https://doi.org/10.48048/tis.2022.5072

Adaptation of Blockchain using Ethereum and IPFS for Fog based E-Healthcare Activity Recognition System

Lakshmi Narayana Kodavali* and Sathiyamurthy Kuppuswamy

Department of Computer Science and Engineering, Pondicherry Engineering College, Puducherry 605014, India

(*Corresponding author's e-mail: kodavali.lakshmi@pec.edu)

Received: 2 December 2020, Revised: 9 August 2021, Accepted: 16 August 2021

Abstract

The cloud storage is far away from us and it is not capable of handling huge bandwidth data due to network latency. The goal of the Fog computing is to decrease the data that needs to be transferred to the cloud for data processing and to increase the efficiency. Fog computing improves the QoS and also reduces network bandwidth. All machine learning algorithm performances are dependent on the quality of the training data. If the training data is inadequate or it is modified by attackers, then the machine learning algorithm will miss predict and may give invalid results. In order to avoid modification of training data, it is preferable to store it in Blockchain. Blockchain is a decentralized model and its data structure is practically difficult to forge, hence it has attracted both industry and research now-a-days. Proposed system uses Ethereum platform to implement Blockchain. Ethereum allows users to create and run decentralized applications (DApps) to make agreements and to conduct transactions directly with each other without any third party by making use of smart contracts. Blockchain is convenient to store only a small amount of data, hence alternative solution to store large amount of data, for example in healthcare, in Blockchain is possible with the help of IPFS (Interplanetary File System). In e-healthcare applications, Activity Recognition System (ARS) is the most significant undertaking in remote checking of patients experiencing physical medical issues for taking quick action. Hence this paper, especially concentrate on the overall framework of the e-healthcare ARS and implementation of Blockchain to store e-healthcare training data to avoid forging, ultimately which improves ARS results. Our implementation results also show that constant and less transaction fee required to store into Blockchain irrespective of size of training data with help of IPFS and also proved transaction throughput increases and network delay decreases with help of IPFS.

Keywords: Blockchain, Ethereum, IPFS, Fog computing, Activity recognition system

Introduction

A Blockchain works in a decentralized environment and it has a sequence of blocks which are connected to one another using cryptography technique [2]. Each block consists of transaction data, hash of the previous block and a timestamp as shown in **Figure 1**. Blockchain is unsusceptible to change data by its design. In a Blockchain, transactions among 2 parties are recorded in an efficient, verifiable and permanent way [5]. Such a Blockchain can present an innovative solution for long standing problems of security related to data storage in centralized systems. Blockchain can be considered as the new face of cloud computing, and is expected to reshape the organizational and individual behavioral models.

Important feature of Blockchain is a distributed database. It means no centralized database or server exists instead the same Blockchain which is replicated in all nodes of the network. Every node in the network receives a copy of Blockchain where each block has a list of transactions in an encrypted format using asymmetric keys. Due to the complexity of mathematical formulas used in cryptography techniques, it is practically difficult to guess the keys and to crack the transactions. The sender can use his public key to encrypt a message to be sent and the recipient can use his private key to decrypt the message. Every new transaction is broadcast and updated to all the network nodes to maintain the consistent database across the whole Blockchain network [7].

Smart Contracts are the programs for predefined rules which are deployed into the Blockchain and these programs execute automatically to make sure that every transaction has to satisfy the predefined conditions to complete the transaction. Smart Contracts work based on the simple "if…then…" statements. Smart Contracts are playing a more vital role in the business among a group of untrusted persons, where every transaction can be completed according to rules agreed by all business stakeholders without the involvement of third-party verification [10]. Smart contract used in this paper consists of 2 functions which are set( ) and get( ) as shown in **Figure 6** for setting and getting IPFS hash values, respectively.

A Consensus algorithm is a procedure in distributed systems to achieve a common agreement on particular data among multiple unreliable parties. Consensus algorithms are the crucial part of Blockchain networks. All network nodes have to agree on every new block to believe that it is the only 1 and final copy, before adding to the blockchain through the consensus protocol [10]. Popular consensus algorithms are: PoW, PoS, DPoS, PBFT and Algorand. Ethereum 2.0 phase 0 released in 2020 has been using PoS to maintain the network.

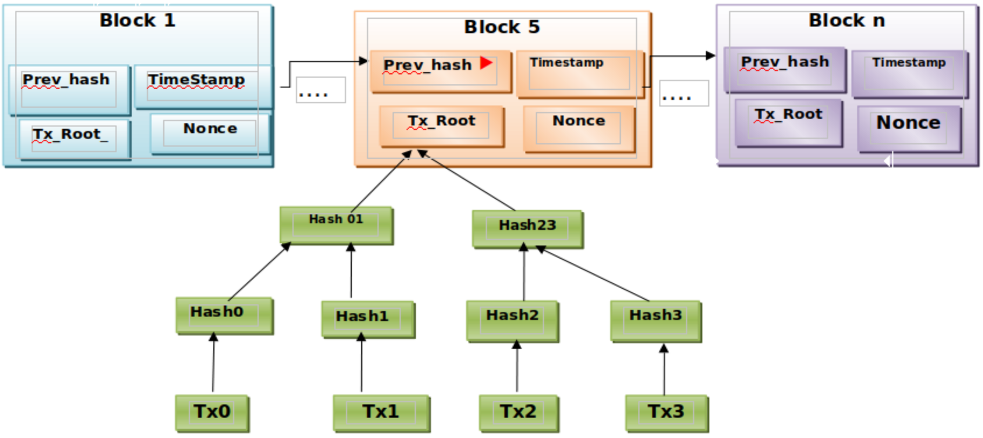


Figure 1 Blockchian internal structure (Merkle tree).

Blockchain can be classified into Public, Private or Consortium. A Public Blockchain does not need any permissions for access. Anyone can become a part of it, and can also be a validator and participate in consensus. One of the very popular public or permissionless Blockchain is Bitcoin and another 1 is Ethereum [3,6]. A Private Blockchain needs access permissions. No one could join unless they were invited by the network administrators. This type of Blockchains is being considered as a middle-ground for organizations when they are not happy with the degree of control given by public networks. Hyperledger Fabric is a popular permissioned distributed ledger platform for Enterprises. A Consortium Blockchain is said to be semi-decentralized. Some popular open-source platforms used for creating their own Consortium Blockchain are Ripple, Hyperledger, Corda, Quorum, MultiChain, Ethermint, Tendermint. Proposed framework in this paper concentrating on public blockchain using Ethereum.

Activity Recognition System (ARS) [9]: Its main purpose is to identifying the actions of people from continuing examinations on the movements of people and the conditions of the environment. Recognizing people's behavior through videos captured by multiple cameras is a difficult task. Nowadays with deep learning introduction, it made easy to detect activities based on RGB. It takes captured RGB videos as input and may recognize people's activity, activity location, start and end time of an activity in the video and video classification.

ARS can be used to monitor patient's activities from time to time and to take appropriate actions immediately after detecting irregular activities. For example, if patients suddenly face breathing problems or heart problems, at that time patient's body movements are something different than regular activities. These irregular movements can be recognized by the proposed system and give signals to take necessary further action. ARS can also be used in Traffic Monitoring Systems, where vehicle traffic signal videos are closely observed and detects any irregular traffic flow (Traffic Jam) and sends signals to control rooms to take appropriate actions immediately.

Another application of ARS can be used in houses or hospitals where old age people are taking rest alone in their rooms. Caring people are busy with their work in other rooms. If a camera is placed in an old age people room with ARS, then activities of old age people are monitored, detects irregular activities of old age people if any and give signal or alarm immediately, so that take caring person or nurses could react to protect them. Other interesting applications of ARS can be used in all important public places like airports, railway stations, bus stations, traffic monitoring places, parks, malls and theaters. These areas were already fixed with cameras. If ARS is used with these cameras, then the ARS system could react within the time for catching thieves instead of tracing them later after an incident has happened.

