Kim *et al. Hum. Cent. Comput. Inf. Sci.* (2020) 10:36
https://doi.org/10.1186/s13673-020-00244-8

Human-centric Computing and Information Sciences

RESEARCH Open Access

Automatic, location-privacy preserving dashcam video sharing using blockchain and deep learning

Taehyoung Kim[1], Im Y. Jung[1*] and Yih-Chun Hu[2]

*Correspondence:
iyjung@ee.knu.ac.kr
[1] School of Electronics Engineering, Kyungpook National University, Daehakro 80, Bukgu, 41566 Daegu, South Korea
Full list of author information is available at the end of the article

Abstract

Today, many people use dashcams, and videos recorded on dashcams are often used as evidence of accident fault. People can upload videos of dashcam recordings with specific accident clips and share the videos with others who request them, by providing the time or location of an accident. However, dashcam videos are erased when the dashcam memory is full, so periodic backup is necessary for video sharing. It is inconvenient for dashcam owners to search for and transmit a requested video clip from backup videos. In addition, anonymity is not ensured, which may reduce location privacy by exposing the video owner's location. To solve this problem, we propose a video sharing scheme with accident detection using deep learning coupled with automatic transfer to the cloud; we also propose ensuring data and operational integrity along with location privacy by using blockchain smart contracts. Furthermore, our proposed system uses proxy re-encryption to enhance the confidentiality of a shared video. Our experiments show that our proposed automatic video sharing system is cost-effective enough to be acceptable for deployment.

Keywords: Automation, Automotive, Data sharing, Security, Privacy, Blockchain, Deep learning

Introduction

A dashcam is a device that records events around a vehicle [1]. According to Embrain Trend Monitor research that surveys the satisfaction of dashcam users [2], the number of people who feel the need for a dashcam in their car has been steadily increasing. Because a dashcam records the event history of a vehicle, its videos can be used as evidence in disputes arising from accidents, both involving and observed by the dashcam owner. If a dashcam acquires video of accidents involving nearby vehicles, that video can be used as a record of the accident. In South Korea, public institutions often make announcements to look for witnesses of accidents and wait for a report. People share videos for various reasons [1], such as publicizing the truth, solving unfair cases, helping others, and benefitting the public.

However, Kim et al. pointed out that waiting for a report and uploading a shared video via the web can reveal personal information such as location information or a fabricated

video can be uploaded [3]. To solve these problems, they proposed a video sharing method that transmits the video's profile and a guard profile to protect location information. Their method protects location information by uploading additional information. Another limitation of manual uploads is that the video owner must directly upload the video themselves.

Hossain proposed a video sharing method for a smart city environment [4]. In this environment, the videos from a user's smartphone or from other cameras in the city are transmitted to the cloud for various purposes. Watermarking and signatures are to be used to authenticate videos and prevent them from being manipulated or shared by unauthorized people. However, this approach doesn't consider user privacy and is thus unsuitable for sharing videos from a dashcam.

In this paper, we propose a dashcam video sharing scheme that guarantees the integrity and confidentiality of videos to be shared. In addition, it improves the location privacy of the video provider by utilizing the anonymity of a smart contract [5] in a blockchain as shown in Fig. 1. The Keccak 256 hash function [6] is used to guarantee video integrity and AES [7] is used to ensure confidentiality. Through deep learning, only when accidents are detected are the applicable videos selected, encrypted, and transmitted to the cloud automatically; this process occurs before the requests for the videos. Smart contracts then facilitate video requests and sharing. The search and retrieval process is pseudonymous; only the user's account address appears in the transaction. We use proxy re-encryption to share the encryption key [8]. In order to prevent the video from being manipulated, the dashcam transmits only a hash of the video. The time and location information is periodically not transmitted. When the accident occurs, the time and location information is transmitted. Thus, the user's location is not revealed.

The contribution of this paper is as follows:

- Our scheme is efficient because it uses deep learning to detect and share only accident video, and we store hash values rather than actual video in the blockchain.

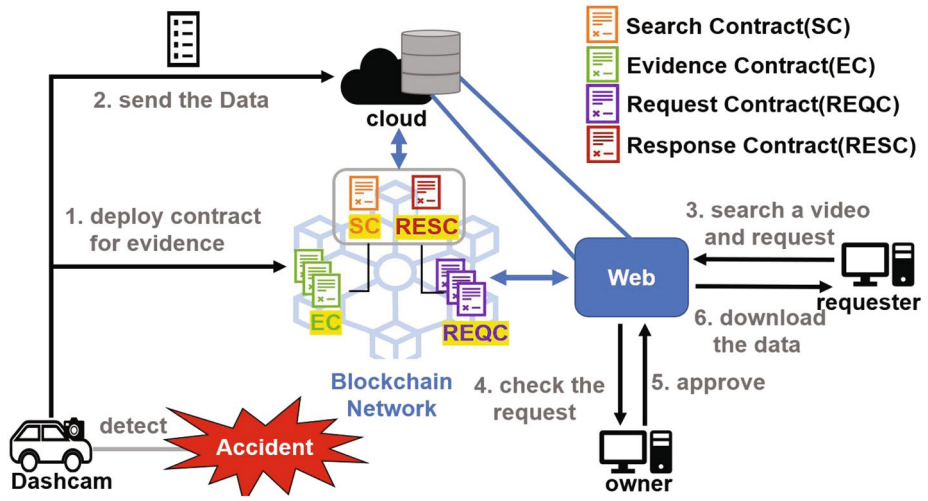

Fig. 1 Video sharing scheme [9]

- A user's location privacy is improved because the information that can reveal it is not continuously uploaded and the request and sharing process are performed anonymously.
- We combine a smart contract with proxy re-encryption to ensure video confidentiality and securely share videos without exposing plaintext to unauthorized people.
- Because the amount of data stored in a smart contract is a small hash value and video retrieval is performed from the cloud, we minimize the cost of the smart contract.

The rest of this paper is organized as follows. "Preliminaries" section describes the basic technologies to understand the proposed scheme: blockchain, deep learning, and proxy re-encryption technique. In "Related works" section, we discuss related work: video sharing schemes, data sharing using blockchain, and event detection using deep learning. In "Automatic accident detection" section, we explain the use of deep learning to detect accident video. In "Automatic video sharing" section, we describe the proposed video sharing scheme using blockchain and deep learning. In "Experiments" section, we discuss an image detection experiment. In "Evaluation" section, the accuracy, overhead, and security for the experiments and implementations are evaluated. Finally, our conclusions and future work plans are presented in "Conclusion" section.

Preliminaries

In this section, we discuss the blockchain and proxy re-encryption for requesting and sharing the video. In addition, we discuss the convolutional neural network (CNN) used for accident detection.

Blockchain

A blockchain is a type of database that stores data in block units and links blocks to each other in chronological order [5, 10–13]. Each block depends on the previous block and all participants share a ledger. Blockchains can provide pseudonymity because transactions can be made only through address without personal information. A blockchain design can be divided into a public blockchain and a private blockchain. In a public blockchain, anyone can join the network, read the data in the blockchain, and deploy new transactions. Anyone can check the data and benefit from the system's integrity and transparency [14, 15]. In a private blockchain, only authorized nodes can participate in the network and receive transactions.

Ethereum is referred to as a second-generation blockchain because it can implement more complex and decentralized applications beyond its cryptocurrency application. Unlike previous blockchains such as Bitcoin [16], Ethereum allows for smart contracts. A smart contract is a program that runs automatically according to the rules proposed by Szebo [17]. Buterin implemented the smart contract on Ethereum consisting of the program code, storage file, and account balance [15, 18, 19]. Anyone can create a contract by posting a transaction to a blockchain. Because the program code is written to the blockchain, the code cannot be changed once the contract has been created. The program code defines the rules, and these rules are executed exactly as defined and cannot be changed. A transaction is traceable because it is recorded in the blockchain. When a contract is created, it allocates storage space in

the blockchain and executes the program code whenever a message is received from the user or another contract. A smart contract can take coins from the outside and put them in the smart contract's balance or transfer its coins to the outside. The use of a smart contract in a blockchain allows the execution of a trusted and promised rule without third-party intervention.

