Received: 5 January 2021 | Revised: 2 March 2021 | Accepted: 30 March 2021

DOI: 10.1111/exsy.12704

ORIGINAL ARTICLE

Expert Systems WILEY

A scalable framework for smart COVID surveillance in the workplace using Deep Neural Networks and cloud computing

Ajay Singh[1] | Vaibhav Jindal[1] | Rajinder Sandhu[1] | Victor Chang[2]

[1]Department of Computer Science and Engineering and Information Technology, Jaypee University of Information Technology, Solan, India

[2]Artificial Intelligence and Information Systems Research Group, School Computing, Engineering and Digital Technologies, Teesside University, Middlesbrough, UK

Correspondence
Victor Chang, Artificial Intelligence and Information Systems Research Group, School Computing, Engineering and Digital Technologies, Teesside University, Middlesbrough, UK.
Email: ic.victor.chang@gmail.com; v.chang@tees.ac.uk

Abstract

A smart and scalable system is required to schedule various machine learning applications to control pandemics like COVID-19 using computing infrastructure provided by cloud and fog computing. This paper proposes a framework that considers the use case of smart office surveillance to monitor workplaces for detecting possible violations of COVID effectively. The proposed framework uses deep neural networks, fog computing and cloud computing to develop a scalable and time-sensitive infrastructure that can detect two major violations: wearing a mask and maintaining a minimum distance of 6 feet between employees in the office environment. The proposed framework is developed with the vision to integrate multiple machine learning applications and handle the computing infrastructures for pandemic applications. The proposed framework can be used by application developers for the rapid development of new applications based on the requirements and do not worry about scheduling. The proposed framework is tested for two independent applications and performed better than the traditional cloud environment in terms of latency and response time. The work done in this paper tries to bridge the gap between machine learning applications and their computing infrastructure for COVID-19.

KEYWORDS
cloud computing, corona, COVID, deep neural networks, fog computing, pandemic

1 | INTRODUCTION

For the last decade, cloud computing has proved to be a prominent method of acquiring or provisioning IT resources for applications whose demand is variable with time (Malawski et al., 2015). Further, it also helps new start-ups and many organizations not to worry about their IT capital investment. However, cloud computing does impose an issue with the latency-sensitive application because cloud resources are located far away from the data generating devices (Mahmud et al., 2018). A new computing paradigm known as fog computing was introduced by Cisco in 2012, aiming to resolve the latency issue by providing IT resources physically close to the data generating devices. Fog computing is an extension of cloud computing, not its replacement, since the IT resources provided by fog computing are not that powerful, so cloud resources are needed at the back end (Verma & Chandra, 2020). Fog and cloud computing clubbed together have capabilities to develop frameworks that can provide scalable as well as virtually unlimited resources (Schooler et al., 2017). This kind of framework will be extremely useful for new applications developed by the developers to stop any pandemic like COVID-19.

Novel Corona Virus (COVID-19) is one of the biggest epidemics of the 20th century (Koonin, 2020). In order to develop and test a scalable cloud and fog computing framework for applications related to COVID-19, an automatic office surveillance system is selected, which is explained in the next paragraphs. COVID has caused a complete lockdown of multiple countries with a ban on international flights. However, slowly countries have started opening their economic activities, including the opening of offices and other workplaces (Lalwani et al., 2020). Although the governments have laid down Standard Operating Procedures (SOPs), it is challenging to monitor individuals following the same. For example, as per the SOPs in India, people should maintain a minimum distance of six feet between themselves to stop the spread of the COVID-19 virus. Organizations can appoint a few personnel to monitor whether the SOPs are followed by every member of the organization in open spaces such as cafeterias, entrances, and exits. However, it is hard to monitor them when they are working on their individual desks. Many employees work on their separate desks in open office layouts in different organizations across the world. This kind of layout exposes employees to easily contact other employees, which can further spread the virus in the organization (Yasaka et al., 2020). Based on the guidelines of WHO ("Advice for the public," 2020), there are two primary requirements to stop the spread of COVID. First, all personnel should wear a mask when they are in proximity to their colleagues. Second, everyone should maintain a distance of six feet between them. There is a need for an artificial intelligence assisted surveillance framework which can detect the violations and warn the employees and the employers about them. This can lead to a safe workplace, which can help the organization sustain in this pandemic.

Many systems, similar to office surveillance, are proposed after the pandemic by various researchers (Griebel et al., 2015; Hong et al., 2020; Park et al., 2020; Rahman et al., 2020; Won Sonn & Lee, 2020; Yasaka et al., 2020). Similarly, many implementations were observed using different platforms to find whether people are wearing masks or keeping a safe distance. However, most of the systems available on the internet are inadequate in how these systems would be deployed on computing infrastructure and their respective QoS. The machine learning-based applications for pandemics like COVID-19 are highly resourced dependent and latency-sensitive. The systems that can make multiple predictions but are not scalable to the required amount are worthless for the governments and medical agencies to control pandemics like COVID. There should be a framework which can integrate different machine learning model and provide the desired QoS from the underline infrastructure. The main objective of this paper is to develop a framework that can manage the IT resource demands of these machine learning-based prediction system whose predictions are resource heavy and time-sensitive. The proposed framework for these kinds of applications should be able to integrate any new module as required by the government or medical agencies. In order to test the proposed framework in this paper, which uses cloud and fog computing resources, two applications are used. The first application uses a deep neural network to predict whether a person sitting inside an office is wearing a mask or not. The second application again uses a deep neural network to predict whether the people in the offices keep a distance of a minimum of 6 feet. The use cases used in this paper are only to test the viability of the proposed framework on the IT resource part. However, the proposed framework can be implemented for an application whose output is time-sensitive, have multiple modules of different resource requirements, and is cost-effective.

In order to achieve the above-said objectives, a smart office surveillance system is proposed, which uses DNN algorithms to detect the violations and distributed computing requirements to fog and cloud computing. The proposed framework uses multiple cameras installed in organizations, such as CCTV feeds and the webcam of individual computers. These cameras will detect two violations: whether an individual is wearing a mask and the distance between two individuals is maintained as per the SOPs. The camera installed at the entrances of buildings, floors, and blocks will count the number of individuals entering an area. If the number of individuals entering an area is more than the designated seats, it will alert the organization. Furthermore, a webcam of individual computers will check whether a person sitting in front of the computer is wearing a mask or not. If that person does not have his mask on his face, his computer will be automatically locked, and he will not be allowed to work until he wears a mask again. These two applications will be hosted on fog and cloud-based IT infrastructure. Major novelties of the proposed framework are:

i. Provide scalable computing infrastructure to different machine learning applications of any pandemic by handling QoS requirements so that application developers can develop without worrying about infrastructure and QoS.
ii. Train fog computing neural network based on the errors generated by dense and deep neural network deployed at cloud computing infrastructure.

The proposed framework uses both fog (Oueis et al., 2015) and cloud (Mell & Grance, 2011) computing infrastructure for the computation part. All the computation tasks are distributed to two layers: Fast Layer (FL) and Accurate Layer (AL). The FL layer manages the latency sensitivity of the proposed framework. FL layer is hosted on the fog computing architecture, thus closer to the smart application. The computing power of FL is not high, so it stores small neural networks in terms of input, output and number of hidden layers. The FL with a small number of hidden layers is fast in detecting the violations, but their accuracy may not be high. However, the neyral network at FL will not be accurate enough so in the proposed framework AL layer is used to train the weights of the neural network at FL. AL has denser DNNs that use multiple hidden layers with many neurons. The AL has the highest accuracy, but it requires computation devices with high configurations, and they may be slow in detecting the violations. The high configuration here means the elastic cloud computing resources, which can be provisioned or removed based on the demands of the AL neural network. Thus, computation power would not be a constraint at the AL layer. In the proposed framework, both

FL and AL work hand in hand in detecting real-time violations and overall violations. The FL uses the fog computing platform, which has computation devices with less power, but the latency is minimum, whereas AL deploys its DNN on cloud computing infrastructure. Aneka (Wei et al., 2017) has been used to evaluate the proposed framework with both fog devices and the public cloud. The results from the performance evaluation of the proposed framework are positive with high accuracy and least response.

