collateral, and is deployed on an edge cloud of mobile edge computing, which is convenient for the masternode to use resources of the edge cloud to run IoT dApps [9]. An edge cloud is always near to mobile users and IoT devices**.** So Edgence can respond to users without long latency [9], [10]. What is more, as the count number of masternodes is smaller than that of full nodes of the blockchain for its higher minimum requirements (collateral and edge cloud), masternodes do good to scalability of the platform by raising the TPS (transactions per second) of the blockchain network substantially [11], [12]. To better meet the various demands of IoT based dApps, a three-tier validation scheme is proposed, namely script validation (e.g., Bitcoin), smartcontract validation (e.g., Ethernum), and masternode validation (more complex dApps that must interact with the real world). To solve the PoW (proof of working) energy consumption problem, Edgence use an innovative method to elect the validator when a new block is generated each time, which can avoid massive calculation of traditional PoW and improve TPS. That is to say, Edgence is specially tailored for IoT dApps.

The rest is organized as follows. Section 2 introduces what a masternode is, and how the network of masternodes manages Edgence in a decentralized way. Section 3 describes how Edgence supervises what happens to it with a three-tier validation. Then Section 4 tell us how a masternode is elected as a validator. Finally, two IoT dApps supported by Edgence are listed to show the potential of Edgence in IoT industry.

II. Masternodes as Decentralized Management

Edgence is under the management of a committee that comprises masternodes. No one can gain absolute control over the platform. The entry threshold of a masternode is much more higher than a general full node of existing blockchains, which can decrease the count number of masternodes and improve their trust against general full blockchain nodes. As a result, consensus across masternodes is easy to reach, and Edgence can have much larger TPS, which is very suitable for IoT usecases.

As shown in Fig. 2, masternodes are like a small cloud that underpins a blockchain network in order to provide extra IoT services and offer new features that cannot be provided by the traditional consensus algorithms and mining methods. Anyone can run a masternode. But in order to do so, she needs to have a fixed amount of fund available as collateral. If the collateral behind a masternode is spent, or if a masternode stops to provide services to the network for more than a certain period of time, it is removed from the masternode list until normal service resumes. In addition, a constant reward will be uniformly distributed to all masternodes through a rewarding system of the blockchain network. If the blockchain network breaks down, neither of masternodes' collateral and reward become worthless, which is not good to every masternode. So the best strategy of a rational masternode is to protect the network and provide IoT services as expected. Compared with a bitcoin-like blockchain that runs entirely by unpaid volunteers, a masternode-driven Edgence can scale more efficiently and deploy services more quickly [5].

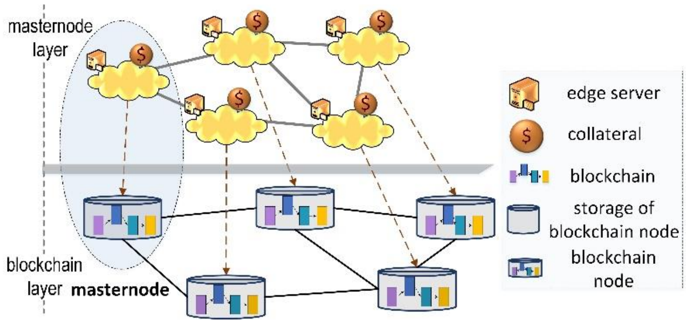

Fig. 2 Masternode layer as decentralized management in Edgence. The set of masternodes can be considered as another layer that is constructed on the blockchain network layer. It connects IoT usecases to the blockchain network. Everyone can join or leave the masternode layer at will, if she has an edge cloud and a fixed amount of fund as collateral.

Physically, a masternode contains a full node of the blockchain, the collateral given by the owner of the masternode, and a server as edge cloud of mobile edge computing [9]. While logistically speaking, masternodes run as another layer for trust on top of the original blockchain layer. With this kind of trust, a masternode can do more than a general blockchain full node can.