In this paper, we use Ethereum to implement a public blockchain in which all the nodes participating in the network can share videos. We created a smart contract using the Solidity language [20], which is the most popular language used in Ethereum. We implemented a web interface that enables users to share videos with web3, a JavaScript API for Ethereum.

Proxy re-encryption (PRE)

Proxy re-encryption (PRE) is a technology that transfers encrypted data between nodes by using a proxy without sharing a symmetric key [8, 21, 22]. The proxy manages the encrypted data, receives the re-encryption key (*rekey*), and transfers it to another node. For example, if Alice (data owner) wants to send encrypted data by using her public key to Bob, she creates a rekey with her private key and his public key. She shares the *rekey* and the encrypted data with him through the proxy. He can then re-encrypt the data and decrypt it using his private key. Therefore, it is possible to transfer data securely without exposing the plaintext during data transmission. In this paper, we adopted and applied a PRE technique called pyUmbral of Nuchper [22]. We designed a protocol that allows the rekey and the public key to be sent according to pre-defined rules by applying the PRE to a smart contract.

Convolutional neural network (CNN)

A CNN is a deep learning network form that is often used for image classification. The CNN has better image recognition performance than traditional machine learning approaches [23, 24]. As a video is a stream of images, many studies have used CNNs to classify videos by dividing the video into frames, recognizing each frame using the CNN, and making predictions by aggregating each result [25]. A CNN consists of an input layer, a convolution layer, a pooling layer, and a fully connected layer [26]. The input layer contains the three-dimensional image data (width, height, and depth) and extracts features for an image in the convolution and pooling layers. The fully connected layer acts as a classifier. A considerable amount of research has been conducted using CNNs [23–28]. In particular, researchers have used transfer learning, a learning method that uses a pre-trained model to retrain the data for the network's purpose [26, 27]. Transfer learning is a technique that improves the performance of a CNN even when there are few datasets. In this paper, we used the VGG16 model [29] which has the lowest power consumption at 10 W and high accuracy; thus it is suitable for execution on a battery-operated dashcam. The VGG16 model has the disadvantage of a relatively long inference time of 200 ms [30]. In this paper, we analyzed one image per second and demonstrated that the VGG16's classification is fully available within 1 second; we present these results in Section "Experiments".

Related works

In this section, we discuss research related to this paper in video sharing, privacy guarantee through blockchain, and image classification using deep learning.

Video sharing

Kim et al. proposed ViewMap for sharing dashcam videos [3]. Hossain proposed a video sharing method in a smart city environment [4]. ViewMap doesn't distinguish between an accident and normal videos. For each minute of video, ViewMap creates a View Profile (VP), which is a compact video format, and uploads it to a server. Because the VP includes time and location information, continuous leakage of this information could violate a user's location privacy [31, 32]. In order to improve location privacy, a Guard VP is created and uploaded. The Guard VP is created by exchanging the last location information of the video with surrounding vehicle's location information. Although the Guard VP can ensure location privacy, it adds significant overhead because it adds data to the VP stream. Furthermore, the user must manually find the requested video clip and upload it on their own. Watermarking and authentication techniques with signatures were considered [4] to prevent videos from being manipulated or shared by unauthorized people. A video owner uploads his/her videos to the cloud for later use. The uploaded videos can be shared by city stakeholders such as insurance companies, law-enforcement officials, and other smart city governance entities. However, the privacy and confidentiality of the videos to be shared are not considered. Also, videos can be shared by city stakeholders but an individual cannot retrieve and acquire a specific video.

Privacy preservation using a blockchain

Many studies have aimed to share data or guarantee privacy using blockchain [14, 21, 33–36]. In general, the blockchain is used only as an index of data, with the actual data stored in separate nodes [21, 33]. For example, references and symmetric keys of medical records can be encrypted and stored in a blockchain [21]. Using proxy re-encryption [8], a symmetric key is encrypted and transferred safely to another node. A smart contract can authorize a requester who can request data; the re-encryption key to obtain the symmetric key is then shared with the requester and the data provider sends the actual data to the requester via HTTPS. In [33], when a patient's medical data is generated, the data is encrypted and transmitted to the cloud. For scalability, an index is created that allows medical data to be retrieved and shared through a blockchain. A user is authorized by the data owner and requests the data by searching through the smart contract. However, in these studies, there are a limited number of nodes that can request data besides the patient. In addition, privacy is ensured by gaining patient approval and allowing only authorized users to access the private network and share data. Thus, this system cannot be used in a scenario where anyone can request and receive data.

Knirsch et al. proposed a protocol to find the optimal charging station near a vehicle through a blockchain [34]. A request is made from the vehicle to the blockchain to find a matching charging station. Each charging station presents its bid and an auction proceeds. The vehicle checks the bids provided by the charging stations, selects one of them,

and moves to that station. Upon arrival at the charging station, the certification process is performed through one-to-one communication between the vehicle and the charging station. Personal information other than ID is not included when making the request. Because this ID is changed with each request, it also ensures user privacy. Jia et al. and Yang et al. proposed methods for ensuring the location privacy of users in a crowdsensing where data was collected, shared by various people, and analyzed [14, 35]. In [14], the authors used a combination of public and private blockchain to ensure privacy. An agent acting as a miner constitutes a public blockchain. Each agent constitutes its own private blockchain. When a requester registers task in the public blockchain, the agent posts the task to its private network. In the private network, a worker can select the task he/she wants. When the task is finished, the worker uploads sensing data to the blockchain. The agent verifies that the sensing data meets the requester's requirements. When the data is met, the worker can be compensated. These studies share data while guaranteeing privacy but do not mention large data volumes. They also propose a method of transferring data to a specifically designated server rather than to an individual sharing their own data. In [35], the server assigns tasks to people. As people move, the sensing data from attached device is encoded and stored in the blockchain. The server can read the data stored in the blockchain. The data contains the user's location, but it does not contain other personal information. The data is pseudonymously managed through the user's ID. It ensures privacy by protecting the data through an encoding process.

In this paper, we propose a scheme for large data volumes, such as videos, while ensuring location privacy. Protocols for search, request, and sharing the videos can be executed with smart contracts for secure operation.

Image classification to detect events

Karpathy et al. classifies videos using multiple frames as inputs to a CNN [23]. Because most of the research on deep learning aims to improve accuracy, deep learning often runs on high-performing servers. However, using multiple images at the same time as a system input requires high processing power; processing multiple images simultaneously on a resource-limited device has low performance. There are cases where it takes a long time to perform deep learning on an embedded device.

Several studies detect a driver's distracted behavior by using deep learning [24, 26, 28]. The distracted behavior is detected on videos from a camera facing the driver. Leonardo et al. conducted a study to detect whether an accident occurred between a vehicle with an installed dashcam and another vehicle [28]. He used the dashcam image as an input and YOLOv3 [36] to recognize the object in the image. [30] compares resource consumption and execution time for various image classification models on the NVIDIA TX1 [37]. Because each model has different power and memory demands, appropriate models should be selected and used to meet their required performance given the specifications of their host deep learning devices. In our proposed scheme, frames are extracted and applied to the CNN at specific times considering the specifications of the dashcam. The video soundtrack is additionally used to improve accuracy.