The rest of the paper is divided as follows. Section 2 discusses the related work of COVID and the use of artificial intelligence and machine learning techniques for predicting and preventing COVID infections. Section 3 explains the proposed framework in detail. Section 4 provides the experiment results and performance evaluation of the proposed framework. Lastly, Section 5 concludes the paper.

2 | RELATED WORK

There was a sudden increase in the publications related to COVID after it has impacted multiple countries. Researchers from all fields like medical, bioinformatics, data science, mathematics, and computer science shifted their focus on predicting, controlling, and monitoring COVID. In this section, papers related to the development of frameworks or architectures using machine learning are discussed briefly. All papers are from the year 2020 unless stated otherwise. This section will also discuss the basic idea of classification used by many researchers in machine learning projects related to COVID-19.

Classification is a process of selecting an input instance and classifying it into one of the pre-selected classes. It may also be seen as a function that takes X as input and predicts Y, where Y is a set of pre-defined classes. For example, a classification algorithm can be developed to classify all emails coming to an account into two classes: spam mail and legitimate mail, as shown in Figure 1 below.

2.1 | Use of ICT for controlling pandemics like COVID-19

The South Korea quarantine management team ('Coronavirus Disease-19: Quarantine Frame-Work for Travelers Entering Korea', 2020) developed a data science-based system that helped them manage citizens returning from overseas countries. They deployed and trained networks that ask various parameters and predict a suitable place for their quarantine. They argued that it has helped remove any human error that can be proved catastrophic in this kind of pandemic. Park et al. (Park et al., 2020) developed an online symptom investigation tool for managing an epidemic. This proposed tool is machine learning-based and the major benefit includes that it removes the patient need to go out for symptoms checking. Likewise, Hong et al. (Hong et al., 2020) surveyed the role of telemedicine in the era of COVID and concluded that it plays a great role. For the same, Sun et al. (Sun et al., 2020) developed a smartphone-based for the rapid tests of coronavirus. Smart cities have a significant role in controlling corona pandemic or any other epidemic in the future and this aspect has been discussed by Sonn and Lee (Sonn et al., 2020; Won Sonn & Lee, 2020) in their paper. They took the case study of smart cities of South Korea and discussed how they helped and what are different ways we can improve them. Furthermore, Yigitcanlar et al. (Yigitcanlar et al., 2020) studied that if all cities in the world had been smart, we would have controlled the COVID in - stages. They also pushed the government to pursue making cities smart in war speed. Similarly, Rahman et al. (Rahman et al., 2020) also discussed the role of deployed smart internet of things devices in smart cities for controlling any pandemic. Modelling in various forms, such as spread prediction, can help officials to work accordingly. The government can prepare based on the modelling; however, many modelling predictions went untrue for the corona epidemic. This may be because the governments took initiatives early based on the modelling of various frameworks. Ibarra-vega (Ibarra-Vega, 2020) studied the role of modelling and how it can help governments extend, impose, or enforce the lockdowns. Proactive contact-tracing is proved to be an effective way to curb the spread of corona and many believe it is the only way without any effective medicine in the market (Abeler et al., 2020; Armbruster & Brandeau, 2007; Yasaka et al., 2020).

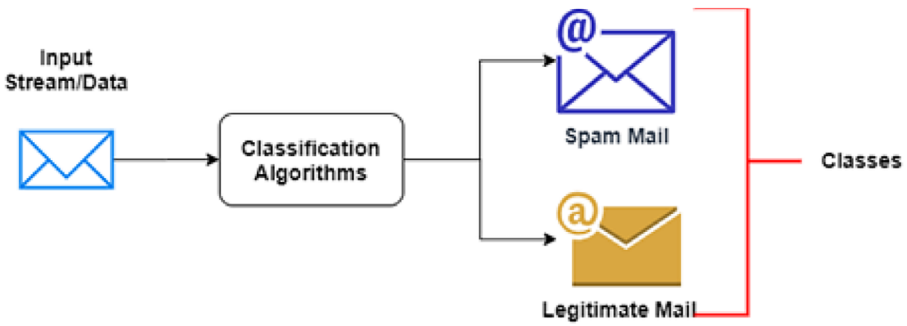

FIGURE 1 Basic working of classification

2.2 | Use of machine learning for controlling COVID-19

Machine learning and artificial intelligence contain various techniques; however, few have been repeatedly used by various researchers in the case of COVID. In this paragraph, a few common techniques are discussed. Most of the papers used the corona dataset available on the Kaggle website, whether it is X-Ray or symptoms data ("COVID-19 Dataset | Kaggle," 2020). Sujath et al. (Sujath et al., 2020) used various machine learning techniques and compared them for creating a forecasting model for corona. They implemented vector autoregression, multilayer perceptron, and linear regression methods. Zhang et al. (Zhang et al., 2020) discussed the role of time-fractional derivatives in forecasting and predicting the corona's growth rate. The major benefit of this method is that it uses data until the last timestamp and extends the prediction based on recent trends. As new data keeps coming, it auto-adjusts itself and provides the most recent forecast. As claimed by them, they were successful in predicting the peak of various countries using their model. However, different regression method for predicting peak cases has been used by various researchers. It is proved to be incorrect for many countries, including India (Parbat, 2020). It is safe to conclude by having actual data that regression methods should not be used in predicting unknown pandemics. A complex model taking various parameters as input is required and should be used in the future, such as (Zhang et al., 2020). One more aspect is the requirement of computing infrastructure for deploying these machine learning models. As such, numerous people infected or under quarantine are in a large number and thus, simple computing infrastructure will not be sufficient. Cloud computing is proposed by a few researchers for using complex machine learning architecture such as deep neural networks (Tuli et al., 2020). Melin et al. (Melin et al., 2020) proposed a neural network-based self-organized map for predicting the flow of spread. This kind of framework is useful for declaring containment zones and their respective buffer zones. One of the successful predicted models was proposed by Lalwani et al. (Lalwani et al., 2020). They predicted the optimal lockdown period of 73 days in the early phase of the pandemic and the actual lockdown was done for 77 days. This is one of the closed predictions among all other models to the best of the knowledge of authors. Salgotra et al. (Salgotra et al., 2020) tested the role of evolutionary algorithms in predicting the corona pandemic by applying genetic algorithm variations. While there are various published frameworks using deep learning, Ozturk et al. (Ozturk et al., 2020) use very specific deep learning analysis methods.

After reading available corona literature related to the proposed model extensively, the following points can be made.

- Multiple researchers have argued that smart cities have a high chance of fighting any pandemic in the future.
- Machine learning can be used to develop frameworks, architectures, and applications that can help fight any epidemic in their specific area.
- Regression model prediction proved to be wrong as they take a limited number of parameters into account.
- With the continuous and rapid growth in the number of cases, researchers should also focus on the deployment infrastructure of the above-stated machine learning frameworks. Without a suitable computing platform, these platforms will not be able to help completely.