Masternodes running on edge clouds nicely support potential IoT based dApps: 1) a masternode should have enough storage resource to host the block data of the blockchain. In addition, it should have enough computation resource to run IoT based dApps and interact with massive IoT devices and users. What is more, as masternodes need to communicate with each other, network bandwidth is also needed. The demand of storage/computation/network resources decides that a masternode should be at least a small edge cloud; 2) for a masternode, to save cost and enhance its competitiveness against other masternodes, it should be near to as many mobile users and IoT devices[10], [13]. That is to say, masternodes should be deployed at the edge of network or near to people.

Edgence can resist Sybil attack with the help of the collateral. Sybil attack is a general problem in public Blockchain, which is why Bitcoin adopts PoW, and ETH adopts PoS (proof of stake) to reach a consensus across the blockchain nodes. Edgence adopts collateral of masternodes to increase the entry threshold of malicious nodes, which makes a successful sybil attack too expensive to perform. Specifically, to conduct a successful Sybil attack on Edgence, the attacker must control many masternodes. While each masternode of Edgence has to host a fixed amount of crypto-currency as collateral. So the attacker must buy enough crypto-currency from the secondary market. Considering the limited supply of crypto-currency and the low liquidity available on the market, it becomes an impossibility to attain a large enough to supply to succeed at such an attack.

From the above, we can conclude that Edgence is a decentralized platform rather than a centralized one as it is managed by masternodes: 1) security: it is not easy for

[1] Edgence and several dApps on it are being under sustainable development by our team right now, and it has been opensourced at Github, and the link is https://github.com/EdgeIntelligenceChain. The homepage of Edgence project is at http://www.edgence.org/ .

someone to control most masternodes; 2) openness: everyone can join in the masternode committee to manage Edgence, and every masternode can withdraw from Edgence and sell the collateral.

III. Three-tier Validation

Validation is vital important in IoT-based dApps, including authentication of massive IoT devices, transaction verification, or whether a dApps is executed as expected. Edgence adopts a three-tier validation scheme, namely script validation, smartcontract validation, and masternode validation (Fig. 3), for two reasons: 1) to better meet the various demands of IoT based dApps; 2) to improve validation efficiency. The powers of script validation, smartcontract validation and masternode validation become bigger one by one, but their costs become larger one by one, too. When a node is encountered a validation task, to reduce the validation cost, first it should judge which type of validation it belongs to, and then choose the most suitable validation type to perform this validation task.

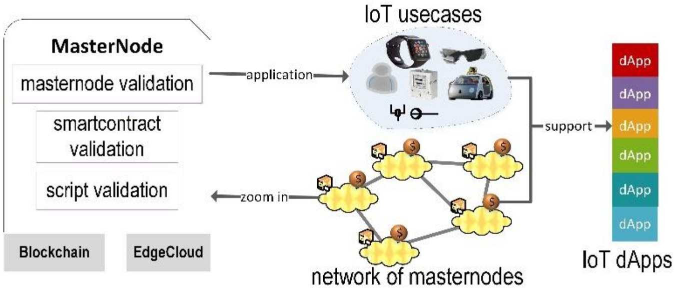

Fig. 3 Three-tier validation of Edgence. Every masternode of Edgence contains a blockchain node and hardware resources, and can perform three-tier validation. Masternode network connect real world and blockchain, and generate various IoT dApps.

A. Script Validation

Script validation is to check whether the next person that wants to spend the currency to be transferred can gain the access of this money. A script is essentially a list of operations recorded with each transaction, and they describe how to perform script validation of this transaction.

The first blockchain to use script validation is Bitcoin, but Edgence disables most operators of Bitcoin's script system, and keeps only a minimally viable set of operators to perform the basic three kinds of transactions, namely Pay-to-Public-Key-Hash transaction[2], and multi-signature transaction[3]. This simplification can decrease the validation complexity and improve the security, which can help to deal with lots of transactions in IoT cases at quite low fees. Note that Edgence allows maximum 100 signatures in a multi-signature transaction (much larger than 20 signatures in Bitcoin), which can help to realize the decentralized management of the fund pool, etc.