Automatic accident detection

The dashcam we studied creates a 50 MB video every minute [3]. Only a few of these videos are interesting to users. Thus, uploading all videos to the cloud is an inefficient use of communication and cloud storage. In this paper, we propose a scheme to share accident video chosen selectively by deep learning. We determine the occurrence of an accident from the images and sound of the video recorded on the dashcam by supervised learning [38]. The learning model is created in the cloud and stored in the dashcam firmware. As shown in Fig.2, when a 1 min. video is created, a total of 60 frames are extracted at a rate of one per second for image detection. Sound is captured for each 1-second interval corresponding to the extracted frames to help determine whether an accident occurred. If the machine learning algorithm indicates an accident, an Evidence Contract (EC) is deployed, and the data related to the accident video is transmitted to the cloud. The cloud manages the accident images separately and performs retraining to increase accuracy for the next incident.

Image classification

In this paper, we used a transfer learning method to reuse the VGG16 image classification model [29] which was trained on Imagenet [39]. Imagenet has 1.2 million images divided into 1000 groups. In order to recognize accidents using VGG16, we used a classifier consisting of a fully connected layer and a softmax layer.

The videos to be shared in this paper are those that captured an accident. The dataset used for learning consisted of images from YouTube videos and Github's images [40, 41]. The images were divided into two classes: normal driving images and accident images. The images were compared to the learned data at the dashcam. According to [30], when general deep learning techniques are used in an embedded device,

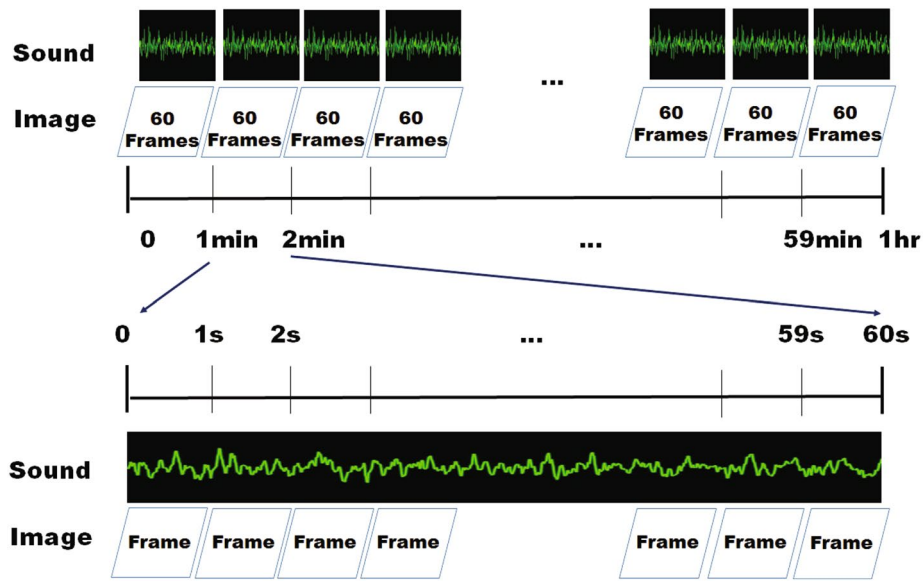

Fig. 2 Image and sound for deep learning

the execution time and power consumption limit performance. Considering this, one frame was analyzed every second.

Sound classification

The one frame per second analysis method is subject to inaccuracies when an accident occurs between sample frames [27]. We also use the video's soundtrack to detect accidents. The sound of an accident is distinct from the sound of normal driving. Before a crash, the tires skid and produce corresponding noise. During the crash, the vehicle structure crumples and/or the windows break. In dashcam videos, the sound inside the vehicle is louder than outside sounds, so there is a limit to our ability to distinguish between the two sound environments. Therefore, we assumed that a microphone was installed outside the vehicle and connected to a dashcam to collect outside sounds.

The process for detecting sounds consists of the soundtrack input, feature extraction from the sound data, and classification. The 1 s recordings were used as the input and features were extracted from the data using the Mel Frequency Cepstral Coefficient (MFCC) [42]. This value was put into a classifier consisting of a fully connected CNN layer to determine whether the sound was an accident sound or a normal sound.

Automatic video sharing

In this section, we propose a protocol for video sharing through a blockchain. Figure 3 shows the entire protocol.

Smart contract

There are four smart contracts in the proposed scheme as shown in Fig. 4: an Evidence Contract (EC) for managing the evidence in a video; a Search Contract (SC) for

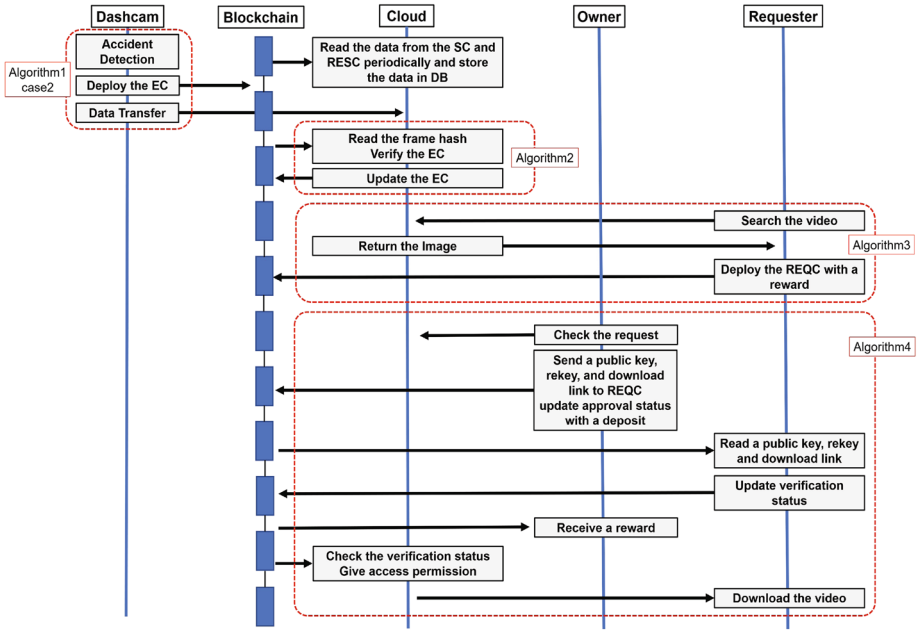

Fig. 3 Video sharing protocol

Table 1 Notations

Symbol	Description
ID	Account address of user
FN_i	File name of ith video
N_i	ith video name
H_i	ith video hash value
H_{frame}	Frame hash value
$Addr_i$	Address of i
SK_i, PK_i	Secret (Private) key and Public key of i
rekey	Re-encryption key
Key_{sess}	Session key for video encryption
E_{key}	Encryption with key

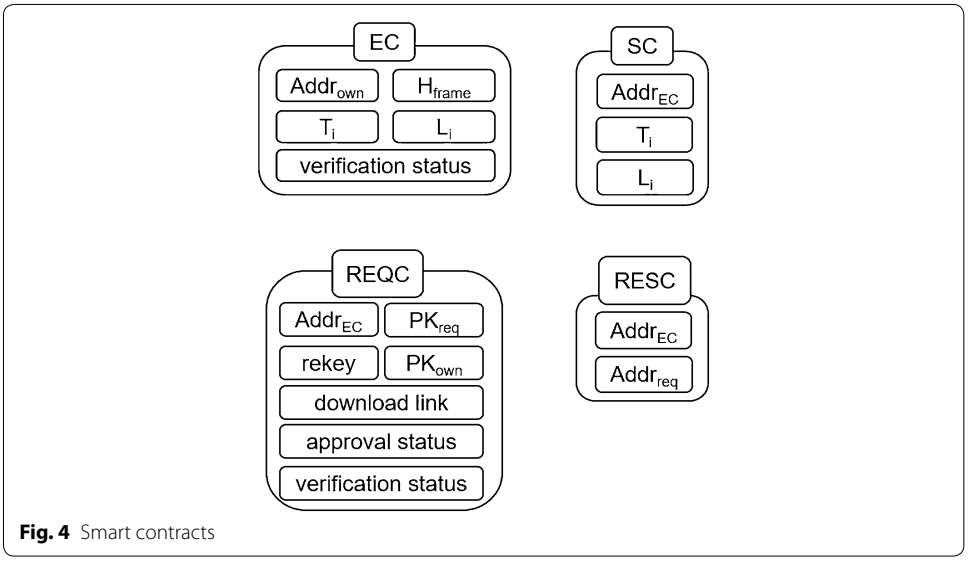

Fig. 4 Smart contracts

managing the validated EC and storing data for the purpose of video retrieval; a Request Contract (REQC) for requesting videos; and finally a Response Contract (RESC) for associating the addresses of the owner and REQC. SC and RESC are deployed by the cloud and only one copy of each exists. The dashcam deploys an EC whenever an accident video is detected. REQC is deployed by the requester for each video request. The notation used in this paper is defined in Table 1.

