



Toward efficient and intelligent video analytics with visual privacy protection for large-scale surveillance

Nguyen Anh Tu¹ · Thien Huynh-The² · Kok-Seng Wong³ · M. Fatih Demirci^{1,5} · Young-Koo Lee⁴

Accepted: 3 May 2021 / Published online: 15 May 2021

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2021

Abstract

Nowadays, the explosion of CCTV cameras has resulted in an increasing demand for distributed solutions to efficiently process the vast volume of video data. Otherwise, the use of surveillance when people are being watched remotely and recorded continuously has raised a significant threat to visual privacy. Using existing systems cannot prevent any party from exploiting unwanted personal data of others. In this paper, we develop an intelligent surveillance system with integrated privacy protection, where it is built on the top of big data tools, i.e., Kafka and Spark Streaming. To protect individual privacy, we propose a privacy-preserving solution based on effective face recognition and tracking mechanisms. Particularly, we associate body pose with face to reduce privacy leaks across video frames. The body pose is also exploited to infer person-centric information like human activities. Extensive experiments conducted on benchmark datasets further demonstrate the efficiency of our system for various vision tasks.

Keywords Intelligent video analytics · Large-scale surveillance · Visual privacy · Human activity analysis · Big data · Apache spark

1 Introduction

With the rapid growth of surveillance systems, CCTV cameras have become ubiquitous [48]. Manually monitoring every video stream to gain insightful information is impractical and inefficient. Instead, using computer vision (CV) can reduce the need for manual monitoring and automate the analysis that provides actionable intelligence to users or security personnel [32]. The effectiveness of smart surveillance technology has been continuously improved as a safety and management tool

✉ Nguyen Anh Tu
tu.nguyen@nu.edu.kz

Extended author information available on the last page of the article

for many industrial applications such as risk monitoring [43], managing staff [2], healthcare [38], and smart home [61]. Even though it has many advantages, there are also some drawbacks to surveillance systems related to privacy issue, i.e., human identity in the videos.

For example, a primary school in Hangzhou China uses facial recognition to monitor the behavior and attentiveness of their students [14]. The system records student actions such as writing, reading, raising a hand, and sleeping at a desk. It can also recognize several facial expressions. However, the system received many criticisms from the parents, students, and others who have concerns about what information is being collected or shared. The main reason is that the facial and body attributes not only provide details about the identity of a subject, but also reveal other person-related information such as gender, race, and age [12]. Such information is useful to profile a person when they are combined with external resources such as images and videos on social media. Hence, when human is the dominant objects of interest in CV applications, there is a need to protect the relationship between collection and dissemination of visual data, i.e., visual privacy [34].

Personal information like face and body is considered the critical cue to infer the person's identity from images or videos. As a result, this kind of visual information, especially faces, is sensitive and presents a significant threat to visual privacy. Specifically, in most person identification systems, facial features are commonly used to identify a human subject because the face is a valid point to determine the identity compared to other visual information such as body and gait [36]. The detected face, together with human identity, can be used to reveal additional person-related details. Without the permission of users, the outputs of these vision tasks can be exploited for various analysis purposes, which again threaten the privacy of individuals.

Hence, face recognition lies at the heart of visual privacy protection where human de-identification (e.g., mosaicking, blurring, and pixelation) can be used to mask the detected faces of human subjects in images or videos. Most of the current works [11, 41] have focused on improving the performance upon still images. However, video-based face recognition has been paid less attention from the community, even though it is more challenging. In particular, a human subject can repeatedly appear over multiple frames. Consequently, some faces of the subject at different scenes might not be recognized due to abrupt pose changes or motion blur. Failed face recognition at one scene is sufficient to leak out the subject's identity and others in the video [9, 17]. Thus, a valuable surveillance system needs to not only address well the privacy issue from visual data but also enable real-time and accurate identification across video frames.

Concurrently, the development of surveillance cameras in public areas (e.g., nursery and school) has grown the vast volume of visual data much faster than ever before [47]. Handling such big data storage and computation management is extremely challenging. Typically, most vision tasks demand more time and powerful resources for effectively mining and extracting insights from the video content, especially when using robust deep learning models. As a result, due to the limit of memory storage and computational capability, single machine systems with multiple processors usually fail to deal with the space-efficiency and time-efficiency problems. This leads to an increasing requirement for distributed solutions to efficiently

and effectively process a huge amount of visual data. However, most of the existing works [32, 67] are rarely carried out in a scalable and distributed manner. They are still mostly dependent on a traditional client/server framework with simple scheduling strategies. Also, current frameworks mainly focus on simple vision applications (e.g., motion detection, object recognition) while the complex vision tasks (e.g., activity recognition) are still not well-supported.

Considering the issues above, in this paper, we propose a comprehensive solution for surveillance systems. The objective is to meet the requirements of real-time video analytics wherein the system enables the processing and analysis of large-scale data of video streams efficiently but simultaneously preserving the privacy of individuals. Accordingly, we provide a versatile system with an extensive offering of parallel vision algorithms ranging from low-level face detection to middle-level pose estimation, face recognition, tracking, and high-level activity recognition. Our system is capable of supporting different types of services consisting of the monitoring service; online service for sending out actionable alerts; and offline service for the provision of activity data and life-log that can be useful to extract the lifestyle patterns and promote a healthier lifestyle. The privacy of these services is guaranteed to be protected.

In particular, the main contributions of this paper are described as follows:

- Our first contribution is the development of a novel surveillance framework benefiting from the high performance of distributed computing tools like Kafka [1] and Spark Streaming [66]. Kafka is adopted for exchanging messages between different modules, and Spark Streaming allows to parse heavy video stream in a scalable manner efficiently. The use of these big data tools will enable us to meet the elastic computing demands of practical surveillance. Notably, Apache Spark, one of the most popular representatives of new big data stacks, has been shown to tackle the challenges of scalable machine learning issues [69]. Meanwhile, Kafka is a valuable tool to support delivering the large amount of stream data with high-throughput and low-latency [25]. The need for running CV tasks with these tools is one of the reasons to inspire the design of our surveillance framework. In other words, our paper delves into the careful implementation of the system that enables efficient parallelization of various vision algorithms in the distributed environment.
- The second contribution is to incorporate visual privacy protections into the identification process. Specifically, we adopt the data association technique presented in [24] to protect the identity of a subject where the face detector and 2D body pose estimator [8] based on deep learning models are combined to improve the face tracking and recognition. Hence, privacy leaks across video frames are considerably reduced. We leverage the body pose based on the fact that although the face appearance of the target subject might change in different scenes, its body appearance is almost maintained. Human de-identification is subsequently applied so that the users can only see the face of their registered subject. Furthermore, to ensure that the identity is preserved in real-time, we propose the algorithm for efficiently parallelizing face detection, body estimation, and data association. This can be done by using the supported Spark operations (e.g., map

and flatMap) to split the video stream into multiple partitions and process them in parallel on multi-node clusters. As a result, we can significantly accelerate the identification process while effectively handling the big data storage imposed by the massive video stream.

- In the third contribution, we introduce a simple yet effective method that captures the dynamics of 2D poses (formed as skeletons) to compute features and predict human activities. This method employs body joints and adopts Fourier Transform to obtain the video representation used for activity prediction. The high-level information (i.e., predicted activities) is valuable to extract the lifestyle patterns and exploitable to generate healthcare recommendations. Accordingly, we describe a parallel and distributed algorithm to efficiently extract the skeletal features used for activity classification on the Spark clusters. Interestingly, the skeleton's use also helps to preserve the anonymity of targets because the identity-related information (e.g., body shape, clothes) is discarded. In other words, the skeleton is considered as privacy-protected data which is safely shareable to other cloud analytic services. It is different from the majority of existing works [50, 51, 53] in activity analysis using the human body from raw images without privacy concerns. To the best of our knowledge, this is the first work to utilize 2D pose to both reduce the privacy leaks and recognize the activities of multiple individuals. Notably, unlike 3D pose [23, 44] generated from depth cameras (e.g., Kinect), the 2D pose is particularly well-suited for surveillance scenarios where 2D cameras are widely employed [4].
- Finally, we verify the efficiency of our proposed framework in the wide variety of video analytics experiments, such as privacy-preserving person identification and activity recognition. Our experimental results on benchmark datasets demonstrate that the scalability of the proposed framework is promising when we run these vision tasks on the Spark clusters and achieve the significant acceleration of the analysis process.

The rest of the paper is organized as follows. We present the privacy requirements of practical surveillance and related works in Sect. 2. We show our architecture overview in Sect. 3. We explained the user registration and secured data storage, and privacy-preserving video streaming in Sects. 4 and 5, respectively. The experiment results and analysis are presented in Sect. 6. The conclusions and future work are in Sect. 7.

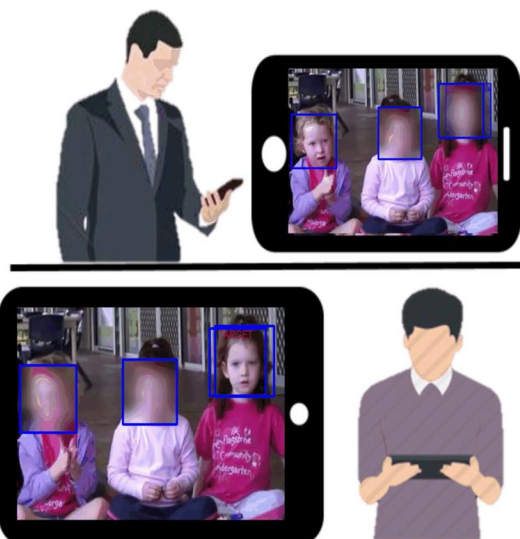
2 Preliminary and related work

2.1 Requirements of visual privacy and efficient computation in large-scale video surveillance

In this research, the surveillance system should provide sufficient anonymity [21] such that no visual information can be linked to the identity of any human subject in the videos, and no party can learn a subject's activity by observing the video stream. Also, no party can impersonate an authorized user to access the system.