B. Smartcontract Validation

A smartcontract is a piece of codes that can help you exchange money, property, shares, or anything of values in a transparent, conflict-free way while avoiding the services of a middleman, which is much more flexible than script. Ethernum is the first blockchain that can run smartcontract massively. Specifically, an asset or currency is transferred into a program. At some point it automatically validates a condition. And it automatically determines whether the asset should go to one person or back to the other person. It also can determine whether the asset should be immediately refunded to the person who sent it. In the meantime, the decentralized ledger also stores and replicates the document. And the ledger gives it a certain security and immutability. In this process, Ethernum records all executed codes as transactions onto a blockchain.

Smartcontracts work as follows: when a new smartcontract is completed by the developer, at first it should be broadcast to the whole blockchain network, and every node will receive one of its copy. The process of smartcontract validation is like script validation: at one time, when the validator is validating the block, he will validate all of the transactions in this block one by one; when he validate a transaction, he will execute the smartcontract of this transaction, and stores the results onto the blockchain. After that, every other nodes should execute the smartcontract again, and check whether the results given by the validator are right. If the validator is found wrong, his chance of validating this block will be taken away.

The runtime of smartcontracts is EDVM (Edgence virtual machine), which is quite different from script validation. The reason is that smartcontracts allow developers to create their own operations with different complexities, which proposes high demand of uniform runtime on all nodes. EDVM makes sure that smartcontracts are separated from each other, and EDVM itself is separated from the host machine. This further improves portability of the smartcontracts. EDVM is compatible with EVM (Ethernum virtual machine), that is to say, most existing smart contracts running on Ethernum could work within Edgence platform, which makes it easy for users to write IoT-based dApps on Edgence.

C. Masternode Validation

With script validation and smartcontract validation, Edgence can work like or compatible with most existing blockchains, including UTXO-based chains, which is represented by Bitcoin, and account-based chains, which is represented by Ethernum. At this point, however, it is still a closed payment system, and cannot sense and react to the real world[14], [15], let alone support for real IoT-based dApps. For example, if a validation process in an IoT usecase takes the outside temperature as a parameter, the validator will first read the temperature value from a sensor and then complete the validation. Note that in script validation and smartcontract validation, all other nodes will perform the same process to check the results given by the validator. As it is hard to ensure that all nodes read temperature values from the identical sensor at the same time, the validation will be a failure. That is to say, a link between the physical universe of IoT usecases and the virtual reality of blockchain invalidates the script and smartcontract validation.

Edgence uses masternode validation to solve the above-mentioned validation failure problem by connecting blockchain to real world. Masternode validation can read data from three sources: 1) blockchain data of Edgence (e.g., a

[2] https://en.bitcoinwiki.org/wiki/Pay-to-Pubkey_Hash
[3] https://en.bitcoin.it/wiki/Multisignature

transaction log, hash value of a block, etc.); 2) machines including other nodes and IoT devices (e.g., various kinds of sensors for different purposes) [15]; 3) the owner of the masternode (e.g., public opinion survey, or election of a validator that are participated by owners of masternodes). When a new dApp that needs masternode validation is broadcast to the whole masternode network, it will be checked by every masternode. If most masternodes approve this dApp, it will be deployed on every masternode, and then users of Edgence can use this dApp by paying money to the masternode that runs this dApp. Note that a new dApp can be created by composing several already-deployed dApps or combining the three kinds of validations.

Masternode validation is to Edgence what IO is to a computer. If the node network of Edgence can be considered as a decentralized computer, masternode validation helps it to exchange data with the physical universe of various IoT usecases, which is impossible to realize by existing blockchains like Bitcoin or Ethernum. The trust of masternode validation comes from masternodes, instead of multi-party authentication in script validation and smartcontract validation.

