New Generation Computing (2021) 39:541–568
https://doi.org/10.1007/s00354-021-00135-1

NEW GENERATION COMPUTING

CASE-CF: Context Aware Smart Epidemic Control Framework

Harsuminder Kaur Gill[1] · Vivek Kumar Sehgal[1] · Anil Kumar Verma[2]

Received: 1 February 2021 / Accepted: 26 August 2021 / Published online: 4 September 2021

Abstract
Novel Coronavirus (COVID-19) has become one of the deadliest pandemics that has affected almost all the nations in the world. Lockdown and systematic re-opening of shopping malls, offices, etc. is still one of the major weapons against this virus. However, the government and medical agencies take long time to reopen the places due to risks involved in this deadly virus. The delay to reopen places has resulted in sharp decline in the growth of economy. In this paper a current context aware framework is proposed which uses multiple inputs for a specific region to decide whether to open it or not. The proposed framework used series of deep neural network models to generate recommendations specific to a particular region. Most of the inputs are real-time and readily available with the government. The main aim is to develop framework which can be used in any kind of pandemic even in small region to easily contain it. However, it has been tested using opensource data available for COVID-19. Data was crawled from web for 22 districts of Haryana state of India. Experimental result proved the efficiency of proposed framework.

Keywords Epidemic · COVID-19 · Neural network · Context aware · LSTM · RNN

Introduction

Novel Coronavirus (COVID-19) is one of the worst hit epidemics of twenty-first century, effecting millions of people worldwide. COVID-19 is a contact transmissible disease which can be easily transmitted from one person to another [1]. The

Communicated by Y. Giga.

✉ Harsuminder Kaur Gill
harsuminder@gmail.com

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

[2] Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology, Patiala, Punjab, India

prime reason for the spread of COVID-19 is the absence of symptoms during initial phase of infection, hence, people spreading the disease are themselves unaware about it [1]. The only successful method to stop the spread of COVID-19 is the breakdown of the chain of spread among people which can done by imposing lockdown in various parts of the country [2]. COVID-19 has halted many businesses across the world and has caused financial loss in billions. Apart from COVID-19 many infectious outbreaks over the last decade has taken countless lives. WHO compiles a list of outbreaks over the world every year which has many new and repeated outbreaks [3]. There were more than fifty infectious disease outbreaks such as Ebola [4], Middle East respiratory syndrome [5] coronavirus, Yellow fever [6] etc. in year 2018 and 2019 itself. Many outbreaks such as Ebola, Cholera etc. occurs more than once in a year at different locations over the globe. However, COVID-19 has proved to be worst among all for business as every country imposed strict lockdown. However, the lockdown practice cannot be exercised for a long period and many countries have started providing relaxation in lockdown strategy. Relaxations in the lockdown needs to be implemented very carefully as unsystematic reopening of places may lead to another wave of infection [7]. Therefore, countries require a framework which can predict lockdown areas efficiently so that the virus can be contained. With the increase in use of sensors to monitor different parameters and advancement in machine learning algorithms, a framework should be developed which can predict areas that should be in the lockdown.

Smart transport [8], smart healthcare [9], smart education [10] are examples of the smart systems that can aid to the development of any smart city. Similarly, Smart Epidemic Control (SEC) can be developed for COVID-19 that utilizes multiple sensor data from different regional areas and provide the prediction as well as prevention of COVID-19 outbreak. Prediction generated by SEC can be of multiple types such as spread prediction, regional prediction, death prediction etc. For every prediction, many recommendations can be suggested by the SEC which if considered can reduce the effect of epidemic. Recommendations can be general while controlling COVID-19, for example, sending alert to the government authorities, warning the public, releasing general precaution messages in different social networking forums, alerting the nearest hospitals, controlling the traffic in the affected areas, etc. SEC should perform general activities immediately after receiving information of any epidemic outbreak. However, SEC should automatically adjust with the current context of the COVID-19 spread to control it efficiently. Therefore, a context aware recommendation generation engine is required for controlling epidemics such as COVID-19 by marking sensitive areas and generating recommendations for the lockdown process. Context aware recommendation engine in SEC will also consider different context of sensors, users, and Government agencies while generating any prediction or recommendation. Number of instances and attributes for SEC and stream data collected from sensors would be huge for normal processing. SEC needs an efficient and effective machine learning technique for appropriate recommendation generations. Context can be anything which has the capability to change the outcome of the system. In case of SECs, context can be infection density, population density, number of hospitals, and hospital facilities. However, the most important context is the geographical location. For example, in case of COVID, a city may be