It is also evident from the literature (Ben Hassen et al., 2020; Kallel et al., 2020; Singh & Kaur, 2020; Whaiduzzaman et al., 2020) that many real-time applications in the epidemic require latency-sensitive computing infrastructure. Most of the framework or architectures proposed in the literature provide the knowledge and working of machine learning applications which can be used to control or detect COVID-19. The main issue found in the literature is the non-existence of discussion for a scalable and reliable computing framework where these applications can be hosted. Latency sensitivity is further an issue which these applications would face when deployed. The work done in this paper tries to bridge the gap between machine learning applications and their computing infrastructure for COVID-19.

3 | THE PROPOSED FRAMEWORK

The proposed framework tries to solve the distributed computing environment issues by taking the example of a smart office surveillance system based on the requirements of the COVID-19 pandemic. The proposed system may not solve all the problems, but it provides a starting point for applications of this nature. The proposed framework is shown in Figure 2, with all its components and communications. It contains mainly five components, which are cameras, fog layer, personal computer, cloud computing and a controller Box. Cameras consist of all types of video feed recorders, such as CCTV cameras and webcams of desktops and laptops. These cameras detect, take the feed and pass to the fog layer, personal computer, or cloud computing platforms, where these feeds are analysed for various detections. Fog layer and Personal computers are called Fast Layer (FL) in this paper and these have neural networks with a smaller number of hidden layers so that detection of violations is fast but may have less accuracy. Cloud computing, called Accurate Layer (AL) in this paper, has an abundance of computing resources, so it analyses feed in detail using dense neural network and generates recommendations. The recommendations generated by cloud and fog computing layers are passed to a Controller Box, which decides the next course of action. The proposed framework is developed on the basic requirement that it should be able to integrate multiple modules. The proposed framework can manage various machine learning modules because the basic entity chosen in the proposed framework for resource allocation is a task. The proposed framework's computing infrastructure treat every task as independent with its

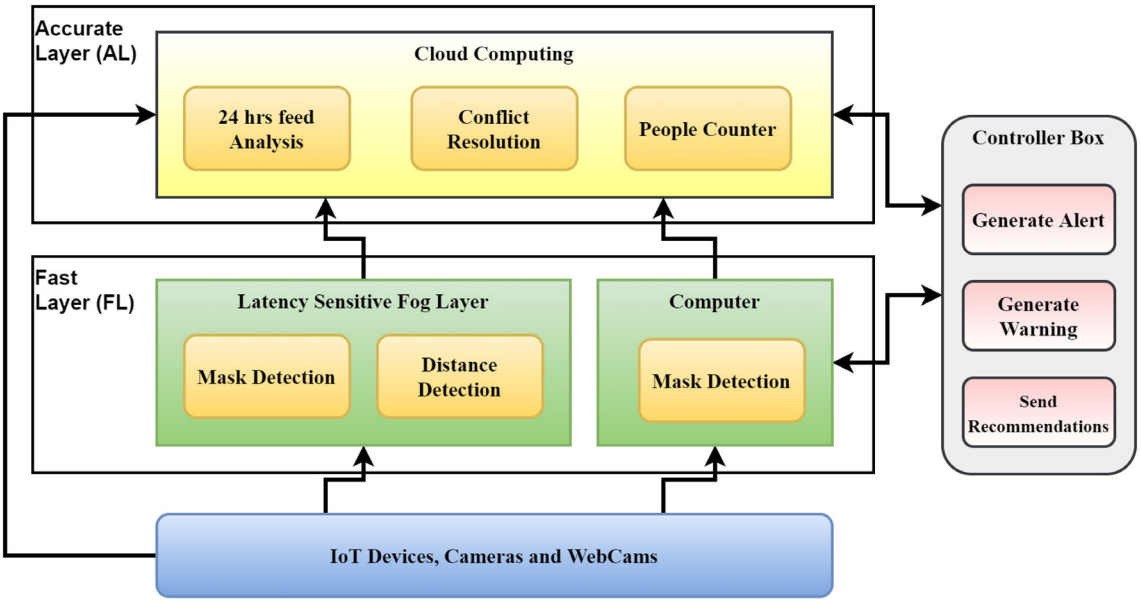

FIGURE 2 The proposed framework

data and QoS requirements encapsulated in it. This makes the proposed framework highly adaptable to achieve different QoS requirements of every task. All these sections are explained in detail ahead.

The main idea behind the selection and naming of these components is their working and relevancy with the prediction process of hosted machine learning applications. Cloud computing itself represents the abundance of computing resources; however, naming it AL further adds meaning to its working with relevant to the proposed framework's neural network. As cloud computing has more and better resources, it should be used for accurate prediction of recommendations by using complex machine learning algorithms. For example, the two use cases described in the proposed framework use DNN, so using the denser and deeper neural network at cloud computing infrastructure can have accurate predictions. Hence, the name Accurate Layer (AL). On the contrary, the fog layer's main focus is to generate predictions with minimum latency, forcing it to use less resource-intensive and faster convergence machine learning models. Based on its time sensitivity to provide the output, it is named as Fast Layer (FL). The controller box is the main scheduler that will be responsible for scheduling tasks generated by different machine learning applications on the proposed framework. It will also act as a communication module that will consolidate the predictions and pass them to the desired agencies. The components division is also done based on the resource allocated to them from different platforms. Fog may be on company computers or network components of the company and cloud will be a third-party service hosted by any cloud service provider. The controller box needs to communicate with all other components, so it should be placed between the user, fog and cloud computing resources keeping latency in mind.

3.1 | Input devices

Every smart system requires an input data stream, which will be analysed and the recommendations will be generated. The camera has been used in the proposed framework to take the feed and submit it to the appropriate layer for analysis. All kinds of cameras are considered, such as CCTV, webcams and inbuilt cams in devices used by the end-user. However, the camera can be replaced by any other IoT device or system. For example, the camera feed is used to count the total number of people entering a building or block. This activity can also be performed by using any infrared sensors also.

3.2 | Fast layer

This layer is deployed on the fog computing infrastructure and the personal computer of the user. If desired computation power can be allocated from a personal computer of the user, then the computation will be done on that system. For example, when the webcam of a computer takes the feed, the proposed framework only uses the computer's computation power to find whether a person sitting in front of the computer is wearing a mask or not. The feed will be communicated to the cloud infrastructure for record-keeping and in-depth analysis. This layer contains both

mask detection features and the distance detection deployed on fog computing for CCTV cameras. Though a personal computer will be used for mask detection of a person sitting in front of the computer, it will not be used for distance detection because of two reasons. First, the proposed framework does not want to use more than the required computation cycle of personal computing to hinder user work. Second, for distance calculation, a side camera is required, which is CCTV and directly connected to the fog layer.

3.2.1 | Mask detection

A CNN is used to find whether a person is wearing a mask or not. Algorithm 1 shows the step followed for detecting the violation of the mask by the user. Also, this algorithm detects whether multiple people are sitting in front of the camera. Then this component sends this information to the controller box, which decides what to do. However, the system gives a warning to the user that he/she is not wearing the mask.

Algorithm 1 : Mask detection algorithm

```
i.   Take frames from the video feed
ii.  For each frame do
iii.     Determine the number of faces in a frame
iv.      If the number of faces is more than 1 then
v.               Show warning
vi.      Else
vii.             Determine the facial landmarks
viii.            Check if the particular landmarks are covered by the mask
ix.              If the mask is present then
x.                       Continue feed
xi.              Else
xii.                     Show warning
xiii.            End if
xiv.     End if
xv.  End for
```

3.2.2 | Distance detection

Similar to the mask detection algorithm described in Algorithm 1, an algorithm has been devised to detect the distance violations by the employees in the company. Algorithm 2 shows the pseudo-code for distance detection. Likewise, this block also sends the violation to the controller box to the mask detection block and let it decide what to do with the violations. Also, this block records the time for which two or more person has done the distance violation.