Dashcam sends data to the cloud

Algorithm 1 transfers data from the dashcam to the cloud. The dashcam creates videos in segments of 1 min. It then creates a hash for each video, H_{video}, and sends it to the cloud for video verification whenever the dashcam is connected to a network (i.e., Wi-Fi or Cellular). The hash function is the Keccak 256 function used in Ethereum. This process is shown in Algorithm 1's case 1. When an accident is detected through deep learning, a session key, Key_{sess}, is generated. Key_{sess} is then encrypted with the owner's public key (PK_{own}). Then, the dashcam deploys an EC

by adding the information such as frame hash H_{frame}, time T_i, and location L_i. The dashcam transmits the encrypted data, image, and EC address to the cloud. The image is provided to help a requester to choose a video to download. We selected the front frame of the video. This process is shown in Algorithm 1's case 2.

Algorithm 1 Dashcam sends data to Cloud

Case 1. No Accident
1: video hash create
$H_{video}=N_i+ID+FN_i+H_{i-1}+H_i$
$H_i=H_{frame}+FN_i+H_{i-1}$
2: send H_{video} to Cloud

Case 2. Accident
1: key generation
Key_{sess} : session key generation
2: data encryption
$En_1=E\_Key_{sess}(Video)$
$En_2=E\_PK_{own}(Key_{sess})$
3: deploy the Evidence Contract (EC)
adding H_{frame}, T_i and L_i
4: Transfer En_1, En_2, *image* and $Addr_{EC}$ to Cloud

Deploy and verify the EC and update the SC

Algorithm 2 shows the process of verifying the EC deployed from the dashcam and updating the SC. When the dashcam sends accident data to the cloud, the cloud verifies the EC. First, the frame hash value is read from the EC and the hash value of the video is calculated. The EC is verified by comparing the calculated hash value with the previously stored hash value. The EC's verification status is updated when verification is completed. Only the cloud can update the verification status. When the EC is updated, the address, time, and location information of the EC are automatically transferred to the SC to update the SC.

Algorithm 2 deploy and verify the EC and update the SC

Dashcam : input *ID*, H_{frame}, T_i, and L_i
1 : deploy the EC

Cloud :
1 : read H_{frame} from the EC
2 : calculate H_i
3 : verify stored H_i and calculated H_i
4 : update the EC's verification status
5 : periodically reads SC's data and store it in DB

EC :
1 : **if** *msg.sender* is Cloud
then change the verification state to *verified*
2 : send the verified $Addr_{EC}$, T_i, and L_i to SC

SC :
1: mapping $Addr_{EC}$, T_i, and L_i

Search and request a video

Algorithm 3 shows how the requester retrieves and requests a video. The requester accesses the cloud to retrieve the desired video based on time and location. The cloud shows the accident image and address of the EC that meets the condition. The requester can easily identify the desired video through the image and requests the video by deploying the REQC, adding the EC's address and his/her public key with a reward. When the REQC is deployed, it automatically sends the REQC address and EC address to RESC.

Algorithm 3 search and request a video

Requester :
1 : search the video using *T, L,* and *image*
2 : Cloud returns the corresponding $Addr_{EC}$
3 : deploy the REQC with a reward, $Addr_{EC}$, PK_{req}

REQC :
1 : load EC object (EC_{obj}) from the $Addr_{EC}$
2 : send $Addr_{REQC}$ and $Addr_{EC}$ to the RESC

RESC :
1 : load EC_{obj} from the $Addr_{EC}$
2 : find the $Addr_{own}$ from EC_{obj}
3 : mapping $Addr_{own}$ and $Addr_{REQC}$

Cloud :
1: periodically reads RESC's data and stores it in DB

Share the video with the requester

Algorithm 4 shows the process where the owner of the video checks and shares the video. The owner accesses the cloud and checks for requests from his/her account address. The RESC maps the address of the REQC to the address of the owner. The cloud reads the RESC's data periodically, stores it in the DB, and displays this information at the request of the owner. The owner obtains the addresses of the REQC and the EC. Through the EC, it is possible to confirm the request for a video and to decide whether to approve it. To approve, the owner accesses the REQC to obtain the PK_{req} and generates a *rekey* through SK_{own} and PK_{req}. Then the owner sends the *rekey*, PK_{own} and *download link* to the REQC with a deposit. The requester can read the *rekey*, PK_{own} and *download link* from the REQC deployed by himself/herself. The requester can acquire Key_{sess} using *rekey* and PK_{own}. The requester then updates REQC's verification status. If verification status is True, the owner gets his/her deposit and reward. Cloud checks this status and gives access permission about the download link to the requester. The requester downloads encrypted video from the cloud and decrypts the video with the Key_{sess}.

Algorithm 4 share the video to requester

Owner :

1 : search a request using $Addr_{own}$
2 : Cloud returns the corresponding $Addr_{REQC}$
3 : access the REQC and check the $Addr_{EC}$
4 : check the video to be shared
5 : if approved, read PK_{req} from the REQC
6 : create a *rekey*
7 : update REQC's approval status
8 : send the *rekey*, PK_{own}, and *download link* to REQC with a deposit

Requester :

1 : read the data such as the *rekey,* PK_{own}, and *download link* from REQC
2 : re-encrypt the encrypted Key_{sess} using *rekey* and PK_{own}
3 : acquire Key_{sess} using SK_{req}
4 : update REQC's verification status
5 : download the encrypted video from Cloud
6 : decrypt the video using Key_{sess}

Experiments

In this section, we report on the experiments to detect an accident through deep learning. We used images to detect the accident and added sounds to improve accuracy. In addition, to find an optimal time interval, we measured the probability of detecting an accident when applying the image plus sound method to actual videos at different time intervals. The next section describes the experimental environment configuration, method, and results for deep learning.

Experimental environments

As shown in Fig. 5, we used a desktop computer to represent the cloud and a Raspberry Pi3 B+ to represent the dashcam's CPU. QXD3000, INAVI's latest dashcam , uses the cortex-a53 CPU. Therefore, the experiment was conducted using the Raspberry board since it has specifications like those of a dashcam. We create the model for deep learning

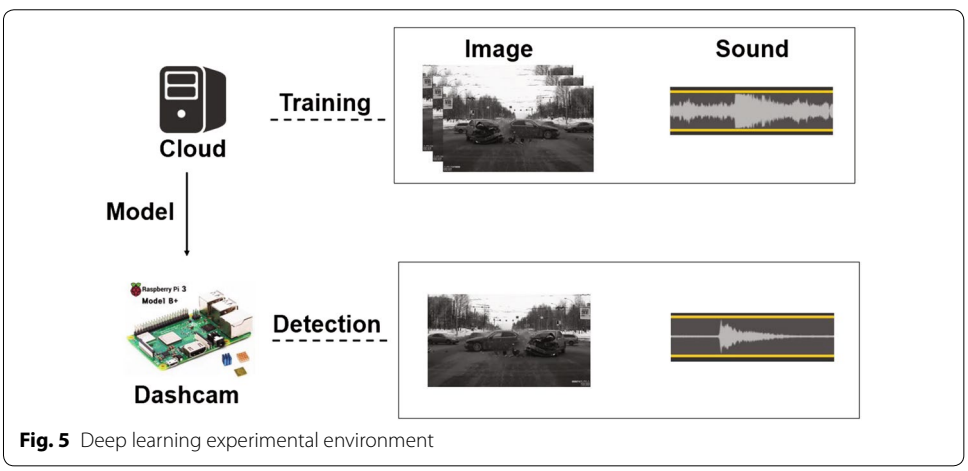

Fig. 5 Deep learning experimental environment

Table 2 Specification

	Role	CPU	Memory
Desktop	Cloud	Intel i7-4790 3.6 GHz	16 GB
RasberryPI 3B+	Dashcam	Cortex-a53	1 GB