All users must be aware of the risk of visual privacy leakage [34]. To further understand these requirements, let us consider a use case of video surveillance system used at a nursery place. In common practice, parents monitor kids at nursery via CCTV stream that is linked to their mobile devices. This convenience also allows other parents to observe the behavior of other children in the CCTV footage. However, this increases the risk of identity theft because the intruders can learn the lifestyle of a target victim. In this use case, we can see several types of visual privacy invasion based on different user's behavior. For example, parents or staff may want to collect personal visual information from others. In another case, a party may upload a funny CCTV footage on social media and causes a severe cyberbullying problem that may affect every part of a victim's lives and causing deep emotional issues. On another occasion, someone may capture a scene of the video stream from an authorized user's device for malicious purposes. A possible solution to protect children's identity is to mask their faces and show different views for different users based on access permission. As illustrated in Fig. 1, when two users view the same stream video, they only can see their child while other kids are masked. Otherwise, CCTV cameras at the nursery place produce a lot of stream data required for continuous monitoring and capturing children's behaviors in real-time. Therefore, it would be more efficient and convenient to integrate intelligent video analytics into big data stacks seamlessly (e.g., Hadoop and Spark) for handling and exploiting such heavy CCTV streams. Accordingly, in the following subsections, we briefly review research studies that are closely related to our proposed framework and satisfy one of the requirements mentioned above. These works include the protection of visual privacy and large-scale video surveillance systems.

Fig. 1 Example of privacy-protected nursery surveillance system



2.2 Human de-identification for visual privacy protection

In the first line of works, there exists a growing body of human de-identification approaches which can be divided into ad-hoc methods and replacement of face with artificial faces. Earlier de-identification methods are ad-hoc methods such as mosaicking, blurring, and pixelation. Some works in the literature intentionally processed video to be in special low-quality conditions [15, 40, 45]. This kind of anonymization technique only allows for the recognition of some target events in the video. The contradiction of using ad-hoc methods is between high privacy protection and low video utility. A higher level of privacy protection can be obtained by replacing the face with another, unrelated face. Bitouk et al. [6] introduced a system that replaces faces by selecting a similar face from a database of real face images. However, the construction of such a database for real applications is ethically and legally problematic. A viable alternative was to use synthetic face images for face replacement, e.g., in the work of Newton et al. [33], who proposed the k-same face de-identification algorithm. The works on full-body de-identification received less attention, although a higher level of de-identification than face-only de-identification was needed [7]. In another work, Orekondy et al. [35] presented a model to redact private information on images by analyzing the trade-off between data privacy and utility. The generative adversarial network (GAN) has been widely used in recent works [27, 39, 59] where the faces were generated by training both the generator and the discriminator of the GAN. Specifically, Ren et al. [39] formulated an adversarial learning framework to simultaneously learn a face anonymizer and activity recognizer. In [59], the authors introduced two external modules consisting of the verifier to enable a larger range of sampling exploration in GAN's generator, and the regulator to preserve the structure similarity according to a single input which better balances the trade-off between image quality and privacy protection. More recently, Lin et al. [27] built a GAN model with a generator based on convolutional neural networks (CNNs) and two self-designed discriminators to improve the discrimination accuracy. Apart from GAN-based methods, the authors in [17] presented a feed-forward encoder-decoder network architecture to de-identify faces in the video at high frame rate and provided natural outputs with preserved expression. Although these works have proved their effectiveness in preserving human identity, none of them showed the ability to be efficiently applied in real-life surveillance scenarios. Unlike these de-identification approaches, our work gives specific focus on how to automatically identify people and protect their identity in the real-time manner. Then, the basic de-identification (e.g., ad-hoc methods) is applied to preserve the anonymity of people. It is worth noting that any advanced de-identification approaches (e.g., [17, 27, 35, 59]) that we mentioned above are applicable to our framework.

2.3 Large-scale video analytics

In recent years, although the smart surveillance systems have been widely developed, there are a limited number of works concentrated on the design of the

distributed architecture for large-scale processing. Tan and Chen [49] proposed the parallel processing system on Hadoop clusters that handled large-scale of video data and performed several video analytics tasks such as face detection, motion detection and tracking. Zhang et al. [67] introduced a cloud-based architecture that provided both real-time processing with Apache Kafka and offline batch data analysis based on the Hadoop MapReduce framework. Later on, Yi et al. [64] developed the low-latency video analytics on edge computing platform by offloading the computation between clients and edge nodes as well as collaborating nearby edge nodes. In [63], authors also adopted MapReduce operations to design an object classification system. Apache Spark [66] with in-memory computing capability [(i.e., Resilient Distributed Datasets (RDDs)], which is a fast and general-purpose engine for large-scale data storage and processing, has emerged as an alternative to inherit attractive features of Hadoop MapReduce. Accordingly, in [62], Yang et al. proposed a parallel video data analysis framework based on Spark to perform action detection and video retrieval. The parallel-computing power of Spark was also exploited in [30] for face recognition from massive videos. Apart from the popular distributed platform like Spark and Hadoop, the video analytics systems have been constructed using cloud and edge computing infrastructures. For example, Chen et al. [10] built the distributed deep learning model for an intelligent surveillance system which was deployed in a multi-layer edge computing architecture. The performance of their system on parallel training was tested by running two analysis tasks, i.e., vehicle classification and traffic flow prediction. Meanwhile, Yaseen et al. [63] proposed a cloud-based object recognition system from large number of video streams which are automatically fetched from the cloud storage and analysed in an unsupervised manner. However, most of these works were only developed for specific vision tasks rather than exploring the comprehensive solutions for practical surveillance. Moreover, different from our work, the visual privacy concern is not considered in these works. Importantly, this paper brings together the research area of visual recognition, big data, and visual privacy protection those have attracted a great deal of interest from scientists and researchers in recent years. Hence, there are increasing demands for designing and developing an intelligent surveillance framework that integrates cutting-edge technologies for these areas.

3 Architecture overview

This section presents the framework of the video surveillance system that supports the whole operating process of visual analysis, including data collection, data processing, privacy protection, and data storage. As shown in Fig. 2, the framework is composed of the following components:

Data source. A set of IP cameras that are placed in monitoring regions and continuously collect the surveillance video data. These data will be sent as live video streams to other components to be processed further.

Message broker. This component is in charge of exchanging messages and communication among different modules. It works as middle-ware of data buffer between system modules (e.g., data source component and data processing component) and

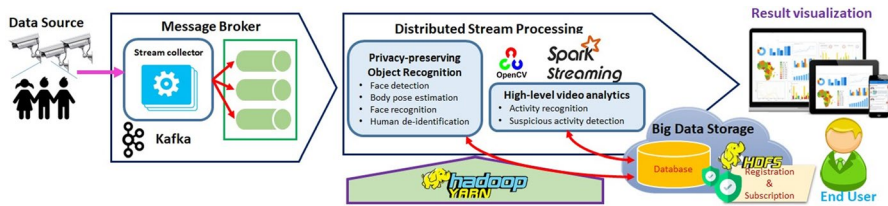


Fig. 2 Architecture of the proposed surveillance framework

hence enables the effective decoupling based on the distributed message queue. Thus, using the message broker can improve both the fault-tolerance and the extensibility of the surveillance system (i.e., a new module is easily connectable without any prior modification in the system).

Distributed stream processing. This component is the main engine and responsible for parsing the massive video streams by using Spark Streaming. Within this component, multiple visual recognition tasks are implemented under the distributed setting. Furthermore, these visual tasks are embedded in different modules to empower the flexibility of our system.

Big data storage. Persistent Big Data Storage relies on the Hadoop Distributed File System (HDFS) responsible for storing both unstructured data (e.g., video streams and learned models) and structured data (e.g., user profile and logs). It also provides storage facilities to store and manage access rights, metadata of video streams, and service subscription information.

Our surveillance system is built on the top of high-performance computing tools, i.e., Apache Kafka for handling the message broker, Apache Spark for distributed processing, and Apache Hadoop Yarn [52] to schedule computing resources. According to the framework components shown in Fig. 2, we will explain the details of data storage design and distributed stream processes in Sects. 4 and 5, respectively.

4 User registration and secured data storage

The end-user is authorized to upload their personal information and data streams. According to users' demand, personal data will be subscribed to relevant services of video analytics. Notably, users need to register our streaming service through the application installed on their devices. After the registration, the user can upload their data to private cloud storage, which is associated with our stream processing service. We store personal data, such as face images and user information in the facial database. Our streaming services also can visualize the results and push the notification in both online and offline manner. Our persistent data storage is developed on the top of HDFS that performs data read-write access operations and data management. Specifically, the data streams stored in the cloud can be automatically fetched or transferred between the cloud storage and HDFS to be processed. Besides using private cloud storage, since we develop our system on Hadoop and Spark clusters,

it is possible to connect our video analytics services with other popular cloud platforms like Google Cloud Storage (GCS) and Amazon Simple Storage Service (S3). This can be done using the Hadoop Compatible FileSystem and allowing big-data processes (e.g., Hadoop or Spark jobs) access to underlying data from GCS or S3 [13]. Therefore, the users can benefit from their preferred cloud service to store the data and still fully utilize our video analytics system. They can also access our video surveillance services without any implementation knowledge. However, it is essential to note that the users should be aware of third-party cloud storage's security and privacy risks. For instance, data on this type of cloud storage can be lost through a malicious attack, natural disaster, or a data wipe by the service provider. Also, an employee within the organization may misuse the credential to gain access to the user account, CCTV footage, and other sensitive information stored in the cloud. Hence, this clearly shows the usability and flexibility of our system. In addition, the life-log data and activity data persisted in the Big Data Storage are regularly synchronized to support generating the personalized recommendation. Then, security and privacy components are further embedded into the cloud storage service that requires authentication and data stream encryption before persisting and sharing data.

To safeguard both the private data and analytics processes from malicious attacks, we need to ensure that unauthorized parties are unable to access (read or modify) personal data in cloud storage. However, authorized parties such as data analysts should be able to efficiently access all (or parts) of this data as necessary. To prevent unauthorized access to cloud storage files, we propose using a cloud-based secure big data storage system [68]. The system ensures the data is secure even in case of partial secret key leakage because it will periodically update the secret keys. Also, technique such as secure fine-grained access control [60] can be used to enforce access control to stored data while allowing for efficient data retrieval. Besides access control (authentication and authorization), we can utilize video encryption solutions to protect the videos in the cloud storage [37]. For example, a cloud-fog-local video encryption service framework proposed in [26] provides a uniform video encryption service for devices with different computing power. Since the focus of this paper is not on video security solutions, we refer readers to a recent survey paper presented in [65]. Also, it is worth noting that the users should ensure the mobile devices used are protected with strong authentication and the latest patches and updates have been applied. For example, the client's devices are tamper-proof with secure hardware elements that prevent physical tampering. Since the mobile security is outside the scope of the paper, we refer readers to a recent study [5] that discusses different privacy-security perceptions of the mobile apps.