Algorithm 2 : Distance detection algorithm

```
i.   Show feed to the user and ask him to mark the distance of 6 feet
ii.  Take frames from the video feed
iii. For each frame do
iv.      Determine the number of people in a frame
v.       Calculate the distance between two people
vi.      If the distance is less than 6 feet do
vii.             Increment the social distance count by 1
viii.            Start recording the time for the violation
ix.              Send location and picture of violation to the controller box
x.       End if
xi.  End for
```

3.3 | Accurate layer

The accurate layer is the cloud computing infrastructure, which has machines with high computation power and the flexibility to scale based on the computing environment of the applications. The AL will take the feed of the last hour and analyse it for the total number of violations using a complex neural network with a large number of hidden layers. These violations are matched with the violations generated by the FL. FL neural networks are trained using the errors generated by them. The re-training of the FL neural network is done when the office is shut down and we have no more people in the office to analyse their violations. Algorithm 3 shows the steps followed for the re-training of the FL neural network. All the violations detected by AL are compared with the FL and if there is any difference, FL neural network is re-trained based on these violations. By using this, the FL neural network will increase its accuracy day by day. Within a few days, it will start detecting almost all the violations. AL layer also has a people counter block that counts the total number of people coming in any building block. This component is explained in detail in Section 3.3.1.

Algorithm 3 : Re-training of FL neural network

i. **For** the last 24 h feed **do**
ii. Submit the feed to AL neural network.
iii. X ← all the violations detected by AL.
iv. Y ← all the violations detected by FL.
v. **If** (X ≠ Y) **do**
vi. Find all locations where violations are different.
vii. Generate a dataset of all of these violations.
viii. Re-train the FL neural network with the dataset
ix. Deploy it to all FL components.
x. **End if**
xi. **End for**

3.3.1 | People counter

It is also essential to count the total number of people in a block for each building based on the SOPs issued by the government. Hence, the proposed framework implemented a people counter that counts the total number of people are entering and exiting from buildings and different blocks. This detection is also done by taking the CCTV feed installed at all the doors. Algorithm 4 provides the steps followed by the people counter algorithm, which sends a signal to the controller box when the number of people in any block, floor, or building is more than designated seats. This block will prevent the rush in the building of an organization in the first place.

Algorithm 4 : People counter

i. X ← Total individual a building can accommodate.
ii. Y ← Total individual a floor can accommodate.
iii. Z ← Total individual a block can accommodate.
iv. Analyse the feed from the CCTV camera at the gate of the building
v. **If** X > Total people in the building
vi. Generate alarm at the main gate.
vii. **Else if** Y > Total people on the floor.
viii. Generate alarm at floor gate.
ix. **Else if** Z > Total people in the block.
x. Generate alarm at the block gate.
xi. **End if**

3.4 | Controller box

The controller box is the brain of the proposed framework. It manages all the functions such as deciding when to send the feed to AL and FL, sending warnings, or recording violations to apply penalties to the employees or organizations that do not follow the SOPs. The controller box is connected to every component of the proposed framework. It has multiple algorithms that work collaboratively. All these algorithms are explained in detail in this section. Algorithm 5 provides the steps to be followed by the controller box when it receives an alert for violation from personal computers or CCTV regarding people not wearing the mask or not keeping the distance.

Algorithm 5 : Generate alert

i. **For** all violation alert from a personal computer webcam regarding a person not wearing the mask **do**
ii. Send a warning message to the computer.
iii. Start a counter of 30 s.
iv. **If** the person not wearing the mask, **do**
v. Lock the computer.
vi. Send a message to his boss.
vii. **End if**
viii. **End for**
ix. **For** all violation alert from CCTV for a person not wearing the mask **do**
x. Turn on the siren close to CCTV.
xi. Alert the nearest guard.
xii. **If** the person not wearing the mask, **do**
xiii. Lock the nearest doors.
xiv. Send a message to the floor supervisor.
xv. **End if**
xvi. **End for**
xvii. **For** all violation alert from CCTV for a distance less than 6 feet **do**
xviii. Turn on the siren close to CCTV
xix. Alert the nearest guard.
xx. **If** the distance is not increased **do**
xxi. Lock the nearest doors.
xxii. Send a message to the floor supervisor.
xxiii. **End if**
xxiv. **End for**

Send recommendation block sends different statistics related to violations generated by all the systems. These will help the building administration to alter their strategies so that these violations can be minimized. Often the number of people working at a particular block is more than others, so this component of the proposed framework can identify this kind of imbalance.

3.5 | Retraining the FL

The FL layer does not have enough computing resources that it can re-train its neural network after collecting the new data. FL neural network only focuses on providing latency-sensitive decisions without worrying about the training. So, in the proposed framework, data collected by input devices will be summed up for a given time where this time is large enough. This collected data then passed to AL to predict the violations at a deeper level by using a large number of hidden layers and more complex neural network architectures. The outputs generated by the AL are compared with FL and then weights and bias of the FL network are changed.

4 | EXPERIMENTAL SETUP AND PERFORMANCE ANALYSIS

The proposed framework has been tested using the parameter sweep model of the Aneka cloud application. This section explains the testbed for the experiment and various parameters for different components of the proposed framework.

4.1 | Testbed

Figure 3 shows the testbed used for the evaluation of the proposed framework. The parameter sweep model of the Aneka (Vecchiola et al., 2009) cloud platform has been used to implement the proposed framework. The parameter sweep model is a model in which the executable file remains the same, but data changes are also called the parameters. For example, in the proposed framework, the trained deep learning neural network is the same for all images, predicting whether the user is wearing a mask. However, the input data changes with different images generated from each computer. Due to this fact, the parameter sweep model is used for the implementation of the proposed framework. Aneka provides pre-built support for handling scheduling of parameter sweep model, so Aneka was used to design the algorithms. Aneka uses local resources and Microsoft Azure cloud for the public cloud. Virtual machines were created on local computers using KVM hypervisor. The configuration of these machines is listed in Table 1. These machines act as fog computing devices as these are close to the CCTV and Webcam feed. Moreover, the configuration of these machines is low as compared to a normal personal computer or public cloud. After training of neural networks involved in the proposed framework, different video feeds are input from the VIRAT dataset (Oh et al., 2011) and three live-recorded webcam feeds were provided. All the virtual machines in the testbed have required packages and python 3.7 environments so that when Aneka submits any job to the slave for processing, they can execute them. This is a prerequisite in the Aneka parameter sweep model. The proposed framework uses transfer learning neural architecture for both neural networks. Firstly, for mask detection, MobileNetV2 is used as a backbone network on the collected dataset. Its output layer is removed and applied to a dense layer with 128 neurons and dropout regularization is used with a dropout value of 0.5. Finally, a softmax function is applied to provide the output of mask detection. As there's already very much information available about MobileNetV2 (Sandler et al., 2018) and for the briefness of the paper, it is avoided. The second network used was YOLO-V3 tiny (Adarsh et al., 2020), which is again a very famous network.