Table 3 Test Dataset

	Accident	No accident	Refs.
Image	264	255	[40, 41]
Sound	138	955	[40, 45, 46]

Table 4 Image Test Results

	Results predicted	
	Accident	No accident
Actual results		
Accident (P = 50)	38 (TP)	12 (FN)
No Accident (N = 50)	2 (FP)	48 (TN)

by learning the accident data from the desktop, and the Raspberry board used this model to determine whether there was an accident. We assumed that the learning model generated from the desktop was loaded into the dashcam firmware. The specifications of the equipment are given in Table 2.

The libraries used in the experiment were Keras 2.2.4 [43] and Tensorflow 1.13.1 [44]. The pre-trained model was VGG16 [29]. The dataset contained 519 images and 1093 sounds. Each configuration is shown in Table 3. We randomly divided the data and used 80% in the training and 20% in the test.

Detection results

In all, 214 accident images and 205 no accident images were used for learning. Further, the 100 images (50 accident and 50 no accident) that were not used for learning were used for testing, the results of which are shown in Table 4. A true positive (TP) is an accident detected for an accident image. Likewise, true negative (TN) is a non-accident decision for a non-accident image. The accuracy can be defined as the ratio of TP and TN to P and N. False positive (FP) and false negative (FN) are cases in which a no accident image is determined to be an accident, and an accident image is determined as a no accident image, respectively.

$$Accuracy = \frac{TP + TN}{P + N} = \frac{38 + 48}{50 + 50} = 86\% \tag{1}$$

Similarly, 108 accident sounds and 780 non-accident sounds were used for learning. We used 30 accident sounds and 175 non-accident sounds for testing purpose. The results are shown in Table 5.

Table 5 Sound test results

	Results predicted	
	Accident	**No accident**
Actual results		
Accident (P = 30)	24 (TP)	6 (FN)
No accident (N = 175)	3 (FP)	172 (TN)

Table 6 F-meausre

	F-meaure	Precision	Recall
Image	0.84	0.95	0.76
Sound	0.84	0.89	0.80

$$Accuracy = \frac{TP + TN}{P + N} = \frac{24 + 172}{30 + 175} = 95.6\% \tag{2}$$

Because F-measure is the variant most often used when learning from imbalanced data [47], we calculated F-measure from TP, FP and FN using the equations, 3, 4 and 5, [48]. The results are shown in Table 6.

$$F - Measure = \frac{2 * Precision * Recall}{Precision + Recall} \tag{3}$$

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives} \tag{4}$$

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives} \tag{5}$$

Many dashcams typically record the sound inside the vehicle; the external sound is often overwhelmed by the internal sound of the vehicle. However, assuming that external sound is captured and used for accident detection, the sound related to the accident can be very unusual in loud noise and collision sound, which can improve accident detection accuracy. In the case of videos that are determined to have no accidents by image, they are finally classified as accidents by sound. This method can improve the accuracy of the decision for the false-negative videos in Table 4. When accident video soundtrack was combined with silent videos that show false-negative accident detection from YouTube, all were classified as accident videos.

Evaluation

In this section, we evaluate the proposed scheme in terms of accuracy, overhead, and security. In order to detect accidents, we proposed a combination approach of image and sound classification but assumed that sound was recorded through a microphone installed outside the vehicle. Therefore, deep learning through sound and image data was analyzed separately because it is difficult to obtain the videos with the external sound of the vehicle.

Accuracy analysis

In the experimental results presented in "Overhead analysis" section, the accuracy of image detection was 86% and the accuracy of sound detection was 95.6% as shown in Eqs. 1 and 2 if we only consider the highest accuracy at our dataset. And, Table 6 shows that the F-measures of the image and sound are equal to 0.84, and are close to 1.

On the other hand, thresholds affect accident detection by deep learning. The optimal threshold can improve detection accuracy by reducing FP and FN. We sought the optimal threshold for accident decision in deep learning considering cost factor. *FN* and *FP* result in extra cost of our scheme. That is, in an environment in which many vehicles with their dashcams observe accidents, as *FN* increases, the opportunity cost of not having the accident videos increases. In contrast, as *FP* increases, the cost to upload no accident videos increases. Therefore, we studied the relation between *FN* and *FP*, and the threshold values as the number of vehicles observing each accident changes. The costs of *FN* and *FP*, $FN(Cost)$ and $FP(Cost)$, were defined as follows.

$$\begin{aligned} FP(Cost) &= (C_{Cloud} + C_{BC} + C_{Net}) * FP * R_{NoAcc} \\ FN(Cost) &= C_{Opp} * FN * R_{Acc} \end{aligned}$$

At the cost evaluation of *FP* and *FN*, we estimated the coefficients at Table 7. C_{Cloud}, C_{BC}, C_{Net}, and C_{Opp} are the cloud cost, the blockchain storage cost, the network usage cost per video each. R_{Acc} and R_{NoAcc} present accident frequency and no accident frequency each.

Figure 6 shows the total costs of $FN(Cost)$ and $FP(Cost)$ as the number of the vehicles observing each accident changes. We choose 1, 2, 4, and 8 as the average number of vehicle witnesses by considering the widths of road lanes and the vehicles located proximate to the accident spots. There are many types of roads; one-lane roads, two-lane roads, four-lane roads, eight-lane roads, and so on. One-lane roads are for one-way traffic, and multi-lane roads are for two-way traffic. In general, dashcam is equipped in the dashboard and records the situation in front of the vehicle. It is difficult to detect long-distance accidents or lateral accidents accurately. On congested roads, many vehicles block the scene of the accident. Thus, we assume the average number of vehicle witnesses is one for one-lane roads, two for two-lane roads, four for four-lane roads, and eight for eight-lane roads. For multiple lanes of more than eight lanes, many vehicles far from the accident spots will witness long-distance accidents or parts of the accident scenes shaded by other vehicles. We sought an optimum threshold with minimum total cost of *FP* and *FN*. $FP(Cost)$ includes the overheads video providers should endure as well as the management cost of our scheme. $FN(Cost)$ shows how much we lost without the clues of

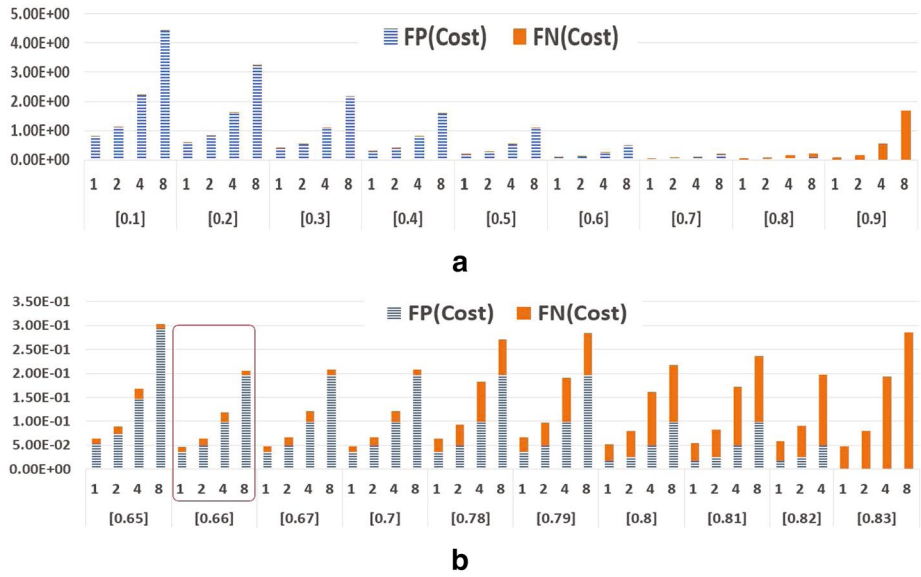

Fig. 6 An optimal threshold by (FN(Cost) + FP(Cost)) as the number of vehicles changes at each threshold (a)# of vehicles observing an accident (1, 2, 4, 8) at threshold [0.1, 0.9] (b)# of vehicles observing an accident (1, 2, 4, 8) at threshold [0.65, 0.83]