5 Privacy-preserving video streaming

In this section, we describe the video streaming scheme that offers prominent vision tasks in a distributed environment. Moreover, we incorporate privacy protections into our system to protect human identities. Notably, we adopt big data technologies to handle large-scale video stream for a distributed system reliably. Video stream

analytics involves a series of tasks, including object detection, tracking, face recognition, and action recognition. These tasks are deployed by parallel processing of video streams and analyzing the visual data with different machine learning and CV libraries. Furthermore, we use deep learning (DL) libraries integrated with Spark to efficiently run CNN-based tasks under distributed settings.

Our overall stream processing is divided into two main components: a message broker and a distributed stream processor. The stream data from a cluster of IP cameras are continuously fed into the stream collector of the broker, which serializes video frames to stream data buffer. The message broker is responsible for sending and receiving messages of different modules in a fault-tolerant fashion. The stream data are then consumed from the buffer and processed by the distributed stream processor, where we will apply a variety of vision algorithms. Finally, the processed data will be stored in the HDFS directory. Our video stream processing is designed using OpenCV, Apache Kafka, and Apache Spark frameworks. It is worth noting that the design of our system allows us to overcome the typical limitations of traditional video analytic systems. This is because previous systems usually collect and process videos at the same time, and a server failure will cause the loss of stream data. Otherwise, data fragmentation might easily occur when traditional scheduling tasks detect the failure node and switch to another node. Meanwhile, our system decoupled from two separate components not only enables efficient task recovery and scheduling but also reliably handles the scalable and fault-tolerant system.

5.1 Message broker using Apache Kafka

Our stream collection component employs Apache Kafka as a distributed messaging model to collect and deliver a large amount of video data with high throughput and low latency. The Kafka pipeline excels in excellent speed and durability for the development of real-time applications. Kafka is comprised of a producer, a consumer, and a broker. A producer publishes messages to a specific category called a topic. The consumer reads these messages and processes them in real-time. To manage the communication between producers and consumers, a Kafka broker stores streams of records like keys, values, and timestamps in topics. Here, the use of a topic consisting of partitions is to ensure high throughput. It also enables multiple consumers to read the mini-batch of stream data from the Kafka broker. To ensure fault-tolerance, the Kafka broker works as a buffer queue storing messages in temporary storage and partitions are replicated to multiple brokers.

5.2 Distributed stream processing with spark streaming

This component as shown in Fig. 3 is a vital part of our distributed video processing system. Using Spark Streaming, the distributed video processing is performed with a stateful data computing scheme and a dynamic amount of resources on the Spark cluster. Every node in the cluster will be potentially able to provide meta-data about each video processing task over every possible video frame. Here, we employ a discretized stream (DStream) provided by Spark Streaming API to



Fig. 3 Workflow of video processing with Kafka and Spark Streaming

consume and process JSON messages from Kafka. Video data is ingested into the system by Kafka producers, and the Spark cluster is then cast as a consumer of the Kafka message queue to consume the data. A DStream that contains data from a certain interval is represented by a sequence of RDDs. Any Spark operation applied on a DStream is relevant to operations (e.g., Map, Reduce, Join) on RDDs. For the streaming operations, we use the OpenCV and distributed machine learning libraries [31] with a variety of built-in algorithms.

5.3 Privacy-preserving object recognition

In this subsection, we describe the first module in the streaming component. It includes object recognition in conjunction with privacy-preserving processes. We begin with methods to recognize the face from videos. Besides, the body pose is concurrently estimated to be associated with the faces. Because face and body contain sensitive information that can be leaked out to reveal the person's identities, several techniques to protect visual privacy are utilized. The methodology of these tasks will be detailed in the following.

5.3.1 Object recognition without tracking

The major steps of object recognition are illustrated in Fig. 4. Our objective is to efficiently detect and recognize objects (i.e., face and body) from live videos. Since the output is related to personal identity, this information needs to be identified in every frame. The straightforward method is to recognize the face frame by frame through two main stages, i.e., face detection [28] and face recognition [41]. Concurrently, the human pose estimator, i.e., OpenPose [8] will be used to localize the human body, which is also served as the input of high-level video analytics. Here, the problem of body pose estimation is to predict the location of various human key points such as elbows, knees, neck, shoulder, hips, and chest. Then, the OpenPose is employed to perform the multistage classification with multiple CNN branches where each stage enhances the outputs of the previous one. The key points for people who appeared in the frame are connected using greedy inference to generate the bounding boxes of human bodies. Finally, to include the identity in each human body, we combine the outputs of face recognition and pose estimation. This is done by measuring the overlap between two regions of the recognized face and estimated head pose.

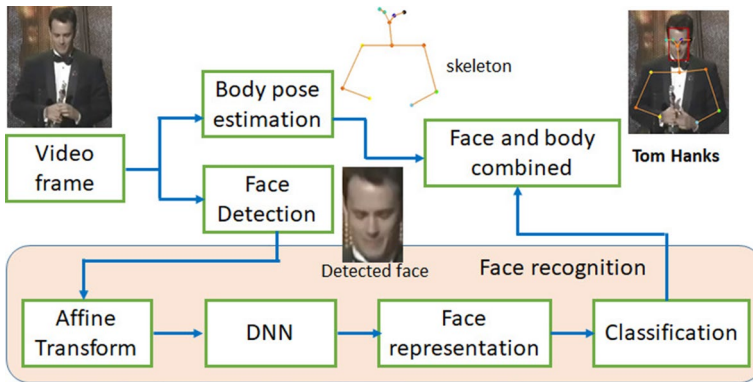


Fig. 4 Workflow of object recognition. For face recognition, the detected face is aligned via the affine transform and passed through the deep neural network (DNN) to compute the feature vector which is fed into the standard classifier to output the person’s identity

Subsequently, we focus on executing the parallel algorithm of object recognition in the distributed environment. This algorithm is implemented using the Spark paradigm to parallelize different vision tasks (i.e., face detection, face recognition, and body estimation) on multi-node clusters. The parallelization can be done by partitioning video stream into frames placed into RDD and processed afterward in parallel by Spark operations within each worker node. Note that Spark supports two types of parallel operations on RDDs including transformations (e.g., *map* and *flatMap*) and actions (e.g., *reduce*). Algorithm 1 describes parallel processing of video streams with Spark from face detection, body estimation to classification. Initially, a *flatMap*() function takes the video stream segment (VSS) as input to read all frames and place them into the Frame RDD. It is worth noting that a *flatMap*() function is executed by Spark workers and will be applied to each of VSSs to obtain all RGB frames from them. After conducting the *flatMap* transformation, we load and pass the pre-trained face detector and pose estimator to the *map*() functions which transform all RGB frames into a set of face and body detections. These detections are then transferred to another *map*() function to recognize the detected faces. Finally, the output placed in the Frame RDD is the combination of the skeleton and human identity.

Algorithm 1 Object recognition without tracking

Input: RDD[VSSID, Video data \mathbf{V}]

Output: RDD[VSSID, $\{(personID, Skeleton)\}$]

1. **flatMap** given a video stream segment \mathbf{V} :
 - (a) **For each** frame \mathbf{I}_t in \mathbf{V} **do**:
 - Emit $\langle VSSID, frame \mathbf{I}_t \rangle$
 - (b) **End for**
 2. **End flatMap**
 3. **Map** given frame \mathbf{I}_t of VSS in Frame RDD:
 - (a) $Faces_t = FaceDetector(\mathbf{I}_t)$
 - (b) $Skeletons_t = PoseEstimator(\mathbf{I}_t)$
 - (c) Emit $\langle VSSID, (Faces_t, Skeletons_t) \rangle$
 4. **End Map**
 5. **Map** given set of faces and skeletons $\{Faces_t, Skeletons_t\}$ of VSS in Frame RDD:
 - (a) $\{personID, Skeleton\} = FaceRecognizer(Faces_t, Skeletons_t)$
 - (b) Emit $\langle VSSID, \{personID, Skeleton\} \rangle$
 6. **End Map**
-

5.3.2 Object recognition with tracking

It will be time-consuming if we perform object detection and recognition in every frame. This might fail to meet the requirement of real-time surveillance. To avoid this critical issue, the process of object recognition will be accelerated significantly by employing object tracking algorithms [57]. This is because computing the relative motions between two successive frames in object tracking is considerably cheaper with a smaller number of candidates to be recognized. Note that, by using the object tracking, the input of face recognition can be a sequence of tracked faces rather than a single face used for the recognition without tracking. As a result, it provides the richer face representation and is useful to improve recognition accuracy.

Face and body association (FBA). Because we utilize human bodies for further analytics, it also needs to be tracked simultaneously with faces. For that reason, we apply the data association technique presented in [24] to keep tracking the identity of the detected body across different scenes. The association forms the score function to determine whether the detected face and the detected body are from the same person. In each frame, these objects combined via the head joints returned by OpenPose [8]. The association is based on the fact that although the face appearance of the target subject might not be clearly visible (e.g., small scale or non-frontal faces) in different scenes, its body appearance is almost maintained. Therefore, identity tracking across frames will be more reliable. Figure 5 illustrates the workflow when incorporating the face and body association.