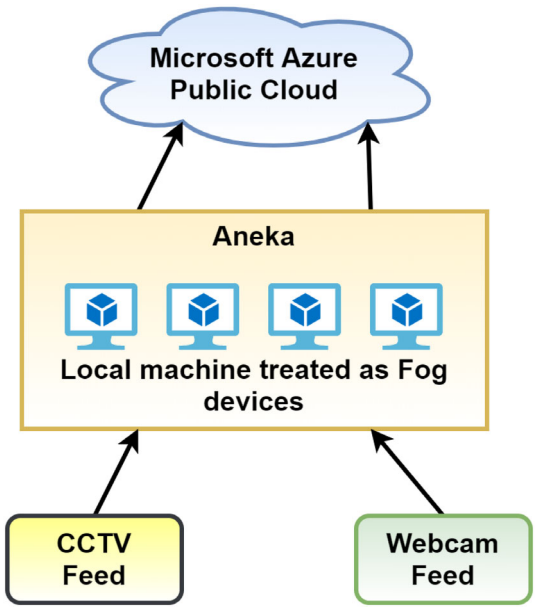

FIGURE 3 Testbed for performance evaluation for proposed framework

TABLE 1 Configuration of physical machines which are treated as fog devices

S. No.	Configuration	Number
1	1 CPU core, 128 MB RAM	3
2	1 CPU core, 256 MB RAM	3
3	2 CPU cores, 512 MB RAM	3

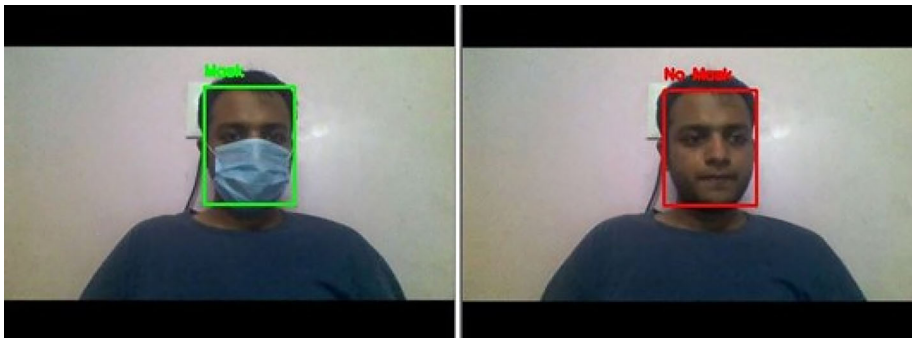

FIGURE 4 Detection of mask by computer webc

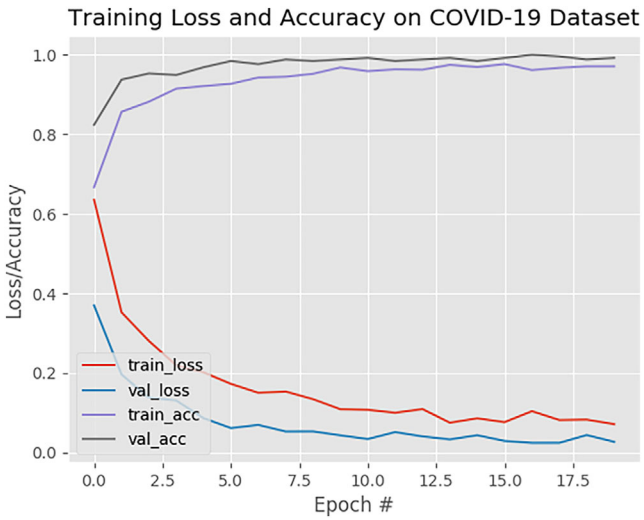

FIGURE 5 Training and testing accuracy and loss of face mask detector

	precision	recall	f1-score	support
with_mask	0.96	1.00	0.98	173
without_mask	1.00	0.95	0.98	171
accuracy			0.98	344
macro avg	0.98	0.98	0.98	344
weighted avg	0.98	0.98	0.98	344

FIGURE 6 Different parameter values for mask detection tool

TABLE 2 Different packages used in the implementation of different components

S. No.	Package	Version
1	Scipy	1.4.1
2	Numpy	1.18.5
3	OpenCV	3.4.5
4	Imutil	0.5.3

4.2 | Mask detection

Mask detection has been implemented using Keras (Chollet, 2015) and Tensorflow (Nelli & Nelli, 2018) framework. Figure 4 shows the output of the component with the mask and no mask video. The Masked Face-Net (Cabani et al., 2020) dataset is used to train the proposed framework package for detecting a masked and unmasked face. Figure 5 shows the loss, accuracy of training and testing of mask detection code and Figure 6 provides values of different parameters calculated by the system. The code for mask detection was written in Python 3.7 using various packages shown in Table 2. Results were successful, where the system was able to detect the face mask with an accuracy of 98%. The wrong detections were false-positive, where even if the user has worn the mask, it was detecting it was not wearing and this is better for the overall precautions. If

TABLE 3 Parameters used in a neural network for mask detection

S. No.	Parameter	RD
1	Type	MobilenetV2 (Sandler et al., 2018)
2	Model	Sequential
3	Number of Input Neuron	the shape of the image is [None,224,224,3]
4	Number of Output Neuron	2
5	Number of Hidden Layers	All layers of Mobilenetv2
6	Activation Function in Hidden Layer	ReLu
7	Output Layer Activation Function	Softmax
8	Optimization	Adam
9	Regularization Technique	Dropout with 0.5 at the second last layer
10	Mini-Batch Size	32
11	Loss Function	Binary Cross-Entropy
12	Metric	Accuracy
13	Return type [State or Sequence]	None

FIGURE 7 Detection of distance by CCTV feed (Oh et al., 2011)

the user flips the webcam upward, it will not affect accuracy as the data is augmented with horizontal and vertical flips, zoom in/out, rotation. An extra feature is also added that if it is a computer webcam and it detects that it is not placed correctly, it will issue a warning. A neural network was trained using an empty chair and front view without and with the user. Table 3 provides the parameters used in the neural network MobileNetV2 (Sandler et al., 2018).

4.3 | Distance detection

Distance detection has been implemented using Keras and Tensorflow framework. VIRAT human activity dataset (Oh et al., 2011) is used to train and test the package of distance calculation and violation detection in the implementation of the proposed framework. Figure 7 shows the output of the component with the distance violation counter. The code for distance detection was written in Python 3.7 using various packages shown in Table 2. Results were successful, where the system was able to detect the distance violations with an accuracy of 98%. None of the parameters were changed from YOLO v3-tiny algorithm, which can be found in (Adarsh et al., 2020).

4.4 | Aneka scheduling

For the performance evaluation of the proposed framework, the python components created for the detection of various violations are used as a parameter sweep model in the Aneka platform. The video for distance violation was chosen from the standard dataset called "VIRAT" (Oh et al., 2011) available on the internet. This dataset contains 250 h of surveillance videos with close to 12.5 h of annotated video data. A total of 8 h of multi-camera video was fetched from this dataset. The fetched video feeds are divided into chunks of 5 min and these are submitted to the fog computing layer for detection of violations. In contrast, a complete video of 8 h has been submitted to the Microsoft Azure cloud for