Table 7 Cost estimation parameters

Parameter	Estimates	Ref. cost
C_{Cloud}	$0.05GB * \$0.023/GB$	Amazon S3[a]
C_{BC}	\$0.04	Ethereum[b]
C_{Net}	$0.924 * 0.05GB * \$12.37/GB$	92.4%[c] Mobile Data Communication[d] over Celluar network
C_{Opp}	$\frac{14}{270.4} * \$25,000 * \frac{1}{2}^{TP}$	hit-and-run reward[e], 270.4 million vehicles[f], 14 million vehicles involved in accidents[g]
$\frac{R_{NoAcc}}{R_{Acc}}$	$\frac{25,270}{n}$	13,476 miles per year, average speed 32 mph, average of n witnesses per accident[h]

[a] https://aws.amazon.com

[b] https://ethgasstation.info

[c] The wireless data traffic in the United States in 2018 (in million GB) : 45,250, https://www.statista.com/statistics/615419/wireless-data-traffic-in-the-us/, Free Wi-Fi data traffic volume in the United States in 2018 (in million GB) : 3,437, https://www.statista.com/statistics/994889/free-wi-fi-traffic-volumes-in-the-us/

[d] https://howmuch.net/articles/the-price-of-mobile-internet-worldwide-2019, The average price of mobile internet 2019 in US

[e] https://abc7.com/$25k-offered-in-hit-and-run-of-13-year-old/5521868/

[f] https://hedgescompany.com/automotive-market-research-statistics/auto-mailing-lists-and-marketing/

[g] https://www.statista.com/statistics/192097/number-of-vehicles-involved-in-traffic-crashes-in-the-us/

[h] https://www.autogravity.com/autogravitas/money/whats-average-miles-driven-per-year-car-lease-guide, http://ridetowork.org/transportation-fact-sheet

accident shots. We want both of them to be minimum. If we seek an optimal threshold as the number of vehicles observing accidents changes, we get the threshold of 0.66 as shown in Fig. 6(b). Figure 7 shows the optimal thresholds, 0.53 or 0.54 or 0.55, so that simply FN/FP is 1. The optimal threshold difference between Figs. 6 and 7 is caused by

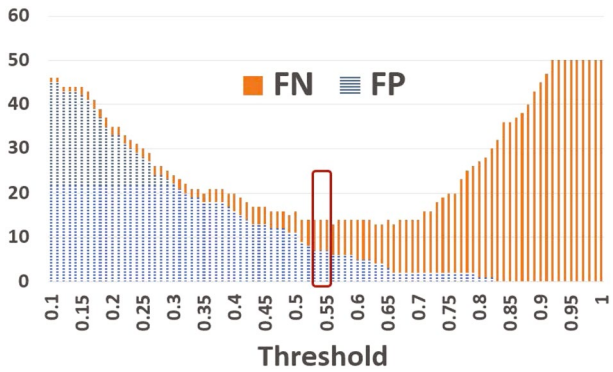

Fig. 7 An optimal threshold by FN/FP

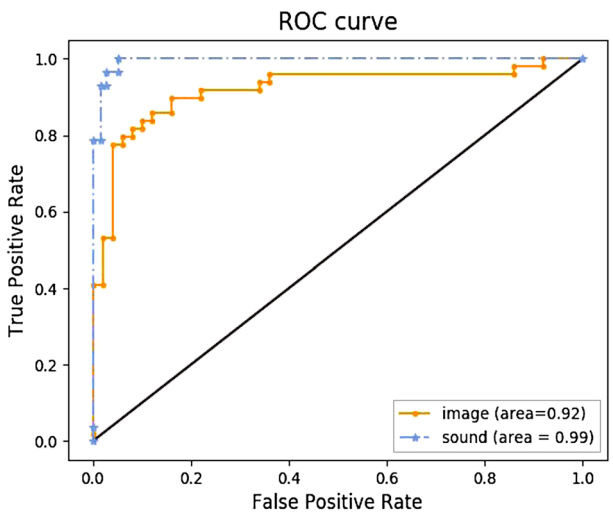

Fig. 8 ROC curve

the characteristics of our dataset and the cost functions, *FN*(*Cost*) and *FP*(*Cost*). At low thresholds, *TP* and *FP* dominates *FN*(*Cost*) and *FP*(*Cost*) each, showing low *FN*(*Cost*) and high *FP*(*Cost*). At high threshold, *FN* and *TN* dominate *FN*(*Cost*) and *FP*(*Cost*) each, showing high *FN*(*Cost*) and low *FP*(*Cost*). Our dataset shows that FP drops down rapidly and converges to 0 at threshold 0.83. That is, after the threshold 0.83, the total cost is the same with *FN*(*Cost*). FN grows slowly at low thresholds. High *TP* at low thresholds lets *FN*(*Cost*) stay at small value compared with *FP*(*Cost*).

Figure 8 shows a Receiver Operating Characteristic (ROC) curve [49] for the image and sound test. The larger the area under the curve in the ROC curve, the better the performance. Considering the characteristics of the image data used in learning, we found that most of the data clearly showed an accident such as collision with another vehicle or obstacle, causing the vehicle to be crushed or the windows to break. For some accidents, such as a light collision between vehicles rather than a serious accident, the results were not well detected. Moreover, among the images that did not depict an accident, the results were not predicted accurately when the vehicle was in very crowded places. The

Table 8 Accuracy comparison with other studies

Our scheme		[27] (%)	[28] (%)
Image (%)	Sound (%)		
86	95.6	87.2	87.03

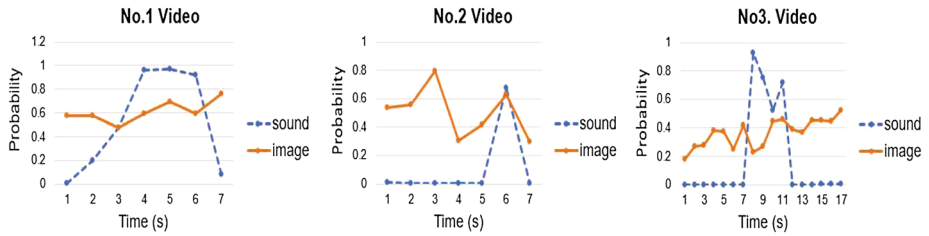

Fig. 9 Comparison graph of image and sound detection

sound characteristics of accident scenes vs. non-accident scenes provided more differentiation than images alone, so we obtained better results with the soundtracks.

Table 8 compares the accuracy of the proposed method with that of methods developed in other studies. The method proposed in [27] predicted the user's distracted behavior by means of a camera installed inside the vehicle. Imagery and data such as the acceleration received from the user's mobile device were combined, resulting in an accuracy of 87.02%. The method proposed in [28] combined the data from a dashcam image and the acceleration data received from a telematics device, and the classification accuracy 87.03%. Compared with the results of [27] and [28], the results obtained in the present study revealed that the accuracy using images alone was similar and the accuracy when adding sounds was higher.