marked green with 500 infectious if it has 1000 ICU beds and 10,000 normal beds. On the contrary, a city is in red zone with only 100 infections if it has 5 ICU beds and 100 normal beds. In smart SEC (such as COVID situation), on rules fit all is not a good model, even with the machine learning algorithms. Therefore, discussion of context is important and should be used to the possible extent.

Deep Neural Networks (DNNs) [11] are multi-layered neural network with large number of intermediate layers and neurons, complex connection patterns, and automatic feature extraction. Large number of layers, neurons and connections requires colossal computing power to train the network, but these neural nets are more capable of extracting features from data. Based on the efficiency of DNNs, they are suitable for SEC installed in smart cities with functions to extract the context of different entities timely and accurately. DNNs have been used to develop many intelligent systems involving complex image processing, speech recognition, and data processing. Success of DNN in these areas has motivated its use in development of SEC which involves colossal number of sensors and users. However, there are many flavors of DNNs available in the market, where each DNN suits some specific kind of problem.

The main objective of the proposed framework is to collect values of various attributes such as number of people infected in an area, total population, total people moving in an area, number of people quarantined, and number of hospitals in an area, etc. and predict whether lockdown should be implemented in an area or not. The proposed framework will also include the sensitive rating of nearby clusters while making the decision to apply the lockdown or not. Furthermore, clusters are proposed to be in hexagonal shape with varying radius. This is done because of two reasons (a) hexagonal structures leave minimal space in between and are good way to analyze all the directions of spread, (b) varying radius provides better control in less or more hit COVID-19 areas. The proposed framework uses multiple types of DNN for the recommendation generations. In addition, all the values are proposed to be calculated in real time so that the proposed framework can take decision based on current context values. The proposed framework has been designed using OpenCV [12, 13] and TensorFlow [14, 15] framework and has been tested for 22 districts of Haryana. The major contribution of the proposed framework are as follows:

- Highlighted the role of contextual information in predicting and preventing the infectious disease outbreaks.
- Listed the basic data required to predict the infection rate in a geographical area and lockdown requirements.
- Designed a series of different framework of DNNs which contributes collectively to make final decision.
- Using of appropriate DNN framework based on the type of contextual information collected.
- Created data set for the evaluation of proposed framework using data from multiple sources.

The rest of the paper is structured as follows. "Related Work" provides the relevant literature of COVID-19 and smart systems. "Context Aware Smart Epidemic

Control Framework (CASE-CF)" describes the proposed framework in detail. Before concluding in "Experiment Evaluation and Performance Analysis" and "Conclusion" provides experimental evaluation of the proposed framework using combination of real and synthetic data for 22 districts of Haryana.

Related Work

The publications in the literature relevant to COVID-19 is fast and has increased in number during epidemic hit across various counties around the globe. In this section, a summary of applicable literature is discussed.