Cloud computing is a technology where users can avail computing resources, storage facilities, infrastructure facilities on demand through online with no maintenance cost, but only requirement is to have good Internet connectivity and bandwidth. It offers different services to the users. Cloud Computing is a blend of a number of concepts such as Service Oriented Architecture (SOA), Virtualization, etc. [1]. Because of this, many organizations are switching to cloud computing. In cloud computing it is easy to access, manage and compute user data, but it has security risks. The conventional security techniques are not enough, so the proposed system has introduced the Blockchain to store training data instead of storing in the cloud [4]. In daily life abnormal activities have been done unusually and they need complex computations to classify. Hence to make use of the benefits of cloud, all the complex video or image classification tasks in the proposed system have been computing in cloud and its classification results will be stored into the Blockchain.

Fog Computing [4]: It was initially presented by the Cisco Systems in the Internet of Things (IoT) networks as a new model to make easy transfer of wireless data to distributed devices. The purpose of fog computing is to decrease the data that needs to be transferred to the cloud for data processing and to increase the efficiency. Fog computing improves the QoS and also reduces network bandwidth. Fog is used to deliver data and place it in a node closer to the user who is at the edge of the network. Here ‘edge’ means edge of the network to which the user is connected, also known as ‘Edge Computing’. The devices (such as Raspberry Pi, routers, smart phones, personal computers etc.) are used for implementing Fog. In the proposed ARS system, Fog computing is very much helpful, since IoT devices (such as cameras, Raspberry Pi, alarm) are having low computing capability and less storage. All routine activities in daily life will be classified in the Fog itself with the help of machine learning classifying programs embedded in Raspberry Pi devices, and training databases stored in Blockchain. To read data from Blockchain, no need to pay any transaction fee. Hence Fog can easily interact with Blockchain with less time.

Machine learning (ML) algorithms are classified by learning style consisting of supervised, semi supervised and unsupervised [8]. In supervised learning, input or training data has a predefined label. Initially a classifier has to be designed with appropriate layers to train on training data and to predict the label of test data. The classifier has to tune well to get a good level of prediction accuracy. In unsupervised learning, training data does not have a label, hence the classifier is designed to group unsorted information based on similarities and differences. TensorFlow is an open-source platform for ML. It has many libraries, tools and community resources that allow developers and researchers to build and deploy ML powered applications easily [12]. The proposed framework uses an ensemble algorithm [11] to perform classification tasks in ARS. Ensemble algorithm’s main purpose is to utilize different learning algorithms to get good predictions than those that are obtained using a single learning algorithm. Ensemble algorithms combine several ML techniques into 1 model to get best results.

Organization of this remaining paper is as shown in **Figure 2**. Section 2 give literature work on Activity Recognition Systems and Blockchain with IPFS & their limitations, section 3 explains Proposed Framework for Fog based e-healthcare, section 4 demonstrates implementation of Blockchain using Ethereum and IPFS, section 5 demonstrates experimental results, finally conclusion & future work will be in section 6.

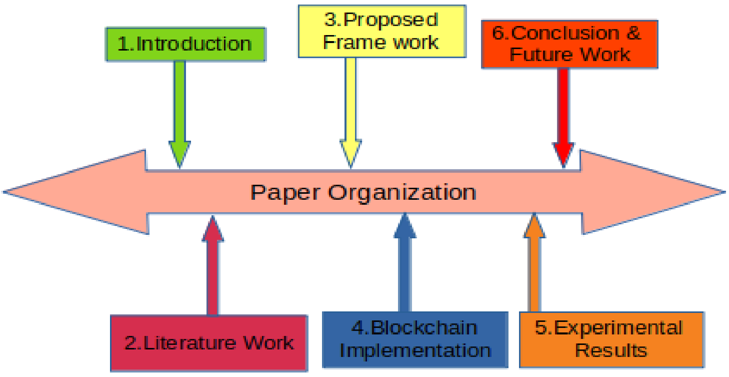


Figure 2 Paper organization.

Literature work

Recently few papers [27-37] were published using Blockchain and IPFS concepts for different real time applications, but these papers have not provided detailed information about how to store data into the Blockchain, how to connect Blockchain with IPFS and how transaction size affects to performance. Existing papers concentrated mostly on algorithms and Blockchain concepts. Nizamuddin *et al.* [33] provided Blockchain implementation details only on smart contract development and its testing using “Remix IDE”, analyzed using “ChainSecurity” smart contract analysis tool.

Blockchain is actually storing a collection of transactions in a block and a series of these blocks are called Blockchain. A transaction is only capable of storing small of amounts of information within a Blockchain and transaction fee (gas) increases as data size increases. Hence an alternative solution to store large amounts of data in Blockchain is possible with help of IPFS in the area of especially healthcare applications.

Many existing works [1,9,13] have been proposed on the Activity Recognition System by using either Blockchain or Fog computing but not considering both. Existing works specify that once data is stored in Blockchain that can’t be modified, but not concentrated on size of the data that is feasible to store in Blockchain and impact of gas (transaction) fees on data size. In the paper [1] both Fog and Blockchain for ARS have been used but the internal details of Blockchain (i.e. data size, performance issues and transaction fees) has not been covered. Hence in the proposed work, Fog has been used to reduce the network bandwidth; and Blockchain has been used to store arbitrary sized training data to avoid forging and also covered data size versus gas fees, throughput and delay comparisons in Ethereum Blockchain with implementation details.

Proposed framework for activity recognition

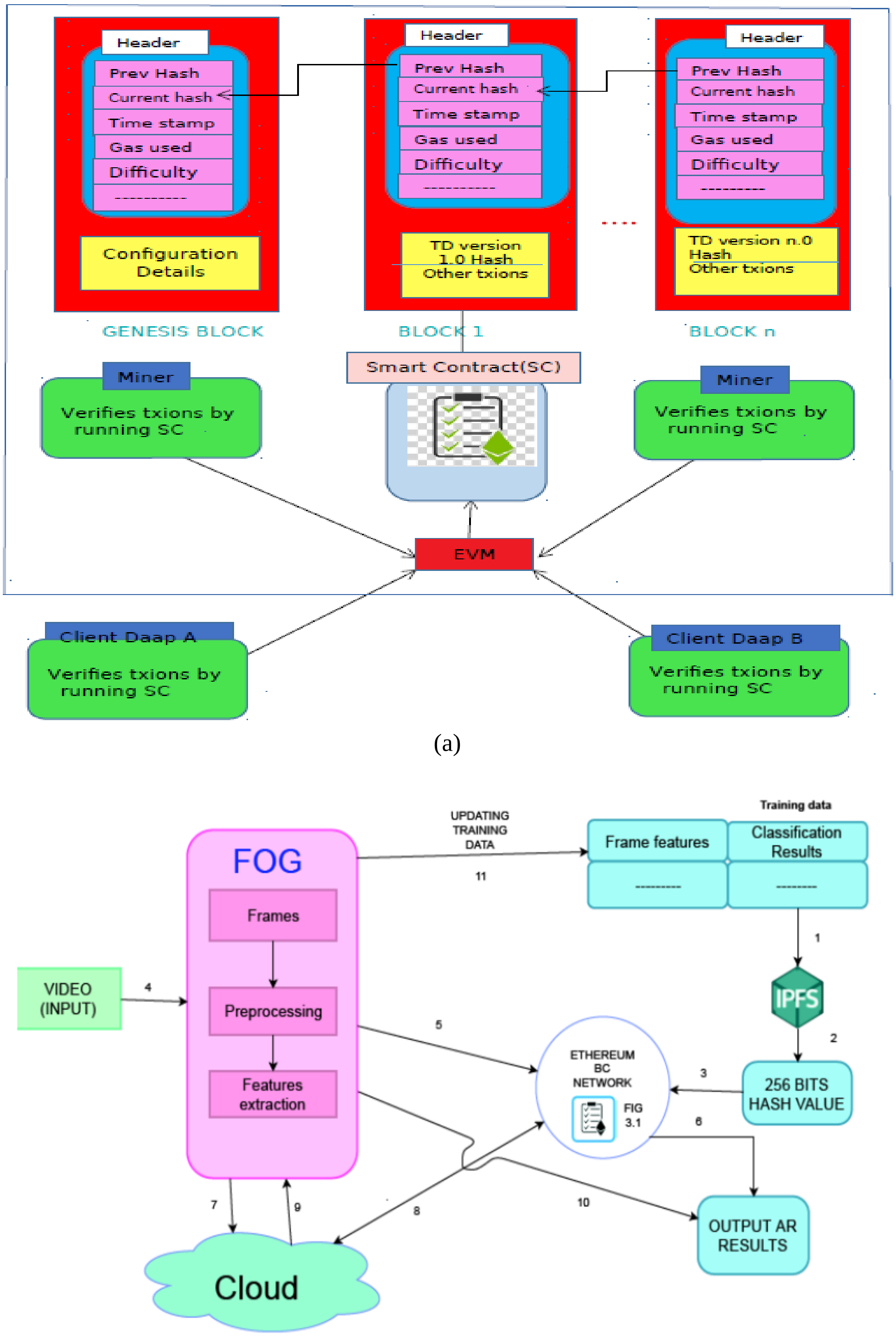
Overall architecture of the proposed Blockchain and Fog based e-healthcare ARS is shown in **Figures 3(a)** and **3(b)**. **Figure 3(a)** named Ethereum Blockchain (BC) Network is part of **Figure 3(b)**. As shown in **Figure 3(b)**, input video (arrow with label 3) is captured by cameras which are fixed in Hospitals or where there is a need to monitor and this video is given as input for Fog devices (Ex: Raspberry Pi IoT Device). Fog device divides the video into frames and performs preprocessing operations on it to extract features that contain significant human activity. Fog searches with these features in Blockchain for getting video classification results. In the proposed system, training data has been maintained in Blockchain to avoid modifications by either active or passive attackers in future.