Figure 9 shows the probability of an accident when applying both image and sound analysis to actual videos that captured another vehicle's accident. These videos have no extraneous noise such as human voices or music and the sound of the accident is clearly heard. The use of sound distinguishes the accident more clearly than that of using the image. Therefore, if sound and image are used together, false negatives can be reduced to achieve higher accuracy. The sound data obtained from the dashcam in the test videos contained a small external sound and a large internal sound. The sound inside the vehicle could be filtered by installing a microphone outside the vehicle. As discussed in [27] and [28], because the amount of data used in learning was small, it will be possible to further improve accuracy by collecting and learning data continuously.

Overhead analysis

Table 9 shows the time taken to detect the accident image and the amount of data used to transmit the 1-min video evidence. When the image was extracted at the rate of 1 frame per second from the 1-min video, the average detection time was 0.62s. In [30], the researchers compared the time taken and resource consumption when using several pre-trained models in an embedded device. When a TX1 board [37] was used, the image inference time was 200 ms for the VGG16 model. In this paper, we experimented with

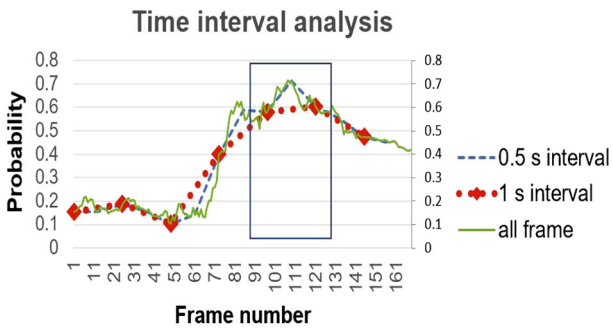

Fig. 10 Accident probability graph with different time interval

Table 9 Overhead comparison with other stuies

	Our scheme	ViewMap [3]	VGG16 [30]
Detection Time	0.62s	Not measured	200 ms
Transferred data size	96B	4,584 B	Not measured

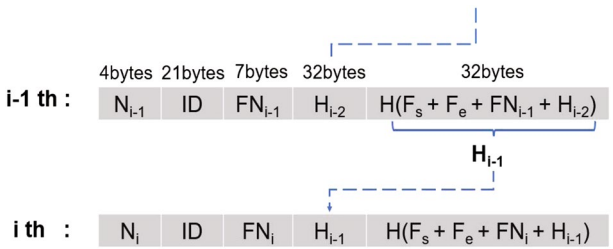

Fig. 11 Evidence of the video [9]

the Raspberry Pi. Because its performance was not as fast as the TX1, the inference time was three times longer, but it was still reasonable for processing an image every second. Figure 10 shows the probability of detecting an accident for each frame with different time intervals for extracting frames. The shorter the time interval, the more accurate the detection. The boxed portion of Fig. 10 shows that the probability of detecting an accident with a 1s interval is reduced because the sampling is too coarse. We will be able to achieve better performance by reducing time intervals as new dashcams adopt more powerful processors.

Each time a file is created, the dashcam creates and transmits the evidence of the video to the cloud, as shown in Fig. 11. The evidence of the video consists of the number (4 bytes), ID (21 bytes), file name (7 bytes), and two hash values (32 bytes each), totaling 96 bytes. This evidence data is created at 1-min intervals and sent to the cloud. The method described in [3] generates a VP of 4584 bytes every minute. Compared with [3], our study reduces network transmissions by 98%. When an accident is detected in the dashcam video, the frames at the beginning and the end of the video, the encrypted video, and the encrypted key are transmitted to the cloud. The average size of one minute of video is 50 MB, each frame has an average size of 200 KB, and

the encrypted Keysess is 158 B long when using the AES encryption algorithm and a 256-bit key length. Although this is a large overhead, there is an advantage in that users can share videos without performing frequent backups. When dashcam storage is full, the oldest video is deleted. For example, with 64 GB cards, videos can be kept for 2-3 weeks with 1-2 hours daily driving [3].

Applications that utilize smart contracts are either computationally intensive or limited to storing large amounts of data [19]. To address this problem, the smart contract is designed to store only small amounts of data such as state variables, hash values, download links, and public keys. In addition, to reduce costs when searching, the cloud reads the data stored in the SC, stores it in the database, and performs the search process. Therefore, our scheme automatically shares the videos. Thus, it is possible to relieve the users of the burden of backing up and preserving past videos. That is, our system reduces the amount of time for which backups need to be kept, since accident participants seeking video will probably do so within a week. Both the requestor and the user can easily share a video.

Security analysis

In this section, we analyze the security of the proposed method in terms of integrity, confidentiality and location privacy. When an accident video is shared, there is potential for inadvertently sharing a manipulated video or disclosing the video owner's location. In this paper, we overcame these issues by using deep learning and a blockchain. Here we discuss how we guarantee the integrity, confidentiality, and privacy of the video.

Integrity

The dashcam creates evidence of a video every minute and sends it to the cloud. The evidence of the video includes the hash value of the previous video. This evidence is transferred to the cloud for ongoing management. When an accident video is identified, an EC is deployed, and the accident related data is transferred to the cloud. The cloud accesses the EC and reads the frame hash value stored in the EC to calculate the hash value of the video. This value is compared with the previously stored value. If a malicious user has manipulated the video, we will discover the alteration when the hash of the previous video does not match.

The video sharing process is done through a smart contract recorded on the Ethereum blockchain. The contract cannot be changed once deployed. The requester makes a request for a specific EC when deploying REQC. Only the owner of the video that deployed the EC through the smart contract can send the rekey and download link to the REQC and change the approval status. The cloud checks the REQC's approval status and sends the correct video through the EC's address when the requester makes their download request. No other person can intervene and transmit data; only the correct data is transmitted through the smart contract.

Confidentiality

To ensure confidentiality of the video, the video is encrypted with Key_{sess}, and Key_{sess} is encrypted with PK_{own}. Because Key_{sess} is created each time accident videos are detected, other videos from the same owner cannot be decrypted even if the requester acquires the Key_{sess}. The method for delivering Key_{sess} is implemented using a *PRE*. The owner of the video uses PK_{req} and SKown to generate a *rekey*. The *rekey* is stored in the REQC and sent to the requester. Only the requester deploying the REQC can read the rekey and the download link. Even if the *rekey* is leaked, the SK_{req} is needed to decrypt the data. Thus, our scheme prevents the data from being exposed.

Location privacy

In general, the method of uploading a video to a public institution on the web does not guarantee anonymity. The proposed scheme is based on a user's Ethereum account. Because no other personal information is included and uploading is only performed pseudonymously through the Ethereum account, the video can be shared without revealing the user's identity. In addition, the user's location cannot be tracked because the user uploads fragmentary location information when an accident occurs without continuously uploading the data with location information.

Conclusion

In this paper, we proposed an automatic dashcam video sharing system using deep learning and smart contracts in a blockchain; the system securely encrypts the accident videos and maintains user location privacy. The video sequences are created every minute and if identified with an accident are transmitted to the Cloud as accident evidence so that the integrity of the EC to be deployed can be verified later. We reduce communication overhead by selecting and sharing only the accident images through deep learning. In our system, accident sounds are learned in addition to accident imagery and are used as supplemental information in order to improve detection accuracy. In addition, the proposed system maximizes user convenience by automatically connecting the requester who wants to obtain specific video clips and the owner who wants to share his/her own videos.

In the future, applying additional methods of deep learning will allow the dashcam to act as a potential city watcher by sharing various scenes in addition to accident videos.

Acknowledgements
Not applicable

Authors' contributions
TK, IJ, YH searched the literature; TK, IJ drew the figures; TK, IJ, YH collected the data and analyzed it; TK, IJ designed the experiments and TK performed the experiments; TK, IJ, YH wrote the paper; IJ, YH improved the paper. All authors read and approved the final manuscript.

Funding
This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Korea (2017R1D1A1B03034950) and by the BK21 Plus project funded by the Ministry of Education, Korea (21A20131600011).