Various countries have actively contributed towards various activities that would reduce the spread of COVID-19. Quarantine management team in South Korea [16] developed a smart quarantine system which includes multiple stakeholders of government and health sectors. They argued that it was successful in controlling the epidemic from overseas citizens. Park et al. [17] discovered a smart epidemic investigation support system that checks patients and provides investigation online. Sonn and Lee [18] discussed the role of smart cities in controlling and flatting the curve of COVID-19 infections in South Korea. They also discussed how intelligent cities can help to prevent epidemics in the future. Yigitcanlar et al. [19] provided a viewpoint that if all the cities in the world would be smart, how it would have impacted the spread of COVID-19. They also argued that the government should focus on making their cities smart to fight any natural or man-made catastrophes in the future. Sonn et al. [20] listed the role of smart cities in the epidemic and how they can help to relax the lockdowns. Hong et al. [21] discussed the role of telemedicine in the epidemic of COVID. Sun et al. [22] developed a smartphone app which can conduct test of COVID-19 based on symptoms. Rahman et al. [23] provided the role of Internet of Things in controlling the spread of COVID-19. Podpora et al. [24] created a speech based human interaction smart system that can control the spread of COVID-19. Ibarra-vega [7] stated the role of smart system modeling in the various phases of a lockdown. She argued that smart systems which have multiple inputs can better judge the intensity of lockdown to be carried out. Kummitha [25] discussed the role of various smart technologies present in the market to curb the spread of COVID-19. Furthermore, there are few papers on contact tracing [26–28].

Multiple papers have been published in recent times which uses machine learning techniques to predict, prevent or assess COVID-19. In 2020, Sujath et al. [29] proposed a forecasting COVID-19 model based on linear regression, multilayer perceptron and vector autoregression. They used data available on Kaggle website. Similarly, Zhang et al. [30] used time fractional derivative models for predicting the dynamics and different scenarios of COVID-19. Their model first fit the time data and based on this; it predicts the future. They predicted peak date for many countries even before those countries reached at their peak. Likewise, Parbat and Chakraborty [31] predicted the number of COVID-19 cases in India based on a support vector regression model using python as a framework. However, their prediction was wrong, because COVID-19 cases did not decline as they predicted. It can be concluded from this paper that COVID-19 prediction cannot be conducted using simple

regression models. A high sophisticated model trained on multiple parameters is required to make accurate predictions. Similar to this, Zhang et al. [32] predicted number of cases for six western countries which are Italy, USA, Canada, France, Germany and the UK. Tuli et al. [33] used machine learning and cloud computing in predicting the growth of COVID all over the world. Melin et al. [34] conducted spatial analysis of COVID cases using self-organized maps. They highlighted the area as well as the direction of spread using various attributes. Lalwani et al. predicted the appropriate lockdown periods in China using the SIRD model. The model predicted the appropriate period of lockdown in China as 73 which was close to actual period of 77 days. Salgotra et al. [35] used genetic programming to forecast and conduct time series analysis on COVID data collected for India. Ozturk et al. [36] developed a deep learning framework that takes CT scans as input and predicted whether the patient is COVID-19 infected.

After reading extensive and latest related work on machine learning methods for COVID detection following points were identified.

- Researchers have voted for the use of smart technology to control any epidemic such as COVID-19.
- Researchers have successfully used deep learning models in various spheres for the detection of COVID-19.
- Also, multiple researchers have devised conceptual models which can help the governments to decide area and level of lockdown to conduct.
- To the best of our knowledge, apart from few conceptual models, there was no model or framework present in the literature similar to the proposed framework that takes current context inputs from various sources.

Context Aware Smart Epidemic Control Framework (CASE-CF)

To control the epidemic in an area, a CASE-CF has been proposed, as shown in Fig. 1. The input to this system is provided by previously installed sensors in the smart city. The CASE-CF converts geographical area into clusters of hexagons those are attached to each other in such a way that leaves no space in between two hexagons. In highly infected areas these hexagonal structures can have radius as low as 100 m and it can be extended to few kilometers for less infected areas. The basic structure of hexagonal cell is shown in Fig. 2. The CASE-CF uses various models of deep learning for feature extraction and prediction, as shown in Fig. 1. The CASE-CF has three stages with output of previous stage as input to next stage.