Fog provides classification results directly if it finds “frame features” in Blockchain training data. Otherwise, it requires more computations to perform classification which cannot be computed in Fog due to its limited capability, so only those video frames are uploaded through internet to cloud computing data centers, then in cloud different machine learning algorithms (Ex: Ensemble algorithms) are applied on it for classifying the given video or frames. Cloud may interact with Blockchain for training data if it requires for classification purposes. After classification, results will be send back to Fog by cloud.

Then Fog adds this new record (consists of frame features and classification result) into the Blockchain training data and provides same results as output to the user. The overview of sequence of operations in the proposed framework of ARS has been shown in **Figure 4**. Fog computing decreases the delay of uploading large volumes of data to the cloud, decreases bandwidth consumption, and associated costs with it. Any ARS performs the 2 basic tasks, i.e., detection of human action and recognizing the class of action. For many human activities, some features or properties are common. In our proposal, the feature descriptors are acquired from key frames of the training data [1].

Figure 3(a) represents Ethereum Blockchain (BC) network consists of nodes which maintain Blockchain, miners, smart contracts and Ethereum Virtual Machine (EVM). Nodes present in the BC network are 2 types, which are light nodes and full nodes. Light nodes can perform operations like initiate transactions and querying Blockchain in the Ethereum network but do not store the entire Blockchain ledger in it. Full nodes maintain entire Blockchain ledger from genesis block (initial block) to the latest block and can perform all operations as done by light nodes. Each block in BC consists of 2 parts which are header and transaction parts.

In **Figure 3(a)**, Header consists of fields like previous block hash, current block hash, nonce, time stamp, gas used, gas limit, difficulty, miner details and others. Transaction part consists of all the transactions accommodated in that block and those transaction details. Each transaction consists of fields like data, account details like from and to addresses, timestamp, gas used, gas limit etc. Miners are responsible to collect transactions from the network and to create a block by combining valid transactions whose gas price is acceptable with current market rate. Created blocks are broadcast to the network and all full nodes add these blocks into their existing Blockchain ledger. Miners validate all receiving transactions with the rules present in smart contract (SC). EVM is the decentralized processing unit of the Ethereum network. EVM is responsible for executing smart contract byte code. Every node in the Ethereum network runs an EVM instance.



(a)

(b)

Figure 3 (a) Ethereum Blockchain network, (b) Frame work of proposed Activity Recognition System (ARS).

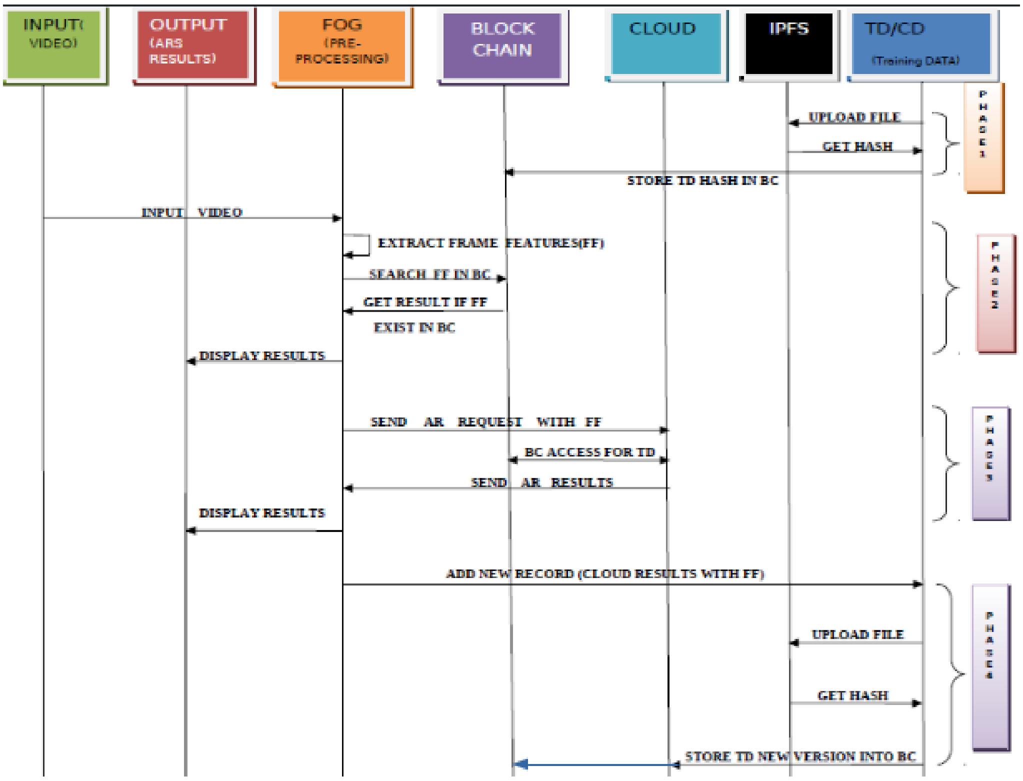
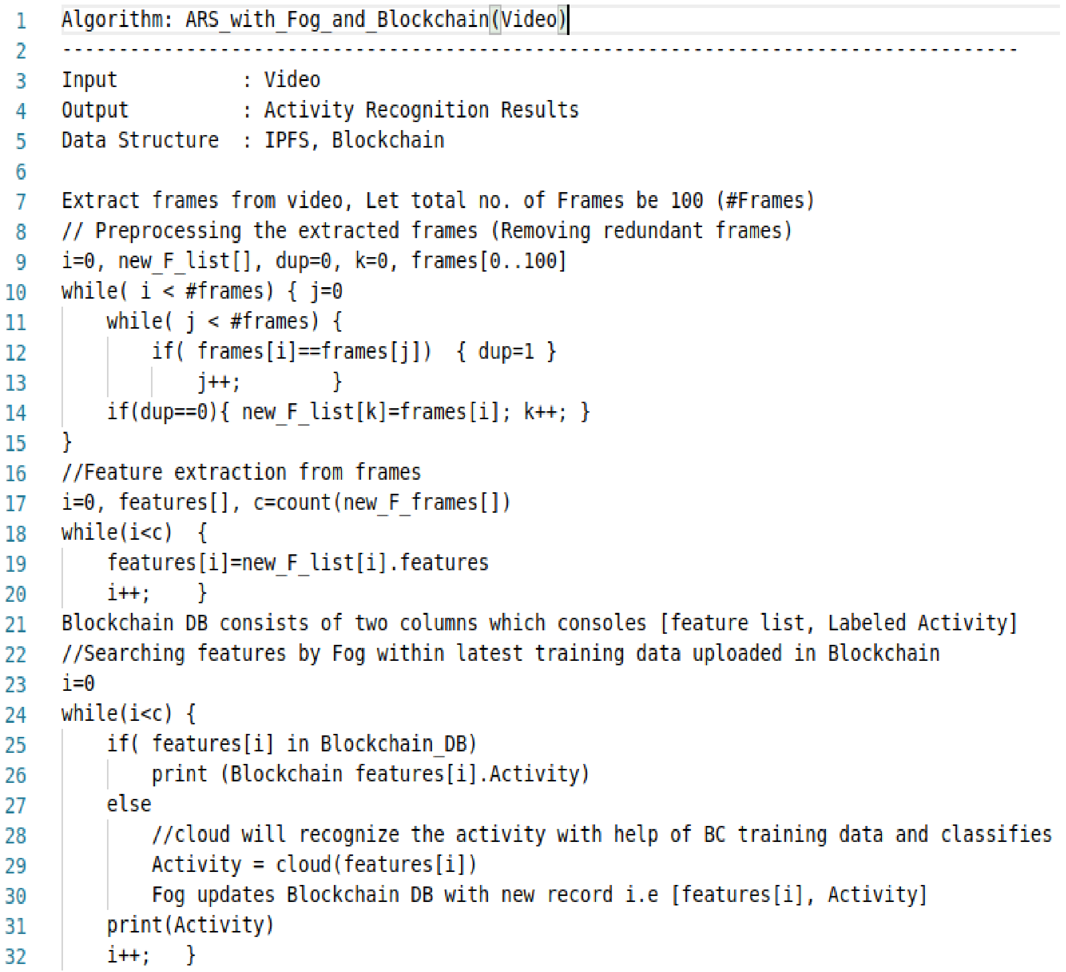