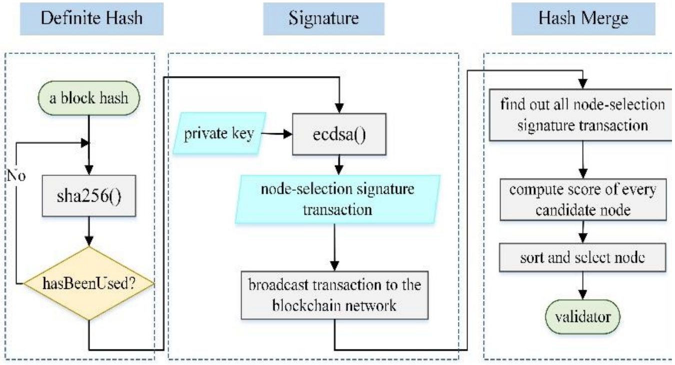
Definite Hash
Signature
Hash Merge
a block hash
No
sha256()
hasBeenUsed?
private key
ecdsa()
node-selection signature transaction
broadcast transaction to the blockchain network
find out all node-selection signature transaction
compute score of every candidate node
sort and select node
validator

Fig. 4 Three steps of validator election in Edgence.

IV. Validator Election in Edgence

Validator election of a blockchain concerns its operational efficiency and use-cost. More specifically, a better validator election method of Edgence can reduce energy consumption in reaching consensus across all blockchain nodes, improve TPS, and lower the usage fee charged from users. Validator election must meet two essential properties, namely randomness (no one can predict the results in advance), and verification (no one can check the selected results) [11]. Bitcoin uses PoW for validator election, but has been criticized for wasting too much energy, which is not suitable for IoT devices. Ethernum is planned to switch from PoW to PoS, which is not good for TPS improvement and may result in negative result of Matthew effect. In Edgence, a simple verifiable method for random node selection in trustless self-organized network is designed, where masternodes' private keys together with well-designed workflow contribute to the verifiability and randomness.

As is shown in Fig. 4, the workflow of the proposed validator selection method can be considered **as** three steps, namely *Definite Hash*, *Signature*, and *Hash Merge*:

- Definite Hash: This step is to produce a definitive hash, where the word *definite* means that all nodes can get the same hash value without communicating with each other. In order to achieve this goal, the hash of a new generated block is chosen as the seed hash. However, block hash can be manipulated by its miner by adjusting the order of packaged transactions inside it, which may be used by some malicious nodes/miners for an attack. To avoid this, every node can check whether the block hash has been used before. If the answer is no, the seed hash is the definite hash we want. Otherwise, the hash of the current hash is computed to generate a new one, until the first unused hash is obtained as the *definite hash*. Note that a record of the used definite hashes should be built for public to check whether a hash has been used before.
- Signature: If a node wants to participate the selection, it should sign the *definite hash* using its own private key. Then the node encapsulates both of the definite hash and its signature into a so-called *node-selection-signature transaction*, and send it to the blockchain. If the *definite hash* is not generated from the specified block, the transaction will be discard in later step of the selection. Once the transaction is confirmed by the network, all nodes can see it, and the sender cannot deny it.

  As the private key of a node is only hosted by the node itself, and not known by others, it cannot predict other nodes' signatures in advance. As a result, every node is equal in this step no matter how much computing power it occupies.
- Hash Merge: At one time point, each node begins to separate *node-selection-signature transaction*s from blocks generated during the whole process of node selection. If any transaction is not legal (e.g., either the definite hash or the sender's signature is not included, or the definite hash is not right), or more than one *node-selection-signature transaction*s are sent by one node, the sender of the transactions will be considered as malicious node, and all transactions from it will be discarded. These malicious nodes, together with the nodes that have not sent a *node-selection-signature transaction*, are considered as to give up the node selection this time, and will not be in the set of candidate nodes in the later step.

  A node or set of nodes is selected from the candidate nodes, which can be done by every node. For each candidate node, compute the hash values of all other candidate nodes' signatures, and then combine these hash values using XOR operator to generate a new hash value. The new generated hash value is used as the score of this candidate node. If it is to select a certain number of nodes, candidate nodes with largest scores are selected. Nodes get the same final result as long as they follow the rules.