The flow of data is as follows; First, camera feeds are used from every entrance to the hexagon to detect the number of people moving in and out of a hexagon. The Convolutional Neural Network (CNN) is proposed to use along with openCV for detection of people moving in and out of the hexagonal cell. In addition, a direct data entry option can be provided to the authorities to record number of people entering in the area if there are no cameras available. Second, a back propagation neural network is trained that takes input as total population in the area, number of infected people, and number of quarantine people and outputs

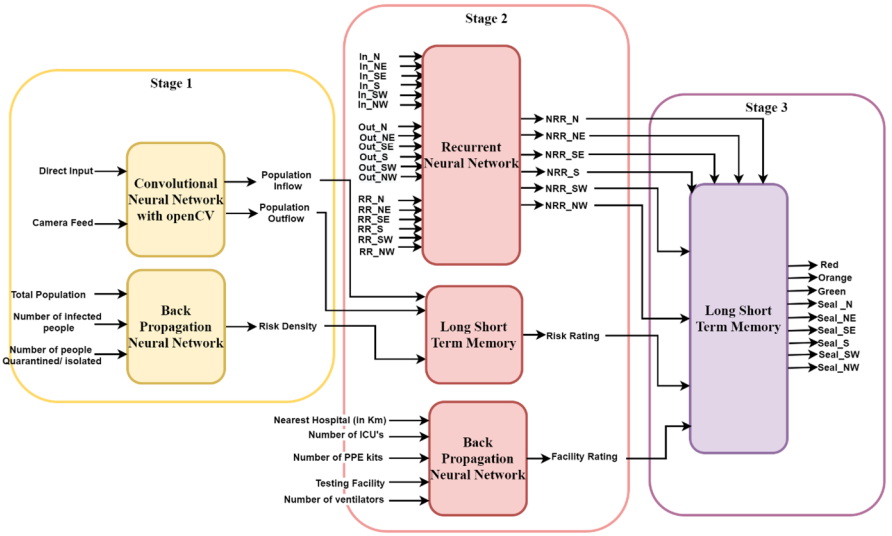
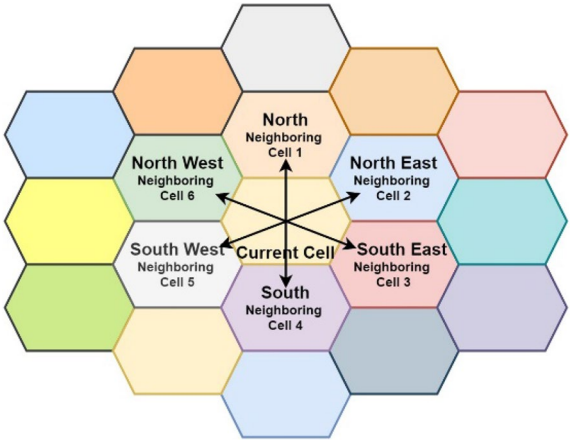


Fig. 1 CASE-CF

Fig. 2 Area division



a number called Risk Density (RD). At the second stage, a Recurrent Neural Network (RNN) takes input from CNN about inflow and outflow from all the six neighbors of a hexagon and outputs a number for all the neighbors known as Neighbor Risk Rating (NRR). A Long Short-Term Memory (LSTM) network takes total inflow and outflow numbers as input along RD from first stage to predict a value called Risk Rating (RR). At the same level, another back propagation neural network takes input such as distance of nearest COVID hospital, number of ICU beds, number of PPE kits, testing facilities, ventilators, and outputs a Facility Rating (FR). FR provides the capability of the hexagon cell to effectively deal with the outbreak if occurs. At the last stage, a LSTM network takes input

of NRR, RR, and FR from second stage and predicts the area sensitivity and recommends which border should completely block the ingoing and outgoing of the traffic for better prevention from the COVID-19 outbreak. Complete neural network is represented in Fig. 1 along with three stages. Working of all the neural networks is explained in detail in subsequent sections.