Figure 4 Sequence diagram for proposed ARS framework.

Sequence of steps to be performed in the proposed Activity Recognition System has been demonstrated in **Figure 4**. These steps are divided into 4 phases. Phase 1 represents the initial version of Training Data (TD) or Classified Data (CD) is stored into the Blockchain by sending TD file to IPFS, in turn it returns corresponding 256-bit hash value, this will be sent to Blockchain via initiating a transaction. In phase 2, input video is provided to Fog for activity recognition. Fog initially performs preprocessing operations on it to extract frame features from input video. Preprocessing operations involves dividing a given video into frames, removing duplicates frames and extracting frame features. Then Fog searches these extracted frame features in Blockchain uploaded latest Training Data to recognize corresponding activity. Fog displays activity recognition results if it is found in Blockchain TD, since TD consists of frame features and corresponding activity as a classified label. Fog initiates Phase 3 only when it could not find frame features in training data present in Blockchain. In this phase Fog sends Activity Recognition (AR) requests to the cloud with frame features as input for classification. Cloud will classify the frame features by applying ensemble algorithms on it with help of the latest version TD which has been uploaded into Blockchain.

Now cloud sends classification results to Fog for the given input frame features. Fog displays this classification results as an AR output. Phase 4 will be executed only when Phase 3 is performed. Phase 4 will update training TD with a new set of frame features and corresponding classification results which were obtained as output of phase 3. This updated TD is again given to IPFS to get its hash value and this hash value will be stored in Blockchain as a latest version of TD. The detailed algorithm for the proposed system is also shown in **Figure 5**.



```
Algorithm: ARS_with_Fog_and_Blockchain(Video)
-------------------------------------------------------------------------------------
Input          : Video
Output         : Activity Recognition Results
Data Structure  : IPFS, Blockchain

Extract frames from video, Let total no. of Frames be 100 (#Frames)
// Preprocessing the extracted frames (Removing redundant frames)
i=0, new_F_list[], dup=0, k=0, frames[0..100]
while( i < #frames) { j=0
    while( j < #frames) {
        if( frames[i]==frames[j])  { dup=1 }
            j++;        }
    if(dup==0){ new_F_list[k]=frames[i]; k++; }
}
//Feature extraction from frames
i=0, features[], c=count(new_F_frames[])
while(i<c)  {
    features[i]=new_F_list[i].features
    i++;    }
Blockchain DB consists of two columns which consoles [feature list, Labeled Activity]
//Searching features by Fog within latest training data uploaded in Blockchain
i=0
while(i<c) {
    if( features[i] in Blockchain_DB)
        print (Blockchain features[i].Activity)
    else
        //cloud will recognize the activity with help of BC training data and classifies
        Activity = cloud(features[i])
        Fog updates Blockchain DB with new record i.e [features[i], Activity]
    print(Activity)
    i++;   }
```

Figure 5 Algorithm: ARS with fog and blockchain.

Blockchain implementation using ethereum

Development tools and technologies

The proposed paper implements Ethereum Blockchain with necessary tools and technologies as shown in **Table 1**. The detailed description of above tools is explained as follows:

Ethereum [16,17] is a public, Blockchain based distributed computing platform that used to create and run decentralized applications (DApps) that enables users to make agreements and conduct transactions directly with each other.

Gas [39] represents the fee required to execute a transaction on the Ethereum network, it should be used with caution. Sender have to specify a gas limit for each transaction submit into the Ethereum network for its execution. The gas limit is the MAX amount that we are ready to pay as transaction fee. The miners have few options after receiving transactions that they could either accepting, decline or refund. First, they can accept the transaction to process if they satisfy with the proposed gas fee by the sender. Second, they may reject the transaction if proposed gas fee is less than compared with market cost. If the sender proposed fee is higher than gas limit, then the miner can refund the remaining gas to the sender. Ether can be represented in smaller unit called Gwei (Giga Weight). Gwei denotes the ninth power of a fractional Ether, so it is also called as nanoether. One Gwei is equal to 0.000000001 Ether. To store 256-bit word into the Blockchain, it would cost 20,000 gas and it requires 640,000 gas to store 1kb of data. Gas cost right now is around 20 Gwei (0.00000002 ETH) as of April 2020.

Most of the people have been using default gas prices when sending a transaction. Even though it is acceptable to use the default gas price, sometimes it is preferable to pay more if you want fast transaction confirmation. Sometimes it is useful to pay less gas price if you don’t want quick confirmations. Hence it is important to monitor the network to know market gas cost [39]. In reality it is costly to store a high volume of data on the Ethereum Blockchain. It is around 0.01 Ether to store 1 kB of data as per Ethereum's yellow paper [38]. To store 1 Mb data, its cost is 10 Ether (0.01 Ether/kB×1000 kB = 10 Ether) at $860/Ether = $8600.00. To store IGB of data, it would cost $8,600,000.00 on the Ethereum Blockchain. To store Ethereum's yellow paper PDF (520Kb), it costs $4472 USD. As per these statistics, it would be more cost to store huge volume of data directly into Blockchain. Hence possible solution for this problem is usage of IPFS.

Table 1 Development environment for the Ethereum blockchain.