Accordingly, for object recognition with tracking, we present a parallel and distributed algorithm that emphasizes parallelizing FBA to achieve high efficiency and speed-up for object tracking. As described in Algorithm 2, we begin with distributing face detection and pose estimation tasks across video frames using flatMap() and

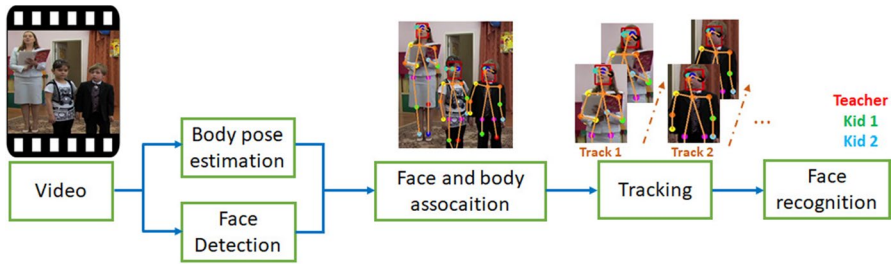


Fig. 5 Workflow of object recognition with face and body association

map() functions. It is important to note that object detection is time-independent, where we can process each frame independently of others without any effect on the detection accuracy. Meanwhile, the data association technique like FBA is time-dependent since the inter-frame dependencies exist when the object moves across consecutive frames. Therefore, independently performing FBA might degrade the tracking accuracy. To effectively address this issue, we develop the association technique that considers this time-dependency constraint and increases the parallelism for object tracking. Specifically, using groupByKey() function, we merge the output of face and body detections from flatMap() function into video shots, which are the consecutive sequence of frames. Then, FBA is performed on each video shot by passing the object tracker to map() function to transform a set of object detections into a set of tracklets. This process is equivalent to splitting the video stream into different shots and applying FBA on each shot afterward. In this way, we can guarantee a certain degree of parallelism for object tracking. Subsequently, we use groupByKey() function to combine tracklets obtained from the map tasks. These tracklets are then sorted and joined in order by using the TrackerMergeing method in another map() function to generate the set of final object trajectories. Finally, the sequence of tracked objects is applied to the face recognizer to obtain a set of person identities with corresponding skeletons in each video stream.

Algorithm 2 Object recognition with tracking

Input: RDD[$VSSID, Video\ data\ V$]

Output: RDD[$VSSID, \{personID, seq. of skeletons\}$]

1. **flatMap** given a video stream segment V :
 - (a) **For each** frame I_t in V **do**:
 - Emit $\langle (VSSID, shotID), (frameID\ t, I_t) \rangle$
 - (b) **End for**
 2. **End flatMap**
 3. **Map** given frame I_t of VSS in Frame RDD:
 - (a) $Faces_t = FaceDetector(I_t)$
 - (b) $Skeletons_t = PoseEstimator(I_t)$
 - (c) Emit $\langle (VSSID, shotID), (frameID\ t, Faces_t, Skeletons_t) \rangle$
 4. **End Map**
 5. **groupByKey** given the key $(VSSID, shotID)$ and a set of object detections $\{Faces_t, Skeletons_t\}$ of VSS in Frame RDD:
 - (a) $\{Faces, Skeletons\} = sort(\{frameID\ t, Faces_t, Skeletons_t\})$
 - (b) Emit $\langle (VSSID, shotID), \{Faces, Skeletons\} \rangle$
 6. **End groupByKey**
 7. **Map** given the sorted set of object detections in Shot RDD:
 - (a) $tracklets = Tracker(\{Faces, Skeletons\})$
 - (b) Emit $\langle (VSSID, shotID), tracklets \rangle$
 8. **End Map**
 9. **groupByKey** given the key $VSSID$:
 - (a) Emit $\langle VSSID, \{shotID, tracklets\} \rangle$
 10. **End Map**
 11. **Map** given the set of tracklets in each video stream segment:
 - (a) $\{sortedTracklets\} = sort(\{shotID, tracklets\})$
 - (b) $\{trackedObject, seq. of Skeletons\} = TrackerMerging(\{sortedTracklets\})$
 - (c) $\{personID, seq. of Skeletons\} = FaceRecognizer(\{trackedObject, seq. of Skeletons\})$
 - (d) Emit $\langle VSSID, \{personID, seq. of Skeletons\} \rangle$
 12. **End Map**
-

5.3.3 Combining object recognition with privacy protection

Once the process of person identification is completed, we need to hide the identity of observed subjects by removing person-identifiable information. Using the de-identifying process, the transformed outputs cannot be used to track back and search for the person and objects related to an incident. This is achieved by employing image processing techniques to obscure the person-identifiable contents. Consequently, some of the critical privacy concerns can be alleviated. De-identification methods will generate bounding boxes to cover the regions of interest. The pixels inside bounding boxes can be modified to achieve a certain degree of content-obscuring. For example, the blurring can be adjusted by varying the Gaussian parameters. Then, after the registration, users can only see the face of authorized subjects as shown in Fig. 1.

In particular, the identity of the registered subject in the database will be matched with the recognized faces. The matched faces remain visible while the others will be obscured with the bounding boxes. Similar to face recognition, human de-identification that performs matching and obscuring is also fully automated. Note that the

efficiency of this process relies on the recognition speed and the number of authorized users. While the speed issue can be mitigated by accelerating face recognition with object tracking, the latter is efficiently handled by using our proposed distributed stream processing. On the other hand, the accuracy of face recognizer reflects the reliability of the privacy protector. In terms of recognition rate, as shown in [24], face and body association can significantly boost the performance compared to face recognition without association or tracking. As a result, FBA prevents privacy leakage much better. We will further demonstrate the effectiveness of our privacy protection method through the qualitative results in the experiments.

5.4 High-level video analytics

The outputs of object recognition enable users to monitor their subjects of interest under privacy-preserving constraints and can be processed further to infer person-centric information such as human activities. Specifically, FBA allows us to identify the bounding boxes of multiple persons associated with their identities. These bounding boxes will be tracked in parallel by adopting a multi-target tracking algorithm. Subsequently, each person track becomes the input of high-level video analytics like activity recognition. In this scenario, human activity recognition (HAR) is capable of detecting the abnormal (e.g., the kid is falling) or suspicious activities (e.g., an intruder breaks into the school) over a period of time. For example, the detected activities will be notified or sent out as alert immediately to the user in an online manner. Meanwhile, for offline service, the sequence of activities such as “sitting” and “running” can be stored in the form of lifelog to generate a personalized recommendation.

The life-log and activity data are persisted in big data storage as shown in Fig. 2 of our architecture. According to the user request (e.g., require to retrieve historical data like daily activities), big data storage also supports the customized queries for data streaming. To satisfy the demand of the personalized experience, all the contents and processes will be performed and stored in a secure and privacy-preserving way.

5.4.1 Pose-based human activity recognition

Given a track of a single person, the objective is to assign activity labels to each bounding box. For this task, we make the best possible use of the human skeleton provided by the pose estimator. Particularly, the skeleton’s joint positions are detected from each frame and then employed to extract skeletal features. Meanwhile, the popular features in HAR, such as dense trajectory [53] and C3D [50], have a common limitation, which is the lack of interpretability. The skeletal feature possesses an attractive property that is more interpretable, intuitive, and concise to alleviate distinguishing different human activities. Interestingly, the pose estimator can also play the role of privacy protector because it thoroughly hides the identity-related information like body shapes and clothes. Thus, the skeleton itself is the

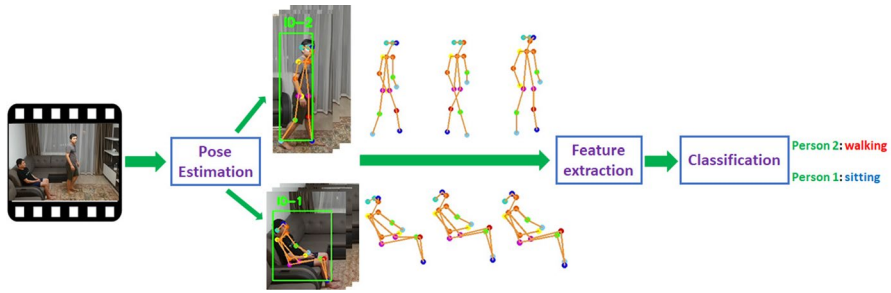


Fig. 6 Workflow of pose-based human activity recognition

Table 1 Notations for pose-based human activity recognition

Notation	Meaning
\mathcal{S}	A set of skeletal joint positions (JP)
$P(\mathcal{S})$	Feature matrix of JP coordinates derived from set \mathcal{S}
$VP(\mathcal{S})$	Feature matrix of JP velocities derived from set \mathcal{S}
$F(\mathcal{S})$	Final JP representation computed from set \mathcal{S}
$\hat{\mathcal{S}}$	A set of pairwise relative positions (RJP)
$Q(\hat{\mathcal{S}})$	Feature matrix of RJP coordinates derived from set $\hat{\mathcal{S}}$
$VQ(\hat{\mathcal{S}})$	Feature matrix of RJP velocities derived from set $\hat{\mathcal{S}}$
$F(\hat{\mathcal{S}})$	Final RJP representation computed from set $\hat{\mathcal{S}}$

privacy-protected data and can be safely shared or outsourced to other higher-level analysis services.

The overall workflow of pose-based HAR is shown in Fig. 6. Given the sequence of skeletons in T frames, a full set of skeletal joints are defined as:

$$\mathcal{S} = \{s_i^t = (x_i^t, y_i^t) | i \in [1, N], t \in [1, T]\} \tag{1}$$

where N is the number of body joints, i -th joint s_i^t is a point with 2D coordinate at frame t . Note that the original coordinates of body joints are normalized to be irrelevant to the absolute body position as well as the body size. Then, a sliding window collects the skeletal data on every T frames, and its output is an activity label for each bounding box. To meet the real-time requirement, the window is slid frame by frame along the time dimension of video streams.

In this work, we use two types of skeletal representation: 2D coordinates of the joint positions (JP) and pairwise relative positions of the joints (RJP). To capture the dynamics of the body poses in spatial and temporal domains, we use the coordinate of each position and measure its velocities between consecutive frames. Following [54], we further adopt Fourier temporal pyramid (FTP) to address noises and temporal misalignment issues in each window. Fourier coefficients are then aggregated to achieve the final representation fed into a classifier to perform activity recognition. Table 1 summarizes the notations used for our pose-based HAR.