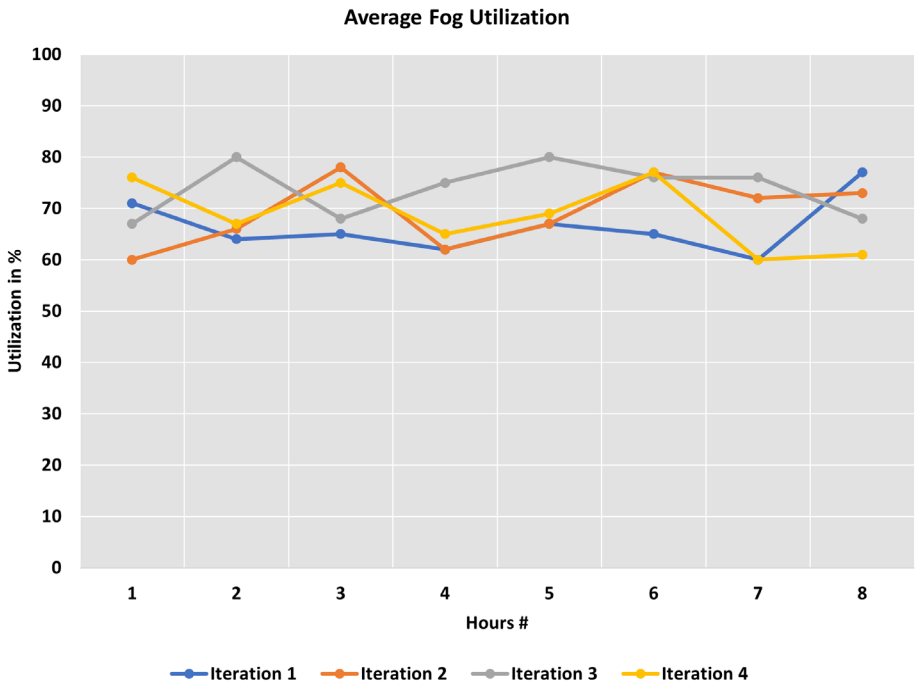

FIGURE 8 Average fog utilization of proposed framework

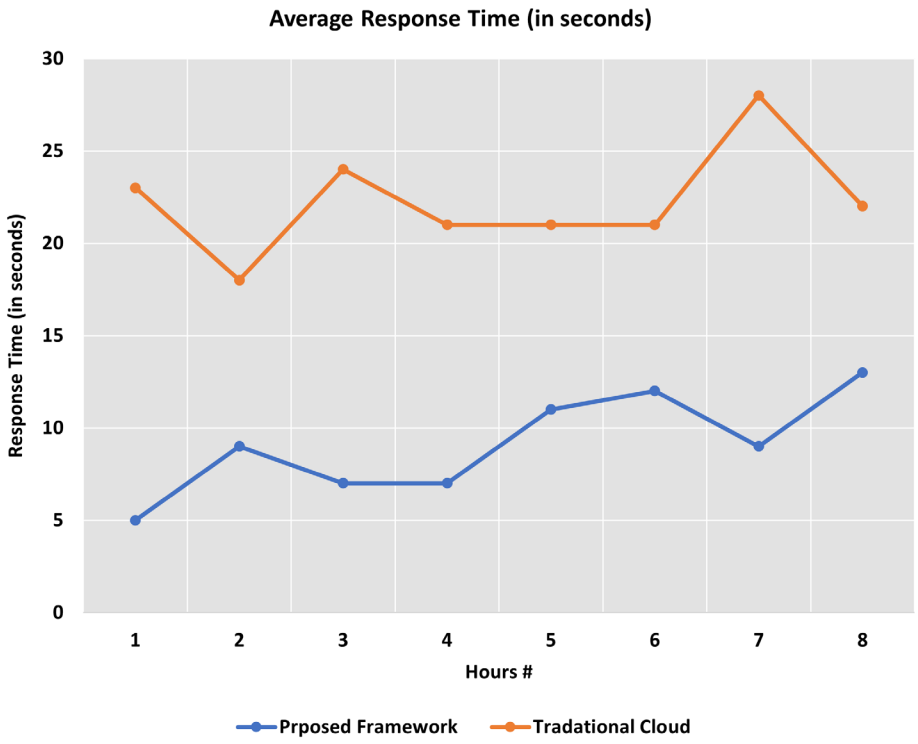

FIGURE 9 Average response time for proposed framework and traditional cloud setup

analysis by a complex neural network. The errors in the detection of violations generated by comparing cloud and fog layer data are used to train the neural networks of the fog layer further. A total of four iterations were done on the same video and we have achieved close to 99% of the same violation detection by both layers. Figure 8 shows the average utilization of fog resources for the detection of violations in the proposed 8 h of videos. The accuracy of the fog layer for each hour is averaged and it is close to 70% for all the iterations. This concluded that the fog computing layer performed consistently for all the iteration. Figure 9 provides the average response time difference by the same neural network for

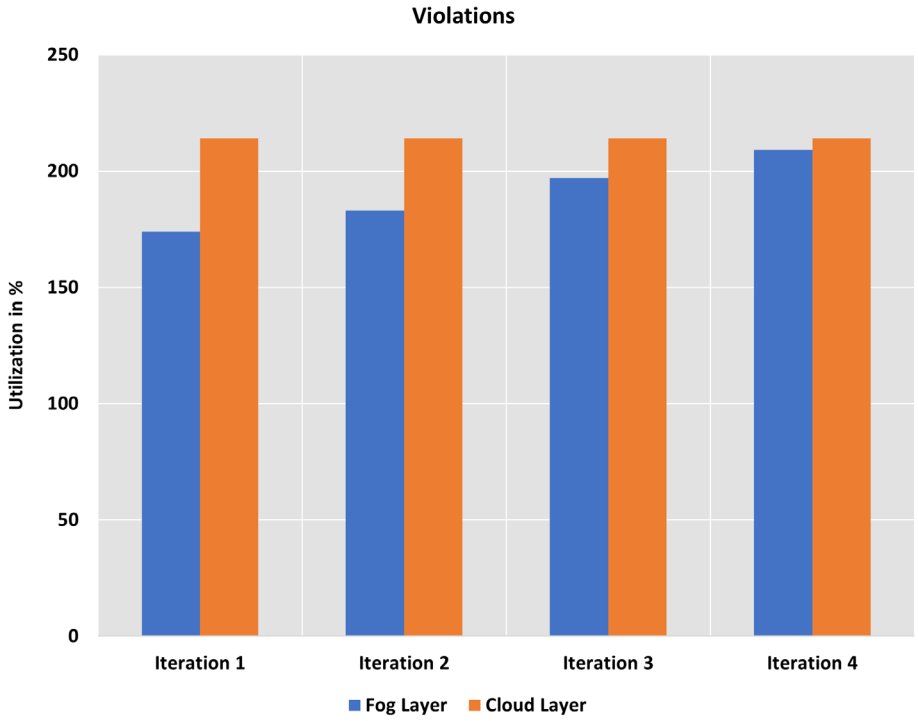

FIGURE 10 The difference in violations generated by fog layer and cloud layer

the fog and cloud layers. Figure 9 showed that fog computing has close to 60% better response time than cloud computing infrastructure. This is because cloud computing video frames need to travel from data generation devices to cloud resources, which takes extra time. Lastly, the graphs in Figure 10 represent the number of violations by the fog layer and the cloud layer, respectively. The initial difference between violations was 41, which was limited to only four after the 4th iteration of the proposed framework.

Cloud and fog computing-based systems are extremely popular. However, the expert systems which can be used by the government and medical agencies to prevent pandemics like COVID-19 have some special set of requirements. After the analysis of results from the proposed framework, some of these features are:

i. A smart pandemic control system should be highly scalable.
ii. The system output should be time-sensitive.
iii. The associated agencies should be able to access this system from anywhere in the world.
iv. It should have a large number of computing resources to compute as well as store the data.
v. Apart from real-time analysis, these systems should have IT resources that can extract rare relationships from the data set, which can be used in the future to control similar pandemics.
vi. This system should be cost-effective, easily deployed and integrated with other modules.

4.5 | Cost of deployment and calibration

In the use case taken for the proposed framework, two cameras are working simultaneously, one for mask detection and one for social distance observations. Mask detection camera is mounted on top of the employees' desk or system and it can be a basic built-in camera for laptops, or an external camera is roughly 5$ and many latest desktops and laptops already have it. However, the second camera is dependent on the area we want to cover, and a basic surveillance camera is good enough for it, which might cost around 10–20$. However, in the current situation, most of the organizations have CCTV and webcams for their systems. So, no extra cost added hardware-wise. For the calibration part, set up is needed to view that it covers the area with a good view and defines a basic distance such as six meters in pixels when installation calibration is required for the camera feeds. For defining the distance program will ask you to mark two points in the view frame.