Component	Description
CPU	Intel Core i3-4010U @ 1.70GHz
Memory	8 GB
Operating Systems	Ubuntu Linux 18.04.1 LTS
Editor	Visual Studio Code
Metamask	v7.7.9
Node.js	v12.13.0 (JavaScript Run time environment)
npm	Node Package Manager for Node.js
Libraries	web3.js, React .js
IDE	Truffle v5.0.5
Database	IPFS and Ethereum Blockchain
Virtual Blockchain	Ganache-2.1.2-linux-x86_64
Real Ethereum Network	Ropsten Test Network
Programming Languages	Java Script, Solidity v0.5.0

IPFS [16], its full form is InterPlanetary File System. It is a protocol and peer to peer network for storing and sharing data in a distributed file system. If we provide any length of file or any type of file as an input to IPFS, then it returns a corresponding hash value of 256 bits as an output. That means IPFS takes arbitrary sized input and produces fixed size output. Even if we change 1 character in the input file, IPFS will provide a different hash value. The original data is initially stored in IPFS, which generates the hash value. This hash value can be stored in Blockchain instead of actual data. This hash value is included in a transaction and then this transaction is added into a block in Blockchain after validation with respect to smart contract rules.

IPFS Content is accessible by peers positioned anywhere in the world. IPFS follows 3 fundamental principles, which are 1.CID (Content Identifier for unique identification or addressing), 2. Merkle DAGs (Directed Acyclic Graphs for Content linking), 3.DHT (Distributed Hash Tables for content discovery). To understand content addressing, it is like asking a book in library by its title, but not by its location (if book location changed, then you could not find that book). Merkle DAGs are similar to Merkle Trees and they can be constructed from the leaves. Merkle DAG nodes are immutable, any change in a node would alter its identifier (CID) and thus affect all other nodes which are present above on it in the DAG. IPFS uses a DHT (distributed hash table) is a data structure with "key to value" pair. A distributed hash table is split across all the peers in a distributed network. To find files which are requested by users, IPFS uses libp2p service which is part of the IPFS.

Ethereum test networks vs Main networks: [18] Ethereum Testnet uses similar technology and software as the Ethereum "Mainnet". However, the Ethereum Mainnet network is used for "actual" transactions with "value", Testnets are used for testing smart contracts (SC) and DApps. There are multiple test networks available which are Ropsten, Rinkeby and Kovan. It is important to remember that all Ethers within these Testnet networks are not worth anything since this is only for testing. You can use any DApp connected to either one of these above test networks without worrying about real money loss. But to perform transactions in Ethereum Main network requires sufficient ether balance in our account for gas reduction purposes.

Metamask [19] is an Ethereum wallet and an extension in our Browser for accessing Ethereum enabled DApps. It provides a secure interface to users when a DApp wants to perform transactions in the Blockchain and it allows the user to create and manage their own identities via private keys.

React [20] (also known as React.js or ReactJS) is an open source, flexible and efficient JavaScript library for building UI (user interface). It permits us to create reusable UI components. React allows us to develop mobile apps or single page apps from reusable UI components.

Truffle [21] is a development environment and testing framework for Ethereum to make developer work easier. Truffle provides the facilities to create, compile, deploy and test Blockchain DApps. Other truffle features are

1) Automated SC testing with Chai and Mocha.
2) Incorporated features like compilation, linking and deployment of SC.
3) Interactive console for direct SC communication.
4) Scriptable deployment and migration framework.
5) Instant rebuilding of assets during development.
6) Network management for deploying to many private & public networks.

Node.js [22] is an open source, cross platform and JavaScript run time environment that executes JavaScript code outside a web browser. Node.js represents a "JavaScript everywhere" paradigm, means same software can be used for both server and client-side applications. "Node.js" is a product name but not a filename with the java script extension "js".

NPM: [22] Its full form is Node Package Manager. It is the default package manager for a Node (Node.js). It put down all modules in a place so that node can detect them, and supervise "dependency conflicts" in a genius way.

Web3.js Library [23] programmers to communicate with the Ethereum Blockchain through any Ethereum node (either local node or a node hosted by the DApp provider or public gateways such as Infura) that allows access via HTTP. The usage of Metamask along with Web3.js is the standard method to integrate web browser applications with Ethereum.

Ganache [24] permits us to perform all operations on Ethereum Blockchain MainNet without the cost. Hence developers prefer to use ganache while developing to test their smart contracts. It provides a built-in block explorer.

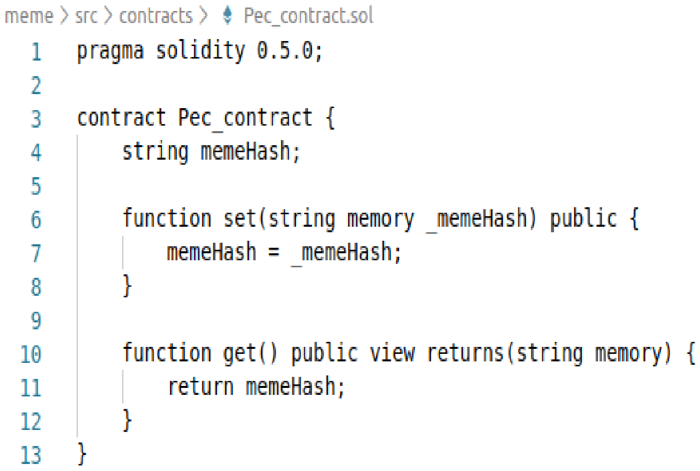
Solidity [25] is an object oriented, easy to use, statically typed programming language designed for developing smart contracts that run on Ethereum Virtual Machine (EVM). Solidity program generates bytecode after compilation which is executable on EVM. Solidity can be used to write SC's for various Blockchain platforms especially on Ethereum.

Dependency software's installation

To develop DApp for Ethereum Blockchain with IPFS, few online tutorials are available [14,15]. As per those tutorials, initial code has been downloaded from github (https://github.com/dappuniversity/starter_kit) and did few modifications as per proposed work requirements. One such modification is our DApp connected to a real Ethereum Blockchain network called Ropsten Test Network instead of using Ganache Blockchain simulator. Initially we tested with Ganache simulator, then later updated code to connect with real Blockchain networks. Another modification is, we included extra "text area" to display the hash link to verify Blockchain uploaded data. Then we changed the smart contract name from "Meme.sol" to "pec_contract.sol". After downloading the initial code into the meme directory, we installed dependency softwares node.js, ganache, truffle, npm, IPFS and Metamask.

Smart contract deployment

Smart contract file named pec_contract.sol created in contract sub directory using solidity high level programming language and this file extension must be ".sol". This smart contract file consists of 2 functions which are set() and put() as shown in **Figure 6**. Set() used to initialize hash value to the memeHash variable and get() used to retrieve hash value from the memeHash variable. After successful compilation of smart contract, it can be deployed into the Blockchain as shown in **Figure 7**.



```
pragma solidity 0.5.0;

contract Pec_contract {
    string memeHash;

    function set(string memory _memeHash) public {
        memeHash = _memeHash;
    }

    function get() public view returns(string memory) {
        return memeHash;
    }
}
```

Figure 6 Smart contract written in solidity language.

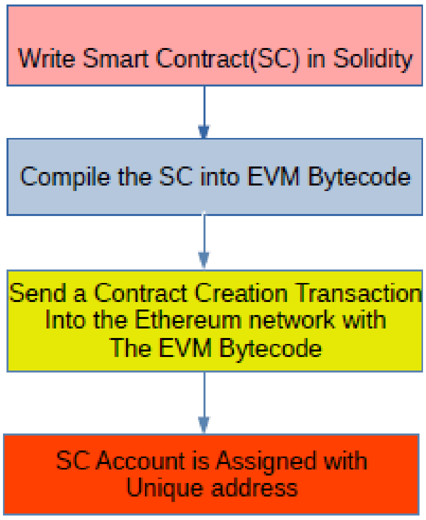

Write Smart Contract(SC) in Solidity
Compile the SC into EVM Bytecode
Send a Contract Creation Transaction Into the Ethereum network with The EVM Bytecode
SC Account is Assigned with Unique address

Figure 7 Smart contract deployment.

After compilation of smart contract (SC), EVM (Ethereum Virtual Machine) Byte code will be produced. New transaction is created with this EVM Byte code and send to Ethereum network to store SC into Blockchain. Once transaction is confirmed, a unique address is assigned to the deployed SC and this address is returned to the UI. Users can access smart contract with this unique address only.