In particular, for JP, the set of skeletal joints \mathcal{S} is arranged into two feature matrices P and VP , which represent the concatenation of JP coordinates and the JP velocities, respectively, as follows:

$$P(\mathcal{S}) = [P_1, P_2, \dots, P_N] \tag{2}$$

$$VP(\mathcal{S}) = [VP_1, VP_2, \dots, VP_N] \tag{3}$$

Here, each matrix $P_i \in \mathbb{R}^{2 \times T}$ for $i = 1, \dots, N$ is the concatenation of joint position i across T frames as follows

$$P_i = [\mathbf{s}_i^1, \mathbf{s}_i^2, \dots, \mathbf{s}_i^T] = \begin{bmatrix} x_i^1 & x_i^2 & \dots & x_i^T \\ y_i^1 & y_i^2 & \dots & y_i^T \end{bmatrix} \tag{4}$$

Meanwhile, the matrix $VP_i \in \mathbb{R}^{2 \times T}$ is the concatenation of velocities for each joint position i as the following expression:

$$VP_i = [\mathbf{s}_i^1, (\mathbf{s}_i^2 - \mathbf{s}_i^1), \dots, (\mathbf{s}_i^T - \mathbf{s}_i^{T-1})] = \begin{bmatrix} x_i^1 & (x_i^2 - x_i^1) & \dots & (x_i^T - x_i^{T-1}) \\ y_i^1 & (y_i^2 - y_i^1) & \dots & (y_i^T - y_i^{T-1}) \end{bmatrix} \tag{5}$$

As a result, two feature matrices $P(\mathcal{S})$ and $VP(\mathcal{S})$ have the same size of $2 \times T \times N$. Moreover, as described in Eqs. 4 and 5, the elements of these matrices can be formed as time-series, hence we apply the $FTP()$ function for each element separately and concatenate Fourier coefficients to obtain the final JP representation as follows:

$$F(\mathcal{S}) = [FTP(P_1), FTP(VP_1), \dots, FTP(P_N), FTP(VP_N)] \tag{6}$$

For the RJP representation, we first convert the set \mathcal{S} in Eq. 1 into a set $\hat{\mathcal{S}}$ of pairwise relative positions as following:

$$\begin{aligned} \hat{\mathcal{S}} &= \{\mathbf{q}_{ij}^t = \mathbf{s}_i^t - \mathbf{s}_j^t | 1 \leq i < j \leq N, t \in [1, T]\} \\ &= \{\hat{\mathbf{q}}_k^t | k \in [1, \hat{N}], t \in [1, T]\} \end{aligned} \tag{7}$$

Where $\hat{N} = \frac{N(N-1)}{2}$ is the number of pairwise relative positions of the joints $\{\hat{\mathbf{q}}_k^t\}$. Accordingly, the calculations of two feature matrices in terms of RJP coordinates and RJP velocities are expressed as follows

$$Q(\hat{\mathcal{S}}) = [Q_1, Q_2, \dots, Q_{\hat{N}}] \tag{8}$$

$$VQ(\hat{\mathcal{S}}) = [VQ_1, VQ_2, \dots, VQ_{\hat{N}}] \tag{9}$$

Here, each element $Q_k = [\hat{\mathbf{q}}_k^1, \hat{\mathbf{q}}_k^2, \dots, \hat{\mathbf{q}}_k^T] \in \mathbb{R}^{2 \times T}$ for $k = 1, \dots, \hat{N}$ is the time-series of pairwise relative position k across T frames. In addition, the element matrix $VQ_k = [\hat{\mathbf{q}}_k^1, (\hat{\mathbf{q}}_k^2 - \hat{\mathbf{q}}_k^1), \dots, (\hat{\mathbf{q}}_k^T - \hat{\mathbf{q}}_k^{T-1})] \in \mathbb{R}^{2 \times T}$ is expressed as the time-series for the velocities of pairwise relative position k across consecutive frames. Hence, the size of two feature matrices $Q(\hat{\mathcal{S}})$ and $VQ(\hat{\mathcal{S}})$ with respect to pairwise relative positions is $2 \times T \times \hat{N}$.

Similar to the JP representation, we also employ FTP to obtain the final RJP representation. This can be done by applying the $FTP()$ function to each element of $Q(\hat{S})$ and $VQ(\hat{S})$ separately. The resulting Fourier coefficients are then concatenated together to form the final RJP representation as following expression

$$F(\hat{S}) = [FTP(Q_1), FTP(VQ_1), \dots, FTP(Q_{\hat{N}}), FTP(VQ_{\hat{N}})] \quad (10)$$

Finally, activity recognition is conducted by using the linear SVM to classify the final representation which could be either $F(S)$ or $F(\hat{S})$. It is worth noting that the 2D pose used in our approach is different from the 3D pose extracted by Kinect cameras [54]. Despite the great success of 3D pose-based HAR in the literature, Kinect still possesses many limitations, especially its applicability in the surveillance scenarios where most CCTV systems are equipped with 2D cameras.

5.4.2 Distributed human activity recognition

To run the HAR model on the distributed system, we further propose a parallel algorithm for feature extraction that can be used to efficiently classify human activities. We achieve this by parallelizing the computation of the final representation $F(S)$ or $F(\hat{S})$ for JP or RJP, respectively, to speed up the process of feature extraction and effectively handle the large-scale data streams, which are regarded as a sequence of RDDs. These data stream RDDs are created from the object recognition module wherein Kafka is employed for transmitting intermediate data to HAR module. As described in the above subsection, we can see that the process of feature extraction is time-dependent because the final representation based on FTP is derived from the sequence of skeletons whose joints are moving from one position to another. Therefore, to avoid degrading the HAR performance, it is important to take the inter-frame dependencies into account during this stage. On the other hand, it is worth noting that we can perform the computation independently on each spatial location of the body joint or pairwise relative joint. Accordingly, after taking a track of skeletons associated with a person identity from data stream RDDs, we use `flatMap()` function to distribute the tasks across joint positions and perform the subsequent computations in parallel on Spark workers. Then, the `map()` function is applied to compute the elements of feature matrix of coordinates ($P(S)$ or $Q(\hat{S})$) and feature matrix of velocities ($VP(S)$ or $VQ(\hat{S})$). These elements are then passed to the $FTP()$ function and use `groupByKey()` function to obtain the final representation ($F(S)$ or $F(\hat{S})$). The process of computing the JP representation on Spark is described in Algorithm 3. Similarly, we can apply the same algorithm to compute the RJP representation by simply replacing S with \hat{S} , and using the respective computations (Eqs. 8 and 9) of feature matrices Q and VQ . Finally, the distributed version of SVM from Spark MLlib [31] is employed to classify these skeletal representations into activities.

Algorithm 3 Process of computing skeletal representation (JP) on Spark**Input:** RDD[$personID, Seq. skeletons S$]**Output:** RDD[$personID, Final representation F(S)$]

1. **flatMap** given a set of skeletal joints S :
 - (a) **For each** joint position i in the set S **do**:
 - Emit $\langle personID, \{s_i^t\}_{t=1}^T \rangle$
 - (b) **End for**
2. **End flatMap**
3. **Map** given the joint position i :
 - (a) Compute the element P_i of feature matrix for coordinates using Eqs. 2 and 4
 - (b) Compute the element VP_i of feature matrix for velocities using Eqs. 3 and 5
 - (c) Compute $FTP(P_i)$ and $FTP(VP_i)$ using Fourier temporal pyramid.
 - (d) Emit $\langle personID, [FTP(P_i), FTP(VP_i)] \rangle$
4. **End Map**
5. **groupByKey** given the key $personID$:
 - (a) Emit $\langle personID, F(S) \rangle$
6. **End Map**

5.4.3 Activities log protection

Once a human subject's identity is protected, our system can perform various vision tasks to automatically produce personal activity data. For offline service, this data can be further adopted to model the high-level context of long-term behaviors. Then, these types of life-log data enable us to extract the lifestyle patterns of target subjects and facilitate user by providing personalized recommendation services. Here, given the children's activities, recommendations can be personalized by using content-based filtering mechanisms [20]. Subsequently, user needs are correctly mapped to the best possible recommendations based on inference rules. Some healthy recommendations .such as "encouraging the children to do exercises if they slept too much" and "W-sitting is not recommend for kids." The activities log contains sensitive contextual information of the human subjects and are vulnerable to adversarial attacks. Therefore, there is a need to guarantee that only authorized parties can use the log to perform required analytical tasks. With a secure big data storage system and access control mechanism, we can prevent unauthorized access to the activities log files. However, they remain susceptible to re-identification attacks [18, 22] through background information and cross-correlation with publicly available resources even when the personal information is removed from the life-log data. To prevent privacy leakage from the log files in case of lost or stolen, we can utilize homomorphic encryption that supports operations in an encrypted form [3, 19].

6 Experimental results and analysis

6.1 Settings

Our video surveillance framework is implemented on the Spark cluster consisting of 11 nodes: one master node, one node as the Kafka broker, and nine worker nodes used for different vision tasks. In particular, we divide these worker nodes into three groups: two nodes for face detection; five nodes for body estimation, and two nodes for tracking and recognition tasks (i.e., recognize face and activity). We assign the resource for each group based on the workload and time consumption of tasks performed in that group. Each node had the same configuration: 4 cores running at 2.2 GHz and 32 GB memory. Each worker node has an NVIDIA GTX 1080 Ti with 11 GB memory. The Hadoop version was 2.7, the Spark version was 2.3.3, and the Kafka version was 2.2.

6.2 Datasets

We report experimental results on four large-scale video datasets consisting of: YouTube Faces (YTF) [58], IARPA Janus Benchmark B (IJB-B) [56], MSR DailyActivity [54], and NTU RGB-D [42]. While YTF and IJB-B are used to evaluate the performance of person identification, MSR and NTU RGB-D are employed to validate the efficiency and correctness of HAR algorithms.

Person identification: YTF contains 3,425 videos of 1,595 subjects, with an average of 2.15 videos per for subject and an average length of 181.3 frames per video clips. IJB-B consists of 7,245 videos and 1,845 subjects with 4.2 videos per subject on average. Videos in IJB-B are collected from various sources under large variation of the pose, scenes, and video length.

Human activity recognition: MSR DailyActivity provides 320 video clips with 16 activities. NTU RGB-D is one of the largest action recognition datasets which contains 56880 short video samples of 60 activity classes. Both MSR DailyActivity and NTU RGB-D datasets were captured by Kinect cameras to generate RGB videos and depth map sequences. However, in this work, we only use RGB videos to investigate the effectiveness of 2D pose estimator.

6.3 Evaluation criteria

We mainly focus on evaluating the scalability and the efficiency of different video analytics tasks. The scalability performance is measured in terms of the speedup. This is considered as an essential factor in measuring how much faster the parallel algorithm is when the number of computing units is increased. With the given computing resource, the efficiency is validated by varying the amount of video data applied to our system. This is done by calculating the time cost as the

number of streams grows. Besides, we further report the accuracy of video analytics based on HAR in different settings.