4.6 | Discussion after results

As observed in Figure 10, the fog system also provides a 10-seconds delay, which is fast compared to real-time systems. Ten seconds is a long delay and should not be observed from local resources like fog nodes. However, as the dataset and resource size were small, it was difficult to observe the benefits of the proposed framework. Thus, a bandwidth limiter was used, limiting the data transfer to the fog and cloud environments to 1 MB/s. The limits used were the same for both cloud and fog environments so that a better comparison can be made. This shows that if the proposed framework is implemented in the real world, it can observe this much delay. Thus, more optimization is still required.

The major advantage of the proposed framework is its adaptability to integrate any machine learning module for COVID surveillance in offices or public places. Section 4.3 of cost and results shows that the proposed framework does not increase hardware cost from the camera perspective. However, organizations that consider implementing need to pay for fog and cloud computing resources. However, it is safe to say that cost of the proposed framework outweighs the benefits provided by the automatic surveillance. Also, fog computing helps to overcome the issue of latency observed by cloud and server-based systems. As the proposed framework uses Aneka's parameter sweep model, it can integrate any machine learning module developed using python or MATLAB very easily and the Aneka software provides APIs for smooth integration. The proposed scheduling algorithm treats every task as a separate identity, which benefits in effectively scheduling them on remote fog nodes. The proposed algorithm increases computation load on Aneka servers, but these servers can be hosted on a better cloud environment with auto-scaling to manage the variable load.

The experimental setup of the proposed framework is tested on two machine learning applications that contain image data. However, the proposed framework is generic for all types of machine learning applications such as voice data, data points, temporal–spatial data. This is due to the inherent property of the proposed framework to consider scheduling task by task irrespective of what type of task it is. Thus, even if the data is of a different format, the scheduling of the proposed framework will manage it equally effectively.

5 | CONCLUSION

Most of the offices are affected by the COVID-19 outbreak. With the non-availability of vaccines, people need to follow the governments' standard operating procedure to stop the spread. Office buildings host many employees, so it has a high probability of spreading coronavirus. In this paper, a framework is proposed, which takes CCTV and webcam feeds and detects people not following the norms. The novelty of the proposed framework is its ability to scale using cloud resources and provide latency-free violation detection using fog computing. The proposed framework has been tested using video of almost 8 h and all the prediction components achieved 98% accuracy, whereas fog provided an average response time under 10 s.

Future work contains the inclusion of more components for the detection of other norms. Additionally, the proposed framework can be generalized using location tracking services, which can track company employee movement after office hours.

ACKNOWLEDGEMENT

This work is partly supported by VC Research (VCR 0000072) for Prof. Chang.

CONFLICTS OF INTERESTS

The authors confirm that there are no conflicts of interest.

DATA AVAILABILITY STATEMENT

Authors do not own the data but COVID-19 data can be publiocly avaialble from who website.

ORCID

Victor Chang https://orcid.org/0000-0002-8012-5852

REFERENCES

Abeler, J., Bäcker, M., Buermeyer, U., & Zillessen, H. (2020). Covid-19 contact tracing and data protection can go together. *JMIR mHealth and uHealth*, e19359. https://doi.org/10.2196/19359

Adarsh, P., Rathi, P., & Kumar, M. (2020). YOLO v3-Tiny: Object detection and recognition using one stage improved model. *6th International Conference on Advanced Computing and Communication Systems, ICACCS 2020*. https://doi.org/10.1109/ICACCS48705.2020.9074315

Advice for the public (2020). https://www.who.int/emergencies/diseases/novel-coronavirus-2019/advice-for-public.

Armbruster, B., & Brandeau, M. L. (2007). Contact tracing to control infectious disease: When enough is enough. *Health Care Management Science*, *10*(4), 341–355. https://doi.org/10.1007/s10729-007-9027-6

Ben Hassen, H., Ayari, N., & Hamdi, B. (2020). A home hospitalization system based on the internet of things, fog computing and cloud computing. *Informatics in Medicine Unlocked.*, *20*, 100368. https://doi.org/10.1016/j.imu.2020.100368

Cabani, A., Hammoudi, K., Benhabiles, H., & Melkemi, M. (2020). MaskedFace-net – A dataset of correctly/incorrectly masked face images in the context of COVID-19. *Smart Health.*, *19*, 100144. https://doi.org/10.1016/j.smhl.2020.100144

Chollet, F. (2015) *Keras: The python deep learning library*, *Keras.Io*.

Coronavirus Disease-19. (2020). Quarantine frame-work for travelers entering Korea. *Osong Public Health and Research Perspectives*, *11*(3), 133–139. https://doi.org/10.24171/j.phrp.2020.11.3.04

COVID-19 Dataset | Kaggle. https://www.kaggle.com/imdevskp/corona-virus-report 2020.

Griebel, L., Prokosch, H.-U., Köpcke, F., Toddenroth, D., Christoph, J., Leb, I., Engel, I., & Sedlmayr, M. (2015). A scoping review of cloud computing in healthcare. *BMC Medical Informatics and Decision Making*, *15*(1), 17. https://doi.org/10.1186/s12911-015-0145-7

Hong, Z., Li, N., Li, D., Li, J., Li, B., Xiong, W., Lu, L., Li, W., & Zhou, D. (2020). Telemedicine during the COVID-19 pandemic: Experiences from Western China. *Journal of Medical Internet Research*, e19577. https://doi.org/10.2196/19577

Ibarra-Vega, D. (2020). Lockdown, one, two, none, or smart. Modeling containing covid-19 infection. A conceptual model. *Science of the Total Environment*, *730*, 138917. https://doi.org/10.1016/j.scitotenv.2020.138917

Kallel, A., Rekik, M., & Khemakhem, M. (2020). IoT-fog-cloud based architecture for smart systems: Prototypes of autism and COVID-19 monitoring systems. *Software: Practice and Experience.*, *51*, 91–116. https://doi.org/10.1002/spe.2924

Koonin, L. M. (2020). Novel coronavirus disease (COVID-19) outbreak: Now is the time to refresh pandemic plans. *Journal of business continuity & emergency planning*.

Lalwani, S., Sahni, G., Mewara, B., & Kumar, R. (2020). Predicting optimal lockdown period with parametric approach using three-phase maturation SIRD model for COVID-19 pandemic. *Chaos, Solitons and Fractals*, *138*, 109939. https://doi.org/10.1016/j.chaos.2020.109939

Mahmud, R., Kotagiri, R., & Buyya, R. (2018). Fog computing: A taxonomy, survey and future directions. *Internet of Everythings*, Internet of Things, (Vol. 41, pp. 103–130). Singapore: Springer. https://doi.org/10.1007/978-981-10-5861-5_5

Malawski, M., Juve, G., Deelman, E., & Nabrzyski, J. (2015). Algorithms for cost-and deadline-constrained provisioning for scientific workflow ensembles in IaaS clouds. *Future Generation Computer Systems*, *48*, 1–18. https://doi.org/10.1016/j.future.2015.01.004

Melin, P., Monica, J. C., Sanchez, D., & Castillo, O. (2020). Analysis of spatial spread relationships of coronavirus (COVID-19) pandemic in the world using self organizing maps. *Chaos, Solitons & Fractals*, *138*, 109917. https://doi.org/10.1016/j.chaos.2020.109917

Mell, P., & Grance, T. (2011). The NIST definition of cloud computing recommendations of the National Institute of Standards and Technology. *Computer Security Division Information Technology Laboratory National Institute of Standards and Technology Gaithersburg*, *28*, 1063–1065. https://doi.org/10.1136/emj.2010.096966