Blockchain dapp architecture

The entire DApp Blockchain system forms a 3-level architecture where the front-end user interface (UI) is responsible for user interaction while the IPFS and web3 libraries are responsible for API calls in between front end and back end. **Figure 8** shows a complete view of the DApp architecture. The front-end UI, which allows us to select training data or any type of file and passing it to the IPFS. Then IPFS returns the corresponding hash value of submitted input training data. Using this hash value, new transaction message is created and send to Blockchain through the web3 library. Transaction conformation details are received by web3 library from Blockchain and sent back to the UI. The web3 library can interact with the Ethereum Blockchain system for function calls and smart contract deployments.

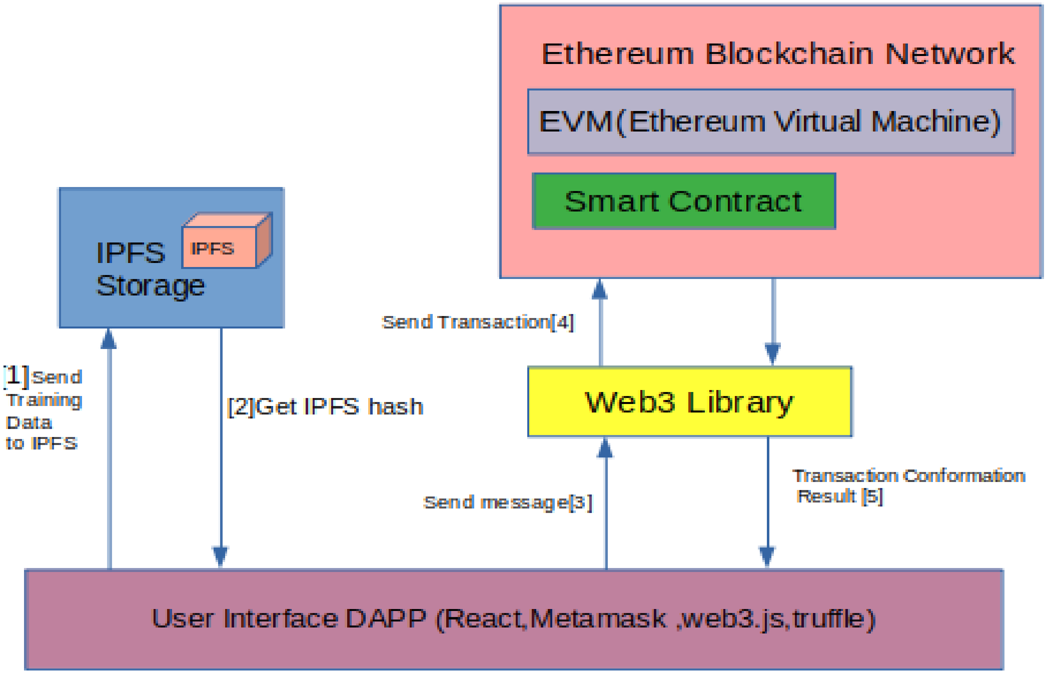

Ethereum Blockchain Network
EVM(Ethereum Virtual Machine)
Smart Contract
IPFS Storage
IPFS
Send Transaction[4]
[1]Send Training Data to IPFS
[2]Get IPFS hash
Web3 Library
Send message[3]
Transaction Conformation Result [5]
User Interface DAPP (React,Metamask ,web3.js,truffle)

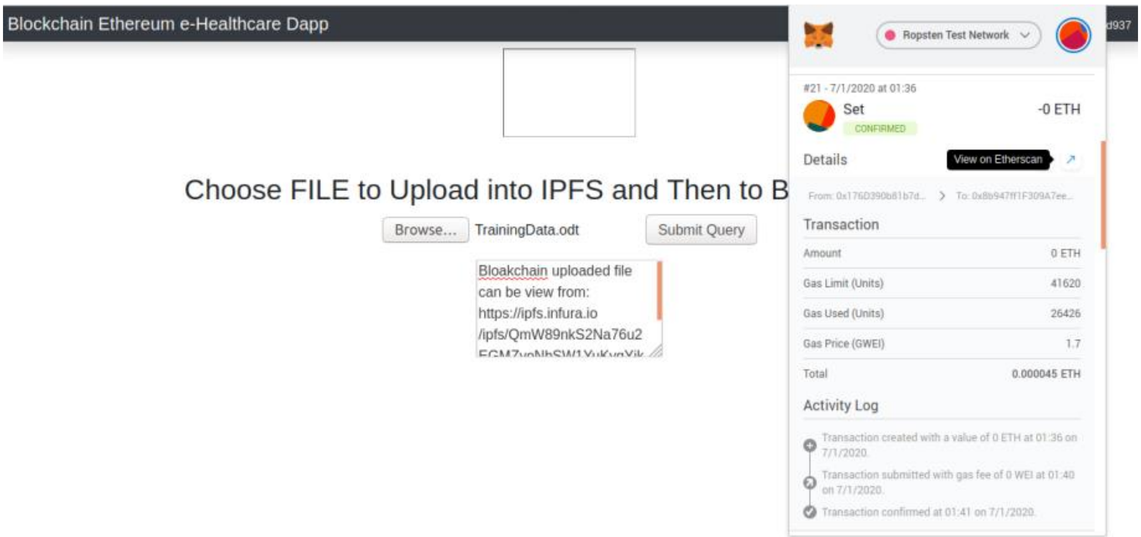
Figure 8 Blockchain DApp architecture.

Connecting DApp to the real Ethereum Blockchain network

Connecting DApp to the Real Ethereum Blockchain Network is possible through Infura. Infura is used by Metamask. Infura communicates with the Ethereum Blockchain and runs nodes on behalf of its users [26].

After successful deployment of smart contract into Ethereum Blockchain Ropsten Test Network, for overall execution first login into Metamask from browser, next in command line, run the command $npm run start from meme directory. It will compile and run App.js file. Then DApp user interface will be opened from browser with http://localhost:3000/. Now it prompts us to connect to Metamask. In DApp user interface, where it allows us to choose training data file (or any file type either document or image or audio or video type) to upload into IPFS and press submit button. Text area present in DApp UI shows link which consists of IPFS hash value of uploaded training data file. With this hash value new transaction is created and submit to Ropsten Test Network (Ethereum Blockchain network) to store into Blockchain. For this transaction, “From” account is our account number which was created at the time of Metamask creation, “To” Account is smart contract deployed account number. Now Metamask shows us total gas fees estimated to submit this new transaction, then if we press the conform button then the transaction is submitted to the network. It will take few seconds to confirm transaction. After this transaction is successfully added into Blockchain then it shows us transaction confirmation details like transaction number, block number, gas used, date & time etc as shown in **Figure 9**.

Sender may also verify our Blockchain uploaded hash value in the field “Input Data” from “https://ropsten.etherscan.io/” website by entering Transaction ID as an input.



Blockchain Ethereum e-Healthcare Dapp

Choose FILE to Upload into IPFS and Then to B

Bloakchain uploaded file can be view from: https://ipfs.infura.io/ipfs/QmW89nkS2Na76u2

#21 - 7/1/2020 at 01:36

Set -0 ETH

CONFIRMED

Details

From: 0x176D390b81b7d... > To: 0x8b947ff1F309A7ee...

Transaction

Amount 0 ETH

Gas Limit (Units) 41620

Gas Used (Units) 26426

Gas Price (GWEI) 1.7

Total 0.000045 ETH

Activity Log

Transaction created with a value of 0 ETH at 01:36 on 7/1/2020.

Transaction submitted with gas fee of 0 WEI at 01:40 on 7/1/2020.

Transaction confirmed at 01:41 on 7/1/2020.