6.4 Performance of efficiency for object recognition

In this subsection, the performance is measured in accordance with the computing resource (i.e., the number of Spark nodes) of each vision task and the amount of data in terms of the number of video streams. To conduct experiments, we combine YTF and IJB-B datasets and divide them into four streams. Each stream(the length is about 10 h) is generated by concatenating video clips from the dataset. Consequently, the total length of full streams is 40 h.

Firstly, we evaluate the scalability of our framework when running different deep learning face detectors on the Spark cluster, where we employ Algorithm 1 described in Sect. 5.3 but exclude the use of pose estimator and face recognizer. Two versions of SSD face detector are examined in this experiment where one detector uses the built-in model from OpenCV, namely SSD-OpenCV, and the other uses the Mobilenet model, namely SSD-Mobilenet. The latter is run on both CPU and GPU. Since there are two nodes for face detection as described above, the comparative results are achieved as the number of nodes increased from 1 to 2. Obviously, as shown in Fig. 7a the results demonstrate the scalability of our framework, which significantly accelerates the face detection process. Remarkably, the speedup is approximately 1.8x when the number of computing nodes is doubled. SSD-Mobilenet with GPU run fastest at a speed of 84 fps on 1 node. SSD-OpenCV is two times slower, but its speed is still satisfactory in real-time. SSD-Mobilenet without GPU is too slow (6 fps) and not suitable for streaming service. Figure 7b shows the relationship between the time cost and the amount of the stream data. We can see that the processing time is almost linear when the number of streams is increased. Hence, the detection is expected to keep scaling well to larger datasets. This further shows that our distributed implementation effectively exploited parallelism. With GPU support, SSD-Mobilenet with GPU is consistently faster than SSD-OpenCV.

In the next experiment, we test the scalability of the pose estimation. Here, we reserve 5 Spark nodes with GPUs because the pose estimator (i.e., OpenPose) is heavily executed through CNN models. We also use Algorithm 1 without a face

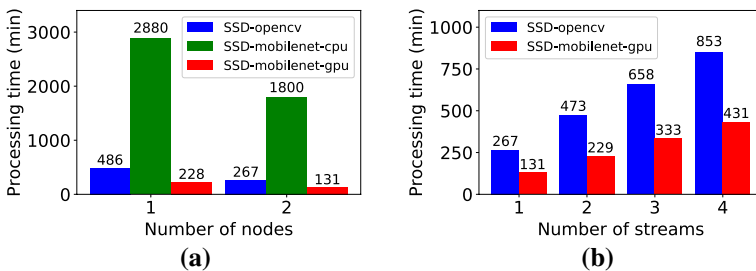


Fig. 7 a Scalability of different face detectors w.r.t number of nodes. b Efficiency of different face detectors w.r.t number of streams

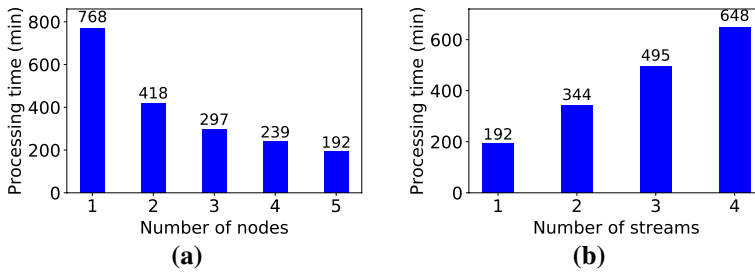


Fig. 8 **a** Scalability of pose estimation w.r.t number of nodes. **b** Efficiency of pose estimation w.r.t number of streams

detector and recognizer to perform the distributed pose estimation. As shown in Fig 8a, the time cost is proportional to the number of nodes. Considering 1 video stream fed into our surveillance, the time-consuming task like the pose estimation can be performed quickly at the speed of 100 fps. This further demonstrates the scalability of our framework since the speed up relative to the run time of one node is consistently increased as the number of nodes. Figure 8b illustrates the relationship between the time cost and number of streams when executing pose estimation. The processing time of this task is increased linearly as the size of stream data grows. With the full of streams, to complete the task, it takes nearly 10 h which is 4 times faster than the total length of the streams. This speed exhibits the acceptable efficiency for real-time scenarios. Thus, in this experiment, we can observe that the pose estimation benefits from parallelism.

Regarding person identification, we examine the efficiency of three face recognition approaches. The first approach performs face recognition without tracking as described in Algorithm 1. The second denoted as FA employs tracking based on face association using Algorithm 2 without a body estimator. The last denoted as FBA associates both face and body to improve the tracking accuracy as validated in [24]. For this approach, we fully use the distributed implementation of Algorithm 2. Apparently, as shown in Fig. 9a and b, face recognition without tracking is slower than the others. With tracking, we can accelerate the recognition speeds to 2 or 3 times. The reason is that given a video stream segment, the process of data

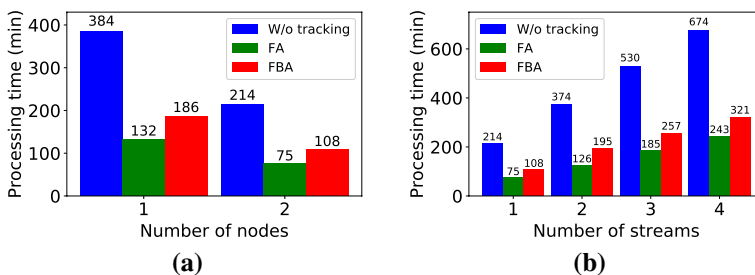


Fig. 9 **a** Scalability of face recognition w.r.t number of nodes. **b** Efficiency of face recognition w.r.t number of streams

association is much faster than the process of face classification, which is needed to be performed in every frame. Moreover, by leveraging the parallelism for object tracking, both FA and FBA are consistently faster than the recognition without tracking when we increase the number of nodes or streams. FBA is relatively slower than FA because the complexity of data association considering body pose is higher. The results from Fig. 9a, b once again verify the efficiency of our proposed parallel algorithms. All approaches scale well as the increasing resource. They efficiently handle large streams at an acceptable time cost. Especially, FBA achieves eight times speedup compared to the length of full streams.

6.5 Qualitative evaluation of visual privacy protection

The effectiveness of human de-identification is primarily dependent on the accuracy of face recognition and tracking, as discussed in Sect. 5, wherein the face recognition accuracy or identification rate can be measured as the percentage of the correctly recognized faces per the total number of tested faces. If the face is recognized incorrectly or the tracking is failed in several frames, it results in the privacy leaks. Consequently, the speed and accuracy of the face detector and tracker must be guaranteed to detect a new face quickly and de-identify it reliably as soon as possible. As shown in the results above, our system meets the real-time requirement for face detection effectively. Furthermore, most of the current face detector can only work well with the frontal view. However, if the small scale or non-frontal faces remain in many frames before it turns into the frontal view, the de-identification might be failed due to the late detection. Estimating human pose can address this issue because the body is always visible. It allows us to determine the head pose even though the face is not clearly visible. As shown in Fig. 10, the proposed method can produce better results of tracking identity when combining the pose estimator

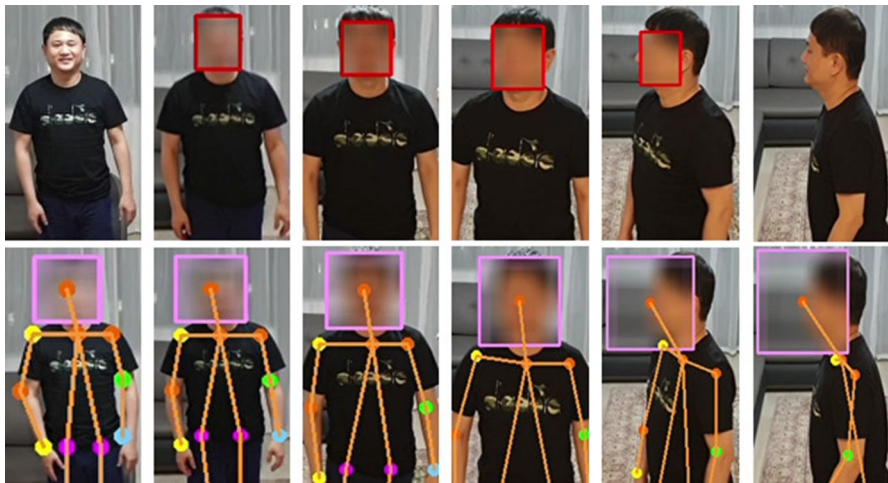


Fig. 10 Example of privacy leaks. Results on the first row are obtained by using only SSD face detector. Results on the second row are obtained by combining SSD face detector with tracking based on FBA

with the face detector. This further validates that our parallelization strategy of FBA handles well the time-dependency issue of data association technique as discussed in Sect. 5.3 where we can maintain both good qualitative performance and high speedup.

6.6 Performance of human activity recognition

In this experiment, we evaluate the HAR performance in terms of efficiency and accuracy. The experiments were conducted by using two activity datasets, i.e., NTU RGB-D and MSR Daily Activity. Particularly, NTU RGB-D is used to examine the efficiency of HAR processed on our framework. For this dataset, we also created four video streams and the length of each stream is 10 h. Meanwhile, MSR Daily Activity is used to evaluate the accuracy of different HAR approaches. In this work, we consider the scenario where the subject performs a sequence of various activities (e.g., walking, sitting, and reading book). Hence, we generate the stitched version of MSR Daily Activity which is built by simply concatenating individual activity instances into sequences. Each sequence contains three to five activities.

We first investigate the efficiency of different HAR methods. Especially, we compare our pose-based HAR approach with the state-of-the-art approach presented in [46]. For the pose-based HAR, we employ two skeletal features, i.e., joint positions (JP) and relative joint positions (RJP). The implementation of distributed HAR was described in Algorithm 3 (Sect. 5.4). The CNN-based approach uses the VGG model [46] to extract the CNN feature maps, which are applied to a classifier to get the prediction scores. Note that the pose-based approach (JP and RJP) run on CPU, and VGG runs on GPU due to its high complexity. The comparative results are shown in Fig. 11(a) and Fig. 11(b). We can see that JP achieves the best efficiency, even compared to VGG with GPU support. RJP is the most time consuming because of its higher dimensional feature vector. Due to the parallelization of computing skeletal representation, both JP and RJP scale well on our system where the computing tasks are effectively distributed to multiple Spark workers. However, JP with lower complexity handles the large-scale stream better that achieves 3 times speedup when running full streams.