Nelli, F., & Nelli, F. (2018). Deep learning with TensorFlow. *Python Data Analytics*. (Vol. 4, 2, pp. 349–407). Berkeley, CA: Apress. https://doi.org/10.1007/978-1-4842-3913-1_9

Oh, S., Hoogs, A., Perera, A., Cuntoor, N., Chen, C-C., Taek Lee, J., Saurajit Mukherjee J. K., Aggarwal, H. L., Davis, L., Swears, E., Wang, X., Ji, Q., Reddy, K., Shah, M., Vondrick, C., Pirsiavash, H., Ramanan, D., Yuen, J., & Desai, M. (2011). AVSS 2011 demo session: A large-scale benchmark dataset for event recognition in surveillance video. *8th IEEE International Conference on Advanced Video and Signal Based Surveillance, AVSS 2011* (pp. 527-528). https://doi.org/10.1109/AVSS.2011.6027400

Oueis, J., Strinati, E. C., & Barbarossa, S. (2015). The Fog Balancing: Load Distribution for Small Cell Cloud Computing. *2015 IEEE 81st Vehicular Technology Conference (VTC Spring). IEEE*, 1–6. https://doi.org/10.1109/VTCSpring.2015.7146129

Ozturk, T., Talo, M., Yildirim, E. A., Baloglu, U. B., Yildirime, O., & Acharya, U. R. (2020). Automated detection of COVID-19 cases using deep neural networks with X-ray images. *Computers in Biology and Medicine*, *121*. https://doi.org/10.1016/j.compbiomed.2020.103792

Parbat, D., & Chakraborty, O. (2020). A python based support vector regression model for prediction of Covid19 cases in India. *Chaos, Solitons & Fractals*, 109942. https://doi.org/10.1016/j.chaos.2020.109942

Park, Y. J., Cho, S. Y., Lee, J., Lee, I., Park, W.-H., Jeong, S., Kim, S., Lee, S., Kim, J., & Park, O. (2020). Development and utilization of a rapid and accurate epidemic investigation support system for COVID-19. *Osong Public Health and Research Perspectives*, *11*(3), 118–127. https://doi.org/10.24171/j.phrp.2020.11.3.06

Rahman, M. S., Peeri, N. C., Shrestha, N., Zaki, R., Haque, U., & Hamid, S. H. A. (2020). Defending against the novel coronavirus (COVID-19) outbreak: How can the internet of things (IoT) help to save the world? *Health Policy and Technology.*, *9*, 136–138. https://doi.org/10.1016/j.hlpt.2020.04.005

Salgotra, R., Gandomi, M., & Gandomi, A. H. (2020). Time series analysis and forecast of the COVID-19 pandemic in India using genetic programming. *Chaos, Solitons & Fractals*, 109945. https://doi.org/10.1016/j.chaos.2020.109945

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018). MobileNetV2: Inverted residuals and linear bottlenecks. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. https://doi.org/10.1109/CVPR.2018.00474

Schooler, E. M., Zage, D., Sedayao, J., Moustafa, H., Brown, A., & Ambrosin, M. (2017). An Architectural Vision for a Data-Centric IoT: Rethinking Things, Trust and Clouds. *Proceedings - International Conference on Distributed Computing Systems*, 1717–1728. https://doi.org/10.1109/ICDCS.2017.243

Singh, P., & Kaur, R. (2020). An integrated fog and artificial intelligence smart health framework to predict and prevent COVID-19. *Global transitions*.

Sonn, J. W., Kang, M., & Choi, Y. (2020). Smart city technologies for pandemic control without lockdown. *International Journal of Urban Sciences*, 1–3. https://doi.org/10.1080/12265934.2020.1764207

Sujath, R., Chatterjee, J. M., & Hassanien, A. E. (2020). A machine learning forecasting model for COVID-19 pandemic in India. *Stochastic Environmental Research and Risk Assessment, pp.*, *34*, 1–14. https://doi.org/10.1007/s00477-020-01827-8

Sun, F., Ganguli, A., Nguyen, J., Brisbin, R., Shanmugam, K., Hirschberg, D. L., Wheeler, M. B., Bashir, R., Nash, D. M., & Cunningham, B. T. (2020). Smartphone-based multiplex 30-minute nucleic acid test of live virus from nasal swab extract. *Lab on a Chip*, *20*(9), 1621–1627. https://doi.org/10.1039/d0lc00304b

Tuli, S., Tuli, S., Tuli, R., & Gill, S. S. (2020). Predicting the growth and trend of COVID-19 pandemic using machine learning and cloud computing. *Internet of Things*, *11*, 100222. https://doi.org/10.1016/j.iot.2020.100222

Vecchiola, C., Chu, X., & Buyya, R. (2009). Aneka: A software platform for NET based cloud compputing. *Advances in Parallel Computing*, 267–295. https://doi.org/10.3233/978-1-60750-073-5-267

Verma, R., & Chandra, S. (2020). A systematic survey on fog steered IoT: Architecture, prevalent threats and trust models. *International Journal of Wireless Information Networks.*, *28*, 116–133. https://doi.org/10.1007/s10776-020-00499-z

Wei, Y., Sukumar, K., Vecchiola, C., Karunamoorthy, D., & Buyya, R. (2017). Aneka cloud application platform and its integration with windows azure. *Cloud Computing: Methodology, Systems, and Applications*, (Vol. 1, pp. 645–679). https://doi.org/10.1201/b11149

Whaiduzzaman, M., Hossain, R., Shovon, A. R., Roy, S., Laszka, A., Buyya, R., & Barros, A. (2020). A privacy-preserving Mobile and fog computing framework to trace and prevent COVID-19 community transmission. *arXiv*. https://doi.org/10.1109/jbhi.2020.3026060

Won Sonn, J., & Lee, J. K. (2020). The smart city as time-space cartographer in COVID-19 control: The south Korean strategy and democratic control of surveillance technology. *Eurasian Geography and Economics*, *00*(00), 1–11. https://doi.org/10.1080/15387216.2020.1768423

Yasaka, T. M., Lehrich, B. M., & Sahyouni, R. (2020). Peer-to-peer contact tracing: Development of a privacy-preserving smartphone app. *JMIR mHealth and uHealth*, *8*(4), e18936. https://doi.org/10.2196/18936

Yigitcanlar, T., Butler, L., Windle, E., Desouza, K. C., Mehmood, R., & Corchado, J. M. (2020). Can building "artificially intelligent cities" safeguard humanity from natural disasters, pandemics, and other catastrophes? An urban scholar's perspective. *Sensors (Switzerland)*, *20*(10), 2988. https://doi.org/10.3390/s20102988

Zhang, Y., Yu, X., Sun, H. G., Tick, G. R., Wei, W., & Jin, B. (2020). Applicability of time fractional derivative models for simulating the dynamics and mitigation scenarios of COVID-19. *Chaos, Solitons & Fractals*, 109959. https://doi.org/10.1016/j.chaos.2020.109959

AUTHOR BIOGRAPHIES

Ajay Singh, Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Solan, India.

Vaibhav Jindal, Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Solan, India.

Rajinder Sandhu, Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Solan, India.

Victor Chang, Artificial Intelligence and Information Systems Research Group, School Computing, Engineering and Digital Technologies, Teesside University, Middlesbrough, UK.

How to cite this article: Singh A, Jindal V, Sandhu R, Chang V. A scalable framework for smart COVID surveillance in the workplace using Deep Neural Networks and cloud computing. *Expert Systems*. 2022;39:e12704. https://doi.org/10.1111/exsy.12704