Now we show the election is verifiable and random:

- Verification*: Verification* means that every participating node must act according to the predetermined rules, or it will be found out at once by others, and it will not lead to bad effect on the later step. Illegal action in *definite hash* step will result in a different *hash1* from other honest nodes. As *hash1* is encapsulated in the *node-selection-signature transaction*, this illegal action in *definite hash* step will be revealed to the public. Illegal action in *signature* step may generate two consequences. One is illegal *node-selection-signature transaction*, e.g., lack of needed data segments, which is easy to be found. The other one is wrong signature, such as signing on the wrong hash instead of the *hash1* in last step, or signing with the wrong private key. Every node can use the node's public key to check whether its signature is right. Illegal action in *hash merge* cannot generate the right selected node set, which will not be recognized by honest nodes.

- Randomness*: Randomness* means that no one can control or predict the selected nodes in advance. The signature of *hash*1 by each node in *signature* step is must be generated using the node's private key. As the private key is hosted by each node and not known by others, no one can know the signature before the signature is broadcasted by the node [6]. What is more, the most widely used signature algorithms, namely ECDSA and Schnorr, can generate different signatures for the same information using the same private key, which add more randomness of the *signature* step. A node's score is generated with all other nodes' signatures. That is to say, a malicious node cannot control its own score or any other node's score. If there are at least one node not under the malicious node's control, the scores become completely different.

The designed validator election of Edgence makes it suitable for IoT usecases. Edgence avoids energy waste in reaching consensus completely. As the number of masternodes is smaller, the time needed for reaching consensus over the whole blockchain network is much less, which helps to improve TPS to support various IoT dApps.

V. IoT-based DApps deployed on Edgence

A. *Decentralized Crowdsourcing*

Crowdsourcing has great potential in promoting AI development by providing a continuous supply of labelled data for model training and evolving. A typical usecase is crowdsourcing based AI (Fig. 5), including AI training, where crowdsourcing service helps to label the raw data to provide enough labeled dataset for AI training [16]; AI augmenting, where if the trained model generates labels of low confidence for some new data, crowdsourcing service would be adopted to replace these labels with labels from humans; and AI evolving, where the new human-labeled data in AI augmenting is used to train the AI model again, in the hope of larger confidence value when encountered with similar data next time.

However, the traditional crowdsourcing platforms are totally centralized, which results in the following problems:

- privacy risks of user data. A centralized platform has the full copy of the tasks' data, which raises possibility of leakage of private or company-sensitive data [7];

- long latency. Multimedia data attached to tasks from a requester must go through core network to reach workers, which may result in long latency. So crowdsourcing is not fit for time-sensitive applications like AR or IOV [3], [4].

To solve these problems, Edgence first decentralizes a crowdsourcing platform by deploying it on many masternodes, and then supervises all activities of every participant and the public fund of the decentralized crowdsourcing platform (Fig. 5). What we need to do is to design a set of incentive functions for three participants, namely requester, worker and masternodes, hence in theory making the platform incentive-compatible, where every participant must cooperate honestly to maximize her own interest [17], [18]. Compared with traditional crowdsourcing platform, a decentralized crowdsourcing platform under management of Edgence can provide low fee cost, time latency, and stable accessibility.

B. *Decentralized AI training*

Due to limited storage and computing power, edge devices at edge network (e.g., mobile phones and surveillance cameras) cannot train locally deep learning models[13],[14]. The traditional deep learning training requires users (i.e., owners of edge devices) to upload local dataset to a cloud center first, and then do the training process there using massive resources of cloud center. However, uploading local dataset to a centralized cloud center controlled by someone else leaves user data privacy in danger. In addition, uploading multimedia data (e.g., photos and videos generated by users) will occupy vast part of bandwidth resource of users and storage resource of cloud center, which will result in weak scalability as the number of terminal devices or data volume of each user increase [12], [19].