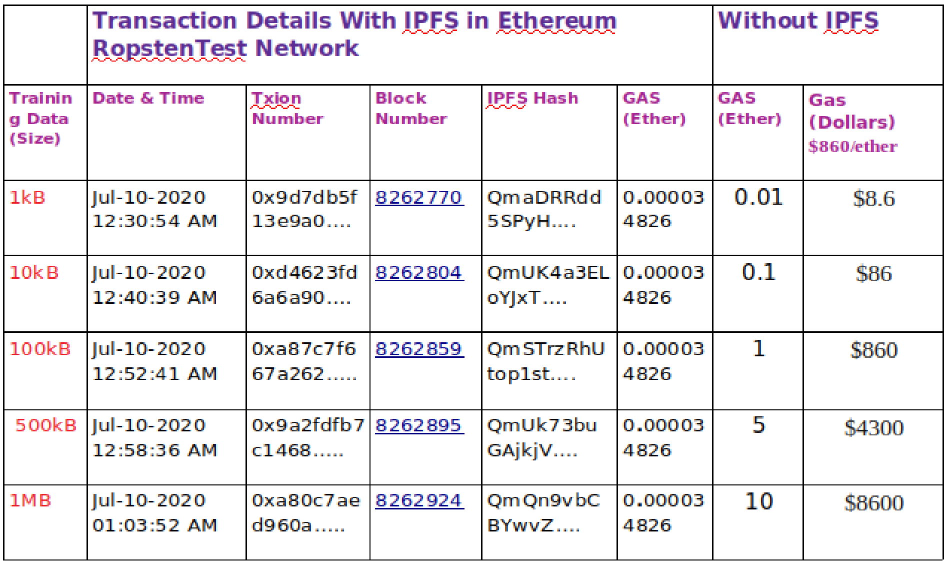
Figure 9 Transaction conformation details displayed in Metamask.

Experimental results (comparing transaction size vs cost, throughput, delay in Ethereum networks):

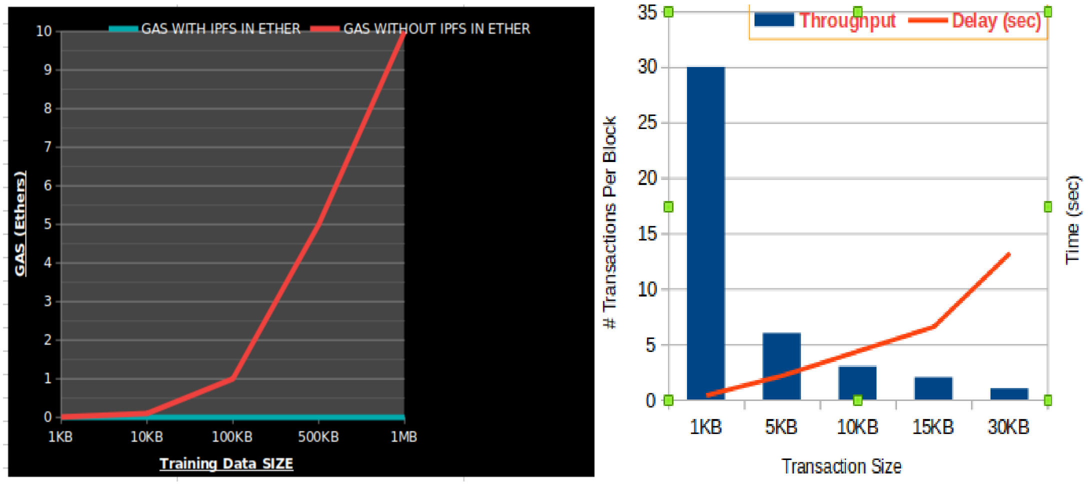
Our Blockchain implementation results with and without IPFS are represented in **Table 2** and comparison results are shown in **Figure 10**. We experiment by uploading different sizes of training data into IPFS in turn it gives a uniform size (256 bit) unique hash value to upload into Blockchain. Hence with IPFS, for any size of training data, the same amount of gas is used to upload into Blockchain, but without IPFS different gas is required to store different sizes of data into Blockchain. **Figure 10(A)** shows clearly that without IPFS, if data size increases, then required gas also increases proportionally. With IPFS, the required gas value is very less and it doesn't change for different sizes of training data to upload into Blockchain.

In Ethereum currently the maximum block size (30 KB) is limited to 1,500,000 Gas [40]. Simple transactions in Ethereum will require 21,000 Gas, so it is possible to accommodate around 70 transactions (1,500,000/21,000) into a single block. If transaction size increases, then it consumes more gas, therefore few transactions (less than 70) only can fit into each block, ultimately it reduces overall throughput (transactions confirmations per second).

Table 2 Transaction cost details with and without IPFS in Ethereum network.



	Transaction Details With IPFS in Ethereum RopstenTest Network					Without IPFS	
Training Data (Size)	Date & Time	Txion Number	Block Number	IPFS Hash	GAS (Ether)	GAS (Ether)	Gas (Dollars) $860/ether
1kB	Jul-10-2020 12:30:54 AM	0x9d7db5f 13e9a0....	8262770	QmaDRRdd 5SPyH....	0.00003 4826	0.01	$8.6
10kB	Jul-10-2020 12:40:39 AM	0xd4623fd 6a6a90....	8262804	QmUK4a3EL oYJxT....	0.00003 4826	0.1	$86
100kB	Jul-10-2020 12:52:41 AM	0xa87c7f6 67a262.....	8262859	QmSTrzRhU top1st....	0.00003 4826	1	$860
500kB	Jul-10-2020 12:58:36 AM	0x9a2fdfb7 c1468.....	8262895	QmUk73bu GAjkjV....	0.00003 4826	5	$4300
1MB	Jul-10-2020 01:03:52 AM	0xa80c7ae d960a.....	8262924	QmQn9vbC BYwvZ....	0.00003 4826	10	$8600



(A) (B)

Figure 10 (A) Comparison between data size and transaction cost, and (B) Transaction size vs throughput and delay without IPFS.

Miners interested in including transactions into a block preferably with more gas price. In Ethereum the average block confirmation time is 13.26 s [40]. If you are not willing to pay more prize per required gas, then transactions with more size have to wait a long time for confirmation, which means taking more delay. **Figure 10(B)** shows transaction size versus throughput and delay without using IPFS. As each block size is limited to a maximum of 30 KB and Block confirmation time is 13.26 s in Ethereum, 1 block can accommodate 30 transactions (with IKB of size each), then each transaction requires confirmation time is 0.442 s i.e (13.26/30). For 5KB size transactions, conformation time would be 2.21 s (ie 5×6 = 30 KB, so 13.26/6) and so on. Based on these statistics, without usage of IPFS, if transaction size increases, throughput decreases and delay increases proportionally. With IPFS, irrespective of transaction size, transaction confirmation time is 0.18 s (3.12/70).

Conclusions

This paper provides new framework for activity recognition system in e-healthcare applications based on Blockchain and Fog computing. This framework provides good security for training data which have been using in machine learning algorithms because proposed work is storing training data in Blockchain which cannot be altered once stored. The implementation for the Blockchain is demonstrated in detail using Ethereum and IPFS. Our results proved that without usage of IPFS in Blockchain, especially in the field of healthcare, the transaction throughput decreases and transaction confirmation delay increases and also shows that with usage of IPFS, all transactions can have the same size, thereby improving transaction throughput and decreasing confirmation delay. The implementation of machine learning and Fog-part will be considered as our future work in the coming days. Fog implementation needs to concentrate on 3 platforms such as software (runC, Docker), hardware (Raspberry Pi3 along with configurations of hostapd and dnsmasq to connect Wi-Fi) and development platforms. Limitation of this work is to Blockchain DApp implementation requires to concentrate on many software installations, it needs to simplify. This paper is very much helpful for the beginners who what to learn public Blockchain implementation to store their data using IPFS.

Acknowledgment

We are very much thankful to Saurav Shekhar (Independent Blockchain & DAO Researcher from Pune, India) who has helped us more while implementing Ethereum public Blockchain.