CNN for Flow Detection

The governments have used different tactics in dealing with areas that have heavy inflow and outflow of people throughout the day. Areas having higher inflow or outflow rate are more prone to COVID-19 infection spread and hence traffic should be controlled. In the CASE-CF, CNN model along with OpenCV is used to automatically detect flow of people from one area to another using CCTV cameras. Algorithm listed in [37] is used to devise a method to detect whether person is moving in or out of the hexagon. Algorithm 1 explains major steps used to perform the action as provided in [37]. However, if cameras are not available in an area, manual entry of data is also provided to keep the check of traffic flow. Table 1 explains the inputs and outputs for the CNN component.

A few important points to keep in mind is that there are many challenges with the usage of camera to count the inflow and outflow such as occlusion, clutter, non-uniform object distribution. Furthermore, type of CCTV camera, range and location of installation is also a topic of concern and should be studied. However, the proposed framework focused on analysis of collected context data using different frameworks of DNNs and not on collection of data. For the proposed framework, assumptions were made that all camera and system for detection of inflow/outflow would wok fine.

Table 1 Input and output of CNN block

Neural Network	Input	Output	Notation of output
CNN with openCV	Camera feeds	Inflow from North	In_N
	Direct Input	Inflow from North West	In_NW
		Inflow from South West	In_SW
		Inflow from South	In_S
		Inflow from South East	In_SE
		Inflow from North East	In_NE
		Outflow from North	Out_N
		Outflow from North West	Out_NW
		Outflow from South West	Out_SW
		Outflow from South	Out_S
		Outflow from South East	Out_SE
		Outflow from North East	Out_NE
		Total inflow from all direction	
		Total outflow from all direction	

Algorithm 1: Count people moving in and out of a hexagon.

```
Create a horizontal line where the road starts.
for Each person in the feed generate a centroid point do
    Assign an ID number to that point.
    for Every ID point track the movement do
        Calculate the Euclidean distance of centroid in previous and current frame.
        if Distance is Small
            Treat them moving and assign new point the previous ID and delete previous centroid.
        Elseif Distance is more than threshold
            Treat it as new point and allocate an ID
        else
            Treat point left the frames and delete it.
    Check the difference, (New Y axis – Y Axis Horizontal Line)
    If Difference is negative
        Person is moving into the hexagon (Inflow)
    else
        Person is going out of hexagon (Outflow)
Continue this algorithm till feed lasts.
```

Back Propagation Neural Network for Risk Density Calculation

This block uses a back propagation neural network that takes three inputs which are total population of a hexagonal cell, number of people infected, number of people currently quarantined. Based upon these three factors, the risk of a particular cell is predicted as risk density, as shown in Fig. 3. The proposed neural network has one input layer, one output layer and two hidden layers of four neuron each. The proposed network is formed after rigorous experiment on the provided data. Details of experiment are given in "Experiment Evaluation and Performance Analysis".

RNN based Neighbor Risk Rating

To control any epidemic, it is extremely important to identify the direction in which the disease is spreading. To detect the direction of flow, neighbor risk rating is calculated using RNN. Table 2 shows the inputs and outputs of the proposed neural network. RNN is widely used when values on successive time stamps are closely