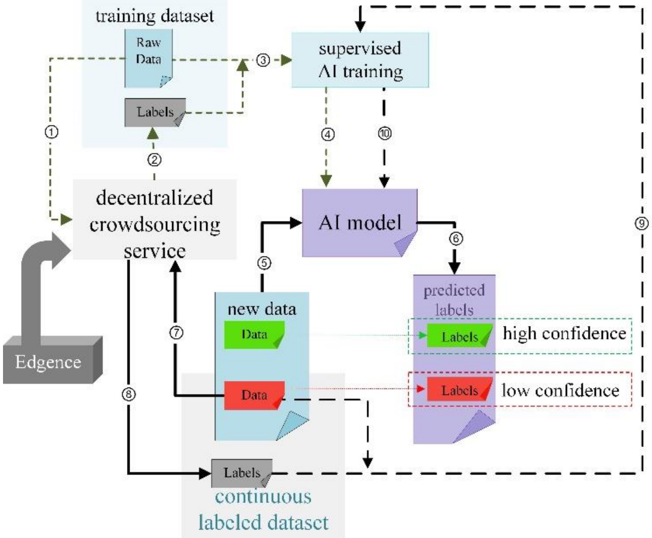

Fig. 5 Decentralized crowdsourcing that is engined by Edgence, and it support decentralized AI training (①②③④), AI augmenting (⑤⑥⑦⑧), and AI evolving (⑨⑩). An existing AI system can make use of a WebAPI of crowdsourcing service provided by a crowdsourcing platform for better performance. The decentralized crowdsourcing platform runs on many of edge clouds of mobile edge computing network or masternodes of Edgence. And in this case Edgence can provide resources and management functions.

To solve this problem, we cut a general deep learning training network into front and back subnetworks, and use Edgence to manage them (Fig. 6). The front subnetwork contains several layers close to input data, and is deployed at every edge device, and trained separately on each edge device using the local dataset. The back subnetwork is deployed at cloud center. In forward propagation of each training epoch, the outputs of all front subnetworks are sent to the back subnetwork for latter training; and in backward propagation, the output of back subnetwork is sent to each front subnetwork. As no original dataset is transferred from edge devices to