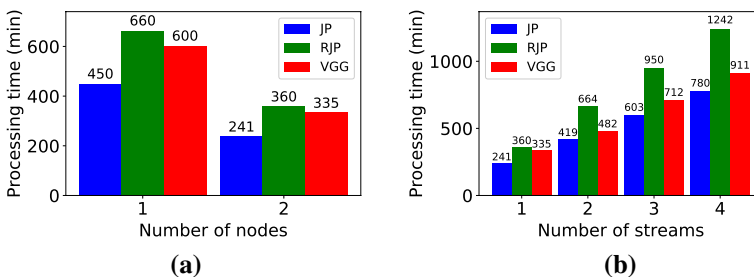


Fig. 11 **a** Scalability of different HAR approaches w.r.t number of nodes. **b** Efficiency of different HAR approaches w.r.t number of streams

Table 2 Accuracy comparison of different methods

Methods	JP	RJP	VGG [46]	VGG [55]	EHPI [29]	LSTM [4]
Accuracy (%)	81%	83%	72%	77%	79%	87%

Table 2 presents the accuracy comparison between different methods. The recognition rate is measured via frame-wise accuracy. Concretely, in our activity dataset, each video frame is associated with a label, and the overall frame-wise accuracy is the percentage of true positives after performing the prediction on the test video stream. Firstly, we consider two CNN-based methods. The first method [46] uses RGB images and VGG model for feature extraction. The second method presented in [55] is the advanced version of [46] that takes RGB images and RGB difference as the input to the VGG model. Here, RGB difference is employed to capture the dynamic appearance (i.e., the salient motion region) between two consecutive frames. Accordingly, as shown in Table 2, the combination of RGB images and RGB difference boosts the performance of [46]. Interestingly, we can observe that our pose-based methods outperform CNN-based methods even though they have lower complexity. This is because the skeleton is more informative and has higher interpretability, enabling us to differentiate human activities better. The accuracy of RJP is slightly higher, while JP balances the trade-off between accuracy and efficiency better. Then, we further compare our pose-based method with other works [4, 29] that also use OpenPose to extract 2D pose and build the skeletal representations. Note that we followed the standard-setting and the implementation protocols described in these works to obtain the results on our dataset, which were not reported in the original papers. Specifically, in [29], the authors proposed an approach to encode the sequence of skeletons over time in an image-like data structure, namely EHPI, which is then fed into CNNs for classification. The accuracy of this representation is lower than JP and RJP by 2% and 4%, respectively. This can be attributed to that, due to the use of FTP, our method better suppress the noises and handle temporal misalignment issues more effectively. Differently, the method in [4] explicitly takes the sequential nature of activities into account wherein the temporal dependencies are modeled using Two-Branch LSTMs. Due to the deployment of the sophisticated structure, the LSTM yields the best result with 87%. Despite the superior accuracy, it is difficult to parallelize the computations of this LSTM network because its inherent properties incur the dependency problem for both time and spatial locations of body joints. Hence, it might be extremely challenging to execute the algorithm in the distributed environment. Therefore, the obtained results in this experiment further demonstrate the effectiveness of the body pose, especially our skeletal representation, in improving HAR accuracy.

6.7 Detailed analysis on scalability

As described in our experimental settings (Sect. 6.1), we assign the resource for different vision tasks according to workload and time consumption. Consequently, five

Spark nodes with GPU supported are dedicated to the expensive pose estimation task that requires heavy computations of CNN models in OpenPose. Meanwhile, for less intensive computing tasks, i.e., face detection, face recognition, and activity recognition, we reserve only two Spark nodes. As shown in the previous experimental results, even with relatively limited computing resources, all these tasks can yield satisfactory speedups, which are well-suited for the real-time requirement of large-scale surveillance. To further demonstrate the scalability of our system, we examine the relation between the processing time and the number of nodes by increasing the reserved nodes from two to five for modules requiring less computation. The results are shown in Fig. 12a–c. We can see that the execution time costs in most cases decrease near-linearly when the number of nodes increases. In Fig. 12a, the ratio between the runtimes of two face detector versions (SSD-opencv and SSD-mobilenet-gpu) is approximately 2 on each number of nodes. The result of HAR behaves similarly in Fig 12c, where the runtime ratio between JP and RJP is consistently around 1.4 as increasing number of nodes. These indicate that our distributed implementations can maintain efficient parallelization by fully exploiting the characteristics of RDD operations. The result of face recognition shown in Fig. 12b is extended from the one in Fig. 9a, where we examine the performance of FBA. It also shows that FBA can achieve near-linear scalability. Therefore, given that execution is performed on large video streams, our implementation is expected to keep scaling on a larger number of nodes. Thus, combining the results of Fig. 12 with that of Fig. 8a, the scalability performance presented in this paper clearly verifies the robustness of our system when we vary the number of nodes from one to five to execute different vision tasks.

7 Conclusions

In this paper, we have proposed a comprehensive solution for large-scale surveillance systems. The proposed solution ensures visual privacy protection to individuals in the videos and supports intelligent video analytics ranging from low-level face detection to mid-level body pose estimation, face recognition, and high-level activity recognition. With our solution, we not only increase the confidence of users to use video surveillance systems but also raise public awareness about technologies

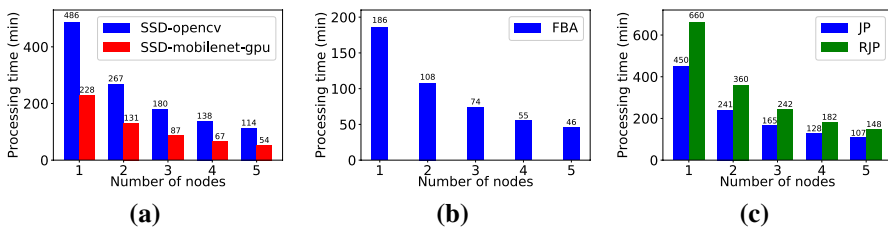


Fig. 12 Scalability w.r.t number of nodes for **a** face detection, **b** face recognition, and **c** human activity recognition

that place privacy at risk. Extensive experimental results on standard datasets verified the effectiveness and efficiency of our framework for different vision tasks. As future work, we plan to incorporate co-privacy [16] in our future work. The concept of co-privacy (or cooperative privacy) considers the best option for a party to obtain privacy protection is by helping another party achieve theirs.

Acknowledgements This work was supported by the Social Policy Grant and funded by the Nazarbayev University.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.


References

1. Apache Kafka (2020) <https://kafka.apache.org/>
2. Ajunwa I, Crawford K, Schultz J (2017) Limitless worker surveillance. *Calif Law Rev* 105:735
3. Alabdulatif A, Khalil I, Yi X (2020) Towards secure big data analytic for cloud-enabled applications with fully homomorphic encryption. *J Parallel Distrib Comput* 137:192–204
4. Avola D, Cascio M, Cinque L, Foresti GL, Massaroni C, Rodolà E (2019) 2-D skeleton-based action recognition via two-branch stacked LSTM-RNNS. *IEEE Trans Multimed* 22(10):2481–2496
5. Balapour A, Nikkhal HR, Sabherwal R (2020) Mobile application security: role of perceived privacy as the predictor of security perceptions. *Int J Inf Manag* 52:102063
6. Bitouk D, Kumar N, Dhillon S, Belhumeur P, Nayar SK (2008) Face swapping: automatically replacing faces in photographs. In: *ACM SIGGRAPH 2008 papers*, pp 1–8
7. Brkic K, Sikiric I, Hrkac T, Kalafatic Z (2017) I know that person: Generative full body and face de-identification of people in images. In: *CVPRW, IEEE*, pp 1319–1328
8. Cao Z, Simon T, Wei SE, Sheikh Y (2017) Realtime multi-person 2d pose estimation using part affinity fields. In: *CVPR*, pp 7291–7299
9. Chamikara MAP, Bertok P, Khalil I, Liu D, Camtepe S (2020) Privacy preserving face recognition utilizing differential privacy. *Comput Secur* 97:101951
10. Chen J, Li K, Deng Q, Li K, Philip SY (2019) Distributed deep learning model for intelligent video surveillance systems with edge computing. *IEEE Trans Ind Inf*
11. Chen JC, Patel VM, Chellappa R (2016) Unconstrained face verification using deep cnn features. In: *IEEE WACV*, pp 1–9
12. Chhabra S, Singh R, Vatsa M, Gupta G (2018) Anonymizing k-facial attributes via adversarial perturbations. In: *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pp 656–662
13. Cloud data access (2018) https://docs.cloudera.com/HDPDocuments/HDP2/HDP-2.6.5/bk_cloud-data-access/content/intro.html/
14. DAR P (2018) A chinese school is using facial recognition to analyze students' behavior. <https://www.analyticsvidhya.com/blog/2018/06/china-school-facial-recognition-analyse-students/>
15. Dai J, Saghafi B, Wu J, Konrad J, Ishwar P (2015) Towards privacy-preserving recognition of human activities. In: *ICIP, IEEE*, pp 4238–4242
16. Domingo-Ferrer J (2010) Coprivacy: towards a theory of sustainable privacy. In: *International Conference on Privacy in Statistical Databases*, Springer, pp 258–268
17. Gafni O, Wolf L, Taigman Y (2019) Live face de-identification in video. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp 9378–9387
18. Garfinkel SL (2015) De-identification of personal information. National institute of standards and technology
19. Gentry C et al (2009) Fully homomorphic encryption using ideal lattices. In: *Stoc*, vol 9, pp 169–178