Competing interests
The authors declare that we have no competing interests.

Author details
[1] School of Electronics Engineering, Kyungpook National University, Daehakro 80, Bukgu, 41566 Daegu, South Korea.
[2] Electrical and Computer Engineering Department, University of Illinois at Urbana-Champaign (UIUC), 901 West Illinois St, Urbana-Champaign 61801, USA.

Received: 22 March 2020 Accepted: 11 August 2020
Published online: 26 August 2020

References

1. Park S (2016) Motives and concerns of dashcam video sharing. In: CHI conference on human factors in computing systems, pp. 4758–4769
2. Embrain Trendmonitor. https://trendmonitor.co.kr/tmweb/trend/allTrend/detail.do?bIdx=1549&code=0304&trendType=CKOREA
3. Kim M, Lim J, Yu H, Kim K, Kim Y, Lee S (2017) Viewmap: Sharing private in-vehicle dashcam videos. In: USENIX symposium on networked systems design and implementation (NSDI), pp. 163–176. USENIX
4. Hossain MS, Muhammad G, Abdul W, Song B, Gupta BB (2018) Cloud-assisted secure video transmission and sharing frame-work for smart cities. Fut Gener Comput Syst 83:596–606
5. Zhang J, Zhong S, Wang T, Chao H, Wang J (2020) Blockchain-based systems and applications: a survey. J Intern Technol 21(1):1–14
6. Keccak. https://keccak.team/keccak.html
7. Dworkin MJ, Barker EB, Nechvatal JR, Foti J, Bassham LE, Roback E (2001) Advanced encryption standard (aes). Federal Information Processing Standards Publication, New York, p 197
8. Mambo M, Okamoto E (1997) Proxy cryptosystems: Delegation of the power to decrypt cipher-texts. Cryptogr Inf Sec 83:54–63
9. Kim T (2019) A privacy-preserving dashcam video sharing on blockchain with automatic accident detection. Ph.D. thesis, Kyungpook National University, School of Elelctronics and Engineering
10. Agyekum KO, Xia Q, Sifah E, Gao J, Xia H, Du X, Guizani M (2019) A secured proxy-based data sharing module in iot environments using blockchain. Sensors 19:1235
11. Alharby M, van Moorsel A (2018) Blockchain-based smart contracts: A systematic mapping study of academic research. In: International conference on cloud computing, big data and blockchain (ICCBB), pp. 1–6. IEEE
12. Nguyen G, Kim K (2018) A survey about consensus algorithms used in blockchain. J Inf Process Syst 14(1):101–128
13. Xia Z, Tan J, Wang J, Zhu R, Xiao H, Sangaiah A (2019) Research on fair trading mechanism of surplus power based on blockchain. J Univ Comput Sci 25(10):1240–1260
14. Yang M, Zhu T, Liang K, Zhou W, Deng RH (2019) A blockchain-based location privacy-preserving crowdsensing system. Fut Gener Comput Syst 94:408–418
15. Lee Y, Rathore S, Park JH, Park JH (2020) A blockchain-based smart home gateway architecture for preventing data forgery. Hum Comput Inf Sci 10:9
16. Nakamoto S (2008) Bitcoin: A peer-to-peer electronic cash system. https://bitcoin.org/bitcoin.pdf
17. Szabo N (1996) Smart contracts : Building blocks for digital markets. In: Extropy
18. Delmolino K, Arnett M, Kosba A, Miller A, Shi E (2016) Step by step towards creating a safe smart contract: Lesson and insights from a cryptocurrency lab. In: International conference on financial cryptography and data security, pp. 79–94
19. Wohrer M, Zdun U (2018) Smart contract: Security patterns in the ethereum ecosystem and solidity. In: International workshop on blockchain oriented software engineering (IWBOSE). IEEE, pp. 2–8.
20. Solidity. https://solidity.readthedocs.io/en/v0.5.9/
21. Daghera GG, Mohler J, Milojkovic M, Marella PB (2018) Ancile: Privacy-preserving framework for access control and interoperability of electronic health records using blockchain technology. Sust Cities Soc 39:283–297
22. Inc N (2018) Umbral: A threshold proxy re-encryption scheme
23. Karpathy A, Toderici G, Shetty S, Leung T, Sukthankar R, Fei-Fei L (2014) Large-scale video classification with convolutional neural networks. IEEE, pp. 1725–1732.
24. Hssayeni MD, Saxena S, Ptucha R, Savakis A (2017) Distracted driver detection: deep learning vs handcrafted features. Electron Imag 20:26
25. Ng J, Hausknecht M, Vijayanarasimhan S, Vinyals O, Monga R, Toderici G (2015) Beyond short snippets: Deep networks for video classification. IEEE, pp. 4694–4702.
26. Ou C, Ouali C, Bedawi SM, Karray F (2018) Driver behavior monitoring using tools of deep learning and fuzzy inferencing. In: IEEE international conference on fuzzy systems (FUZZ-IEEE). IEEE, pp. 1–7.
27. C Streiffer C, Raghavendra R, Benson T, Srivatsa M (2017) Darnet: a deep learning solution for distracted driving detection. In: ACM/IFIP/USENIX Middleware Conference, pp. 22–28. ACM/IFIP/USENIX
28. Leonardo T, Francesco S, Luca B, Samuele S, Leonardo S, Matteo S, Alessandro L (2018) Classification of crash and near-crash events from dashcam videos and telematics. In: International conference on intelligent transportation systems (ITSC). IEEE, pp. 2460–2465
29. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv 1409.1556
30. Canziani A, Paszke A, Culurciello E (2016) An analysis of deep neural network models for practical applications
31. Xiao F, Lu M, Zhao Y, Menasria S, Meng D, Xie S, Li J, Li C (2018) An information-aware visualization for privacy-preserving accelerometer data sharing. Hum Comput Inf Sci 8:29
32. Ni L, Liu Y, Liu Y (2020) Privacy protection model for location-based services. J Inf Process Syst 16(1):96–112

33. Chen L, Lee W, Chang C, Choo KR, Zhang N (2019) Blockchain based searchable encryption for electronic health record sharing. Fut Gener Comput Syst 95:420–429
34. Knirsch F, Unterweger A, Engel D (2017) Privacy-preserving blockchain-based electric vehicle charging with dynamic tariff decisions. Comput Sci Res Dev 33:7179
35. Jia B, Zhou T, Li W, Liu Z, Zhang J (2018) A blockchain-based location privacy protection incentive mechanism in crowd sensing networks. Sensors 18:3894
36. Redmon J, Farhadi A (2018) Yolov3: An incremental improvement. arXiv:1804.02767v1
37. Nvidia. https://www.nvidia.com/en-us/
38. Wang J, Yang Y, Wang T, Sherratt R, Zhang J (2020) Big data service architecture: a survey. J Intern Technol 21(2):393–405
39. Deng J, Dong W, Socher R, Li L, Li K, Fei-Fei L (2009) Imagenet: A large-scale hierarchical image database. IEEE, pp. 248–255.
40. Googlea. https://www.youtube.com
41. CarCrashDetector. https://github.com/Giffy/CarCrashDetector
42. Logan B (2000) Mel frequency cepstral coefficients for music modeling. In: International symposium music information retrieval.
43. Keras. https://keras.io/
44. Tensorflow. https://www.tensorflow.org
45. MIVIA LAB. https://mivia.unisa.it
46. Audioblocks. https://www.audioblocks.com
47. He H, MA Y, (2013) Imbalanced learning: foundations, algorithms, and applications. IEEE Press, New York
48. Jeni L, Cohn J, Torre FDL (2013) Facing imbalanced data–recommendations for the use of performance metrics, pp. 245–251
49. Hanley JA, Mcneil B (1982) The meaning and use of the area under a receiver operating characteristic (roc) curve. Radiology 143:29–36

Publisher's Note