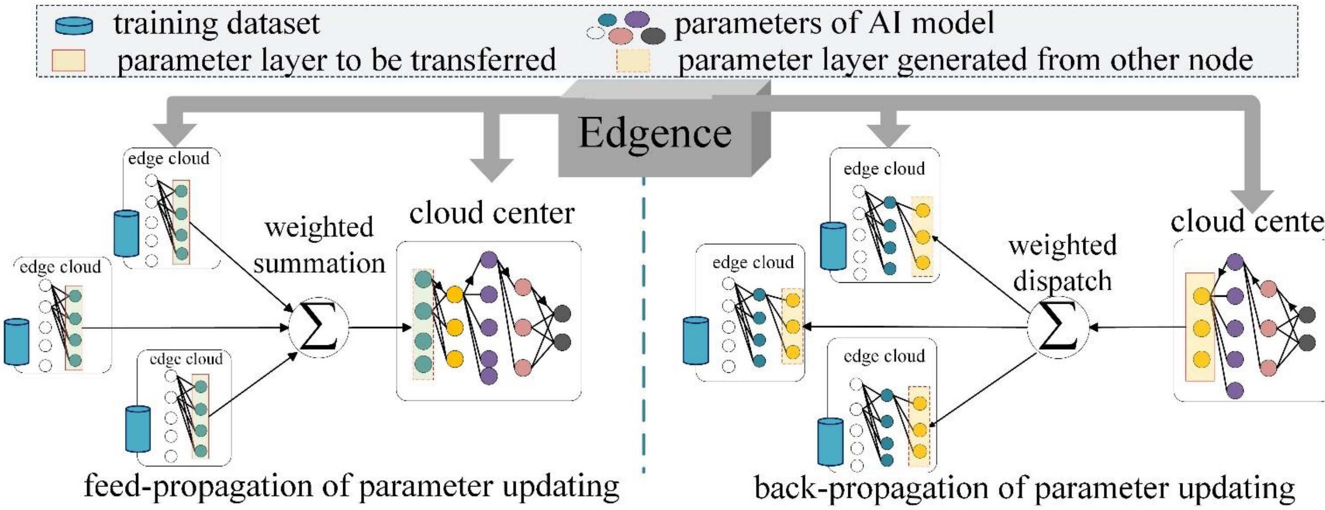

Fig. 6 Decentralized AI training under the management of Edgence. Feed-propagation and back-propagation are two ways of updating model parameters during AI training. One parameter layer is identified by one color. The first several layers are trained at many edge clouds, which is to utilize the datasets of mobile users. The last several layers are trained at a remote cloud center, which keeps communicating with edge clouds during the whole training. Edgence's work is to supervise all activities of edge clouds and remote cloud center.

cloud center, the framework can improve data privacy protection and high scalability. The front and back networks are all implemented in agents, which is better for better cooperation among them, reducing the administrative burden, and offer a more flexible way to protect data transferring from tampering [20], [21].

Conclusion

This work introduces how Edgence uses edge clouds and blockchains to intelligently manage massive decentralized IoT applications (dApps). To support IoT dApps, Edgence is tailored from several ways, including masternodes as its decentralized management, three-tier validation, and validator election method. At last, decentralized crowdsourcing and AI training are listed to show the scalability and cost control of Edgence for IoT dApps.

In the future, we will proceed with the development of the Edgence platform. Moreover, we will complete and test some example dApps on it, including decentralized crowdsourcing and AI training.

Acknowledgement

This work was supported in part by the National Science Foundation of China (Grant No. 61922017). Shangguang Wang is the corresponding author.

References

[1] H. El Bakoury, M. A. R. Chaudhry, W. Cerroni, H. He, and A. Barbir, “Standards for major internet disruptors: Blockchain, intents, and related paradigms,” IEEE Communications Standards Magazine, vol. 2, no. 3, pp. 14–15, 2018.

[2] Y. Zhang, Y. Wu, H. Moustafa, D. H. Tsang, A. Leon-Garcia, and U. Javaid, “Multi-access mobile edge computing for heterogeneous IoT,” IEEE Communications Magazine, vol. 56, no. 8, pp. 12–13, 2018.

[3] F. Gao, L. Zhu, M. Shen, K. Sharif, Z. Wan, and K. Ren, “A blockchainbased privacy-preserving payment mechanism for vehicle-to-grid networks,” IEEE Network, pp. 1–9, 2018.

[4] J. Kang, R. Yu, X. Huang, S. Maharjan, Y. Zhang, and E. Hossain, “Enabling localized peer-to-peer electricity trading among plug-in hybrid electric vehicles using consortium blockchains,” IEEE Transactions on Industrial Informatics, vol. 13, no. 6, pp. 3154–3164, 2017.

[5] W. Zhao, J. Liu, H. Guo, and T. Hara, “ETC-IoT: Edge-node-assisted transmitting for the cloud-centric internet of things,” IEEE Network, vol. 32, no. 3, pp. 101–107, 2018.

[6] H. Liu, Y. Zhang, and T. Yang, “Blockchain-enabled security in electric vehicles cloud and edge computing,” IEEE Network, vol. 32, no. 3, pp. 78–83, 2018.

[7] T. Wang, G. Zhang, A. Liu, M. Z. A. Bhuiyan, and Q. Jin, “A secure IoT service architecture with an efficient balance dynamics based on cloud and edge computing,” IEEE Internet of Things Journal, pp. 1–13, 2018.

[8] Z. Li, Z. Yang, and S. Xie, “Computing resource trading for edge-cloudassistedinternet of things,” IEEE Transactions on Industrial Informatics, vol. 15, no. 6, pp. 3661–3669, 2019.

[9] K. Wang, H. Yin, W. Quan, and G. Min, “Enabling collaborative edge computing for software defined vehicular networks,” IEEE Network, pp. 1–6, 2018.

[10] Z. Xiong, Y. Zhang, D. Niyato, P. Wang, and Z. Han, “When mobile blockchain meets edge computing,” IEEE Communications Magazine, vol. 56, no. 8, pp. 33–39, 2018.