20. Gong S (2012) Learning user interest model for content-based filtering in personalized recommendation system. *Int J Digital Content Technol Appl* 6(11):155–162
21. De Guzman JA, Thilakarathna K, Seneviratna A (2019) Security and privacy approaches in mixed reality: a literature survey. *ACM Comput Surv* 52(6):1–37
22. Henriksen-Bulmer J, Jeary S (2016) Re-identification attacks—a systematic literature review. *Int J Inf Manag* 36(6):1184–1192
23. Huynh-The T, Hua CH, Tu NA, Hur T, Bang J, Kim D, Amin MB, Kang BH, Seung H, Shin SY et al (2018) Hierarchical topic modeling with pose-transition feature for action recognition using 3d skeleton data. *Inf Sci* 444:20–35
24. Kim K, Yang Z, Masi I, Nevatia R, Medioni G (2018) Face and body association for video-based face recognition. In: *IEEE WACV*, pp 39–48
25. Landset S, Khoshgoftaar TM, Richter AN, Hasanin T (2015) A survey of open source tools for machine learning with big data in the hadoop ecosystem. *J Big Data* 2(1):24
26. Li H, Gu Z, Deng L, Han Y, Yang C, Tian Z (2019) A fine-grained video encryption service based on the cloud-fog-local architecture for public and private videos. *Sensors* 19(24):5366
27. Lin J, Li Y, Yang G (2021) Fpgan: Face de-identification method with generative adversarial networks for social robots. *Neural Netw* 133:132–147
28. Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu CY, Berg AC (2016) Ssd: Single shot multibox detector. In: *ECCV*, Springer, pp 21–37
29. Ludl D, Gulde T, Curio C (2019) Simple yet efficient real-time pose-based action recognition. In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, IEEE, pp 581–588
30. Lv J, Wu B, Liu C, Gu X (2018) Pf-face: A parallel framework for face classification and search from massive videos based on spark. In: *2018 IEEE Fourth International Conference on Multimedia Big Data (BigMM)*, IEEE, pp 1–7
31. Meng X, Bradley J, Yavuz B, Sparks E, Venkataraman S, Liu D, Freeman J, Tsai D, Amde M, Owen S et al (2016) Mllib: Machine learning in apache spark. *J Mach Learn Res* 17(1):1235–1241
32. Nazare AC Jr, Schwartz WR (2016) A scalable and flexible framework for smart video surveillance. *CVIU* 144:258–275
33. Newton EM, Sweeney L, Malin B (2005) Preserving privacy by de-identifying face images. *IEEE Trans Knowl Data Eng* 17(2):232–243
34. Orekondy T, Schiele B, Fritz M (2017) Towards a visual privacy advisor: Understanding and predicting privacy risks in images. In: *Proceedings of the IEEE International Conference on Computer Vision*, pp 3686–3695
35. Orekondy T, Fritz M, Schiele B (2018) Connecting pixels to privacy and utility: automatic redaction of private information in images. In: *CVPR*, pp 8466–8475
36. O’Toole AJ, Phillips PJ, Weimer S, Roark DA, Ayyad J, Barwick R, Dunlop J (2011) Recognizing people from dynamic and static faces and bodies: dissecting identity with a fusion approach. *Vis Res* 51(1):74–83
37. Pei D, Guo X, Zhang J (2017) A video encryption service based on cloud computing. In: *2017 7th IEEE International Conference on Electronics Information and Emergency Communication (ICEIEC)*, IEEE, pp 167–171
38. Plageras AP, Psannis KE, Ishibashi Y, Kim BG (2016) Iot-based surveillance system for ubiquitous healthcare. In: *IECON 2016-42nd Annual Conference of the IEEE Industrial Electronics Society*, IEEE, pp 6226–6230
39. Ren Z, Jae Lee Y, Ryoo MS (2018) Learning to anonymize faces for privacy preserving action detection. In: *ECCV*, pp 620–636
40. Ryoo MS, Rothrock B, Fleming C, Yang HJ (2017) Privacy-preserving human activity recognition from extreme low resolution. In: *Thirty-First AAAI Conference on Artificial Intelligence*
41. Schroff F, Kalenichenko D, Philbin J (2015) Facenet: A unified embedding for face recognition and clustering. In: *CVPR*, pp 815–823
42. Shahroudy A, Liu J, Ng TT, Wang G (2016) Ntu rgb+ d: a large scale dataset for 3d human activity analysis. In: *CVPR*, pp 1010–1019
43. Shao Z, Cai J, Wang Z (2017) Smart monitoring cameras driven intelligent processing to big surveillance video data. *IEEE Trans Big Data* 4(1):105–116
44. Shotton J, Fitzgibbon A, Cook M, Sharp T, Finocchio M, Moore R, Kipman A, Blake A (2011) Real-time human pose recognition in parts from single depth images. In: *CVPR 2011*, IEEE, pp 1297–1304
45. Simoens P, Xiao Y, Pillai P, Chen Z, Ha K, Satyanarayanan M (2013) Scalable crowd-sourcing of video from mobile devices. In: *Proceeding of the 11th Annual International Conference on Mobile Systems, Applications, and Services*, pp 139–152

46. Simonyan K, Zisserman A (2014) Two-stream convolutional networks for action recognition in videos. In: *Advances in neural information processing systems*, pp 568–576
47. Subudhi BN, Rout DK, Ghosh A (2019) Big data analytics for video surveillance. *Multimed Tools Appl* 78(18):26129–26162
48. Sultana T, Wahid KA (2019) Choice of application layer protocols for next generation video surveillance using internet of video things. *IEEE Access* 7:41607–41624
49. Tan H, Chen L (2014) An approach for fast and parallel video processing on apache hadoop clusters. In: *2014 IEEE International Conference on Multimedia and Expo (ICME)*, IEEE, pp 1–6
50. Tran D, Bourdev L, Fergus R, Torresani L, Paluri M (2015) Learning spatiotemporal features with 3d convolutional networks. In: *IEEE ICCV*, pp 4489–4497
51. Tu NA, Huynh-The T, Khan KU, Lee YK (2018) MI-hdp: A hierarchical Bayesian nonparametric model for recognizing human actions in video. *IEEE Trans Circuits Syst Video Technol* 29(3):800–814
52. Vavilapalli VK, Murthy AC, Douglas C, Agarwal S, Konar M, Evans R, Graves T, Lowe J, Shah H, Seth S, et al. (2013) Apache Hadoop YARN: Yet another resource negotiator. In: *Proceedings of the 4th Annual Symposium on Cloud Computing*, ACM, p 5
53. Wang H, Kläser A, Schmid C, Liu CL (2013) Dense trajectories and motion boundary descriptors for action recognition. *Int J Comput Vis* 103(1):60–79
54. Wang J, Liu Z, Wu Y, Yuan J (2012) Mining actionlet ensemble for action recognition with depth cameras. In: *CVPR*, pp 1290–1297
55. Wang L, Xiong Y, Wang Z, Qiao Y, Lin D, Tang X, Van Gool L (2016) Temporal segment networks: towards good practices for deep action recognition. In: *ECCV*, Springer, pp 20–36
56. Whitelam C, Taborsky E, Blanton A, Maze B, Adams J, Miller T, Kalka N, Jain AK, Duncan JA, Allen K, et al. (2017) IARPA Janus benchmark-B face dataset. In: *IEEE CVPRW*, pp 90–98
57. Wojke N, Bewley A, Paulus D (2017) Simple online and realtime tracking with a deep association metric. In: *ICIP*, pp 3645–3649
58. Wolf L, Hassner T, Maoz I (2011) Face recognition in unconstrained videos with matched background similarity. In: *IEEE CVPR*
59. Wu Y, Yang F, Xu Y, Ling H (2019) Privacy-protective-gan for privacy preserving face de-identification. *J Comput Sci Technol* 34(1):47–60
60. Xu S, Yang G, Mu Y, Deng RH (2018) Secure fine-grained access control and data sharing for dynamic groups in the cloud. *IEEE Trans Inf Forensics Secur* 13(8):2101–2113
61. Yang A, Zhang C, Chen Y, Zhuansun Y, Liu H (2019) Security and privacy of smart home systems based on the internet of things and stereo matching algorithms. *IEEE Internet Things J* 7(4):2521–2530
62. Yang S, Wu B (2015) Large scale video data analysis based on spark. In: *2015 International Conference on Cloud Computing and Big Data*, IEEE, pp 209–212
63. Yaseen MU, Anjum A, Rana O, Hill R (2018) Cloud-based scalable object detection and classification in video streams. *Futur Gener Comput Syst* 80:286–298
64. Yi S, Hao Z, Zhang Q, Zhang Q, Shi W, Li Q (2017) Lavea: Latency-aware video analytics on edge computing platform. In: *Proceedings of the Second ACM/IEEE Symposium on Edge Computing*, pp 1–13
65. Yu JY, Kim Y, Kim YG (2021) Intelligent video data security: a survey and open challenges. *IEEE Access* 9:26948–26967
66. Zaharia M, Chowdhury M, Das T, Dave A, Ma J, McCauley M, Franklin MJ, Shenker S, Stoica I (2012) Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing. In: *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation*, USENIX Association, pp 2–2
67. Zhang W, Xu L, Duan P, Gong W, Lu Q, Yang S (2015) A video cloud platform combining online and offline cloud computing technologies. *Pers Ubiquitous Comput* 19(7):1099–1110
68. Zhang Y, Yang M, Zheng D, Lang P, Wu A, Chen C (2018) Efficient and secure big data storage system with leakage resilience in cloud computing. *Soft Comput* 22(23):7763–7772
69. Zhou L, Pan S, Wang J, Vasilakos AV (2017) Machine learning on big data: opportunities and challenges. *Neurocomputing* 237:350–361

Authors and Affiliations

Nguyen Anh Tu¹  · Thien Huynh-The² · Kok-Seng Wong³ · M. Fatih Demirci^{1,5} · Young-Koo Lee⁴

Thien Huynh-The
thienht@kumoh.ac.kr

Kok-Seng Wong
wong.ks@vinuni.edu.vn

M. Fatih Demirci
muhammed.demirci@nu.edu.kz; mfdemirci@etu.edu.tr

Young-Koo Lee
yklee@khu.ac.kr

- ¹ Department of Computer Science, Nazarbayev University, 53 Kabanbay Batyr Ave, 010000 Nur-Sultan, Astana, Republic of Kazakhstan
- ² ICT Convergence Research Center, Kumoh National Institute of Technology, 61, Daehak-ro, Gumi-si, Gyeongsangbuk-do 39177, Republic of Korea
- ³ College of Engineering and Computer Science, VinUniversity, Vinhomes Ocean Park, Gia Lam District, Hanoi, Vietnam
- ⁴ Department of Computer Science and Engineering, Kyung Hee University, Global Campus, 1732 Deokyoungdae-ro, Giheung-gu, Yongin-si, Gyeonggi-do 446-701, Republic of Korea
- ⁵ Department of Computer Engineering, TOBB University of Economics and Technology, Sogutozu Cad, No: 43, Ankara, Turkey