References

[1] N Islam, Y Faheem, IU Din, M Talha, M Guizani and M Khalil. A Blockchain-based fog computing framework for activity recognition as an application to e-healthcare services. *Future Generat. Comput. Syst.* 2019; **100**, 569-78.

[2] W Wang, DT Hoang, P Hu, Z Xiong, D Niyato, P Wang, Y Wen and DA Kim. A survey on consensus mechanisms and mining strategy management in Blockchain networks. *IEEE Access* 2019; **7**, 22328-70.

[3] Z Zheng, S Xie, HN Dai, X Chen and H Wang. Blockchain challenges and opportunities: A survey. *Int. J. Web Grid Serv.* 2018; **14**, 35275.

[4] S Kumari and S Singh. Fog computing: Characteristics and challenges. *Int. J. Emerg. Trends Tech. Comput. Sci.* 2017; **6**, 113-7.

[5] AS Bruyn. 2017, Blockchain an introduction. University Amsterdam, Amsterdam, Netherlands.

[6] Y Yuan and FY Wang. Blockchain and cryptocurrencies: Model, techniques, and applications. *IEEE Trans. Syst. Man Cybern. Syst.* 2018; **48**, 1421-8.

[7] AP Joshi, M Han and Y Wang. A survey on security and privacy issues of Blockchain technology. *Am. Inst. Math. Sci.* 2018; **1**, 121-47.

[8] K Das and RN Behera. A survey on machine learning: Concept, algorithms and applications. *Int. J. Innovat. Res. Comput. Comm. Eng.* 2017; **5**, 1301-9.

[9] X Li, T Pang, W Liu and T Wang. Fall detection for elderly person care using convolutional neural networks. *In*: Proceedings of the 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics, Shanghai, China. 2017, p. 1-6.

[10] Cointelegraph, Available at: https://cointelegraph.com/bitcoin-for-beginners/how-Blockchain-technology-works-guide-for-beginners, accessed July 2020.

[11] TG Dietterich. Ensemble methods in machine learning, Available at: http://web.engr.oregonstate.edu/~tgd/publications/mcs-ensembles.pdf, accessed July 2020.

[12] Tensorflow, Available at: https://www.tensorflow.org, accessed July 2020.

[13] T McGhin, KKR Choo, CZ Liu and D He. Blockchain in healthcare applications: Research challenges and opportunities. *J. Netw. Comput. Appl.* 2019; **135**, 62-75.

[14] L Audhikesavan. Step by step approach to create DApp - using Ethereum, ReactJS & IPFS - part 1, Available at: https://medium.com/coinmonks/step-by-step-approach-to-create-dapp-using-Ethereum-reactjs-ipfs-part-1-42ea4cf69488, accessed July 2020.

[15] M Chan. Build a simple Ethereum + InterPlanetary File System (IPFS) + React.js DApp, Available at: https://itnext.io/build-a-simple-Ethereum-interplanetary-file-system-ipfs-react-js-dapp23ff4914ce4e, accessed July 2020.

[16] P Mundhe. Ethegram - an Ethereum and IPFS-based decentralized social network system. *Int. Res. J. Eng. Tech.* 2020; **7**, 1978-82.

[17] Ethereum, Available at: https://Ethereum.org/en/whitepaper, accessed July 2020.

[18] Hummingbot, Available at: https://medium.com/hummingbot/finance-3-0-wiki-testnet-vs-mainnet-8ab5b78d93, accessed July 2020.

[19] P Long. How to install and use Metamask, Available at: https://blog.wetrust.io/how-to-install-and-use-metamask-7210720ca047, accessed July 2020.

[20] React, Available at: https://reactjs.org, accessed July 2020.

[21] Trufflesuite, Available at: https://www.trufflesuite.com/truffle, accessed July 2020.

[22] Node, Available at: https://nodejs.org/en/docs, accessed July 2020.

[23] Mycryptopedia, Available at: https://www.mycryptopedia.com/what-is-web3-js-a-detailed-guide, accessed July 2020.

[24] Codementor, Available at: https://www.codementor.io/@swader/developing-for-Ethereum-getting-started-with-ganache-l6abwh62j, accessed July 2020.

[25] Solidity, Available at: https://solidity.readthedocs.io/en/ v0.6.10, accessed July 2020.

[26] H Kang, Available at: https://medium.com/coinmonks/deploy-your-smart-contract-directly-from-truffle-with-infura-ba1e1f1d40c2, medium.com/coinmonks, accessed July 2020.

[27] X Wu, Y Han, M Zhang and S Zhu. *Secure personal health records sharing based on Blockchain and IPFS*. *In*: W Han, L Zhu and F Yan (Eds.). Communications in computer and information science. Vol 1149. Springer, Singapore, 2020, p. 340-54.

[28] I Podsevalov, O Iakushkin, R Kurbangaliev and V Korkhov. *Blockchain as platform for Fog computing*. *In*: S Misra, O Gervasi, B Murgante, E Stankova, V Korkhov, C Torre, AMAC Rocha, D

Taniar, BO Apduhan and E Tarantino (Eds.). Computational science and its applications - ICCSA 2019. Springer, Cham, 2019, p. 596-605.

[29] N Nizamuddin, HR Hasan and K Salah. *IPFS - Blockchain-based authenticity of online publications*. *In*: S Chen, H Wang and LJ Zhang (Eds.). Blockchain - ICBC 2018. Springer, Cham, 2019, p. 199-212.

[30] SH Jang, J Guejong, J Jeong and B Sangmin. *Fog computing architecture based Blockchain for industrial IoT*. *In*: JMF Rodrigues, PJS Cardoso, J Monteiro, R Lam, VV Krzhizhanovskaya, MH Lees, JJ Dongarra and PMA Sloot (Eds.). Computational science - ICCS 2019. Springer, Cham, 2019, p. 593-606.

[31] S Wang, Y Zhang and Y Zhang. A blockchain-based framework for data sharing with fine-grained access control in decentralized storage systems. *IEEE Access* 2018; **6**, 38437-50.

[32] M Naz, FA Al-zahrani, R Khalid, N Javaid, AM Qamar, MK Afzal and M Shafiq. A secure data sharing platform using blockchain and interplanetary file system. *Sustainability* 2019; **11**, 7054.

[33] N Nizamuddin, K Salah, M Ajmal Azad, J Arshad and MH Rehman. Decentralized document version control using ethereum blockchain and IPFS. *Comput. Electr. Eng.* 2019; **76**, 183-97.

[34] A Saqib, G Wang, B White and RL Cottrell. A blockchain-based decentralized data storage and access framework for PingER. *In*: Proceedings of the 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/ 12th IEEE International Conference on Big Data Science and Engineering, New York, United States. 2018, p. 1303-8.

[35] I Jovović, S Husnjak, I Forenbacher and S Maček. Blockchain and IPFS: A general survey with possible innovative applications in industry 4.0. *In*: Proceedings of the 3rd EAI International Conference on Management of Manufacturing Systems, Dubrovnik, Croatia. 2018, p. 1-10.

[36] JT Hao, Y Sun and H Luo. A safe and efficient storage scheme based on blockchain and IPFS for agricultural products tracking. *J. Comput.* 2018; **29**, 158-67.

[37] A Rajalakshmi, KV Lakshmy, M Sindhu and PP Amritha. A Blockchain and IPFS based framework for secure Research record keeping. *Int. J. Pure Appl. Math.* 2018; **119**, 1437-42.

[38] G Wood. *Ethereum: A secure decentralised generalised transaction ledger EIP-150 revision.* International Information Technology University, Almaty, Kazakhstan, 2017.

[39] P Garg. Understanding Ethereum's gas and transaction fees, Available at: https://cryptobriefing.com/understanding-Ethereums-gas-transaction-fees/, accessed July 2020.

[40] Coin Metrics. The Ethereum gas report, Available at: https://coinmetrics.io/the-ethereum-gas-report/, accessed April 2021.