[11] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich, “Algorand: Scaling Byzantine agreements for cryptocurrencies,” in Proceedings of the 26th Symposium on Operating Systems Principles, pp. 51–68, ACM, 2017.

[12] O. Novo, “Scalable access management in IoT using blockchain: a performance evaluation,” IEEE Internet of Things Journal, pp. 1–8, 2018.

[13] N. Kumar, J. J. Rodrigues, M. Guizani, K.-K. R. Choo, R. Lu, C. Verikoukis, and Z. Zhong, “Achieving energy efficiency and sustainability in Edge/Fog deployment,” IEEE Communications Magazine, vol. 56, no. 5, pp. 20–21, 2018.

[14] Y. Liang, Z. Cai, J. Yu, Q. Han, and Y. Li, “Deep learning based inference of private information using embedded sensors in smart devices,” IEEE Network, vol. 32, no. 4, pp. 8–14, 2018.

[15] F. Xiao, Z. Guo, Y. Ni, X. Xie, S. Maharjan, and Y. Zhang, “Artificial intelligence empowered mobile sensing for human flow detection,” IEEE Network, vol. 33, no. 1, pp. 78–83, 2018.

[16] J. Xu, S. Wang, N. Zhang, F. Yang, and X. S. Shen, “Reward or pena1ty: A1igning incentives of stakeho1ders in crowdsourcing,”

IEEE Transactions on Mobile Computing (Early Access), DOI: 10.1109/TMC.2018.2847350, pp. 1–13, 2018.

[17] J. Xu, S. Wang, B. Bhargava, and F. Yang, "A blockchain-enabled trustless crowd-intelligence ecosystem on mobile edge computing," IEEE Transactions on Industrial Informatics, vol. 15, no. 6, pp. 3538–3547, 2019.

[18] Y. Wang, Y. Gao, Y. Li, and X. Tong, "A worker-selection incentive mechanism for optimizing platform-centric mobile crowdsourcing systems," Computer Networks, vol. 171, pp. 107–144, 2020.

[19] X. Qiu, L. Liu, W. Chen, Z. Hong, and Z. Zheng, "Online deep reinforcement learning for computation offloading in blockchain-empowered mobile edge computing," IEEE Transactions on Vehicular Technology, vol. 68, no. 8, pp. 8050–8062, 2019.

[20] H. Wei, W. Feng, C. Zhang, Y. Chen, Y. Fang, and N. Ge, "Creating efficient blockchains for the internet of things by coordinated satelliteterrestrial networks," IEEE Wireless Communications, 2020.

[21] W. Liang, Y. Fan, K.-C. Li, D. Zhang, and J.-L. Gaudiot, "Secure data storage and recovery in industrial blockchain network environments," IEEE Transactions on Industrial Informatics, 2020.

BIOGRAPHIES

JINLIANG XU received the bachelor degree in electronic information science and technology from Beijing University of Posts and Telecommunications in 2014. Currently, he is a Ph.D. candidate in computer science at the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications. His research interests include Mobile Cloud Computing, Blockchain, AI, and Crowdsourcing.

SHANGGUANG WANG received his PhD degree of computer science at Beijing University of Posts and Telecommunications in 2011. He is Professor and Vice-Director at the State Key Laboratory of Networking and Switching Technology (BUPT). He has published more than 150 papers recent years, and played a key role at many international conferences, such as general chair and PC chair. His research interests include service computing, cloud computing, and mobile edge computing. He is a senior member of the IEEE in 2011, and Editor-in-Chief of the International Journal of Web Science.

AO ZHOU is an associate professor at the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications. She received her Ph.D. degree in computer science at Beijing University of Posts and Telecommunications of China in 2015. Her research interests include cloud computing, service reliability.

FANGCHUN YANG received his Ph.D. degree in communication and electronic system from the Beijing University of Posts and Telecommunication in 1990. He is currently a professor at the Beijing University of Posts and Telecommunication, China. His research interests include network intelligence and communications software. He is senior member of the IEEE in 1994, and a fellow of the IET.