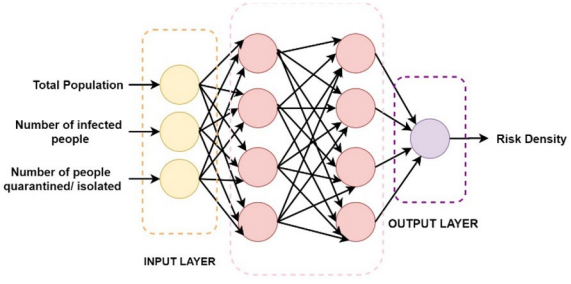


Fig. 3 Back propagation neural network for calculation of risk density

Table 2 Input and output to the RNN model

Neural Network	Input	Notation of input	Output	Notation of output
Recurrent Neural Network	Inflow from North	In_N	Neighbor Risk Rating of North border	NRR_N
	Inflow from North East	In_NE	Neighbor Risk Rating of North East border	NRR_NE
	Inflow from South East	In_SE	Neighbor Risk Rating of South East border	NRR_SE
	Inflow from South	In_S	Neighbor Risk Rating of South border	RR_S
	Inflow from South West	In_SW	Neighbor Risk Rating of South West border	NRR_SW
	Inflow from North West	In_NW	Neighbor Risk Rating of North West border	NRR_NW
	Outflow from North	Out_N		
	Outflow from North East	Out_NE		
	Outflow from South East	Out_SE		
	Outflow from South	Out_S		
	Outflow from South West	Out_SW		
	Outflow from North West	Out_NW		
	Risk Rating of North border	RR_N		
	Risk Rating of North East border	RR_NE		
	Risk Rating of South East border	RR_SE		
	Risk Rating of South border	RR_S		
	Risk Rating of South West border	RR_SW		
	Risk Rating of North West border	RR_NW		

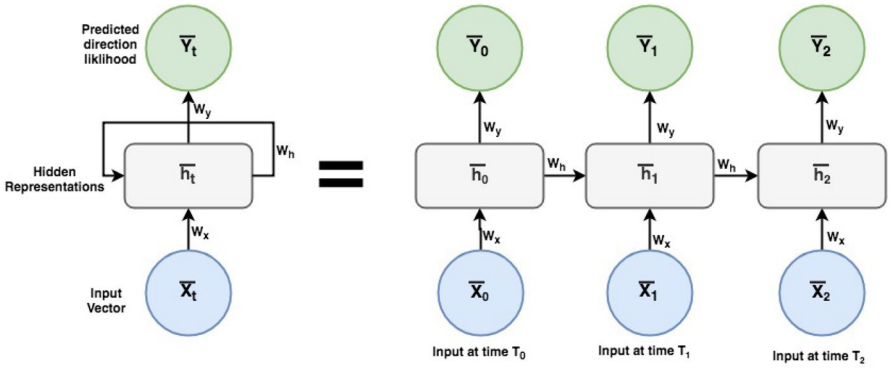


Fig. 4 RNN structure to detect direction of flow of disease

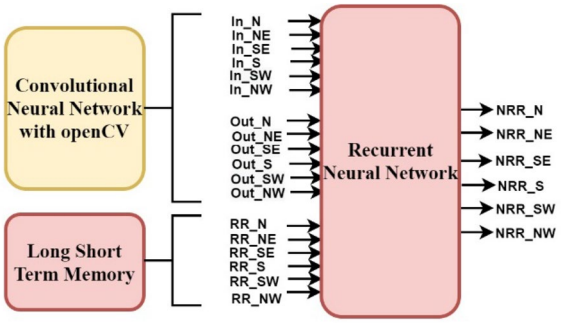


Fig. 5 Recurrent neural network to calculate neighbor risk rating from each direction

related to each other. RNN maintains the relationship between successive time stamps using previous output values as an input along with the current input values. This component takes input as flow of population from each border of hexagonal cell and risk density from each direction from CNN and LSTM, respectively, explained in "CNN for Flow Detection" and 3.4. Based upon these inputs, risk rating of each neighbor is calculated as NRR, as shown in Fig. 4. All the calculations and outputs are performed for all six neighboring boundaries to calculate the risk rating from each direction. Figure 4 shows the structure of RNN to predict the neighbor risk rating. $\overline{X_t}$ is set of inputs at time, t. Hidden state at time t, is computed using formulae,

$$h_t = g_h\left(w_h x_t + w_h h_{t-1} + b_h\right), \tag{1}$$

$\overline{Y_t}$ gives output or predicted direction of flow. It is computed using,

$$y_t = g_y\left(w_y h_t + b_y\right), \tag{2}$$

where w_h and w_y are weight matrix, b_h and b_y are bias and g_h and g_y are activation functions (Fig. 5).

At time t_0, input is x_0

Therefore, h_0 and y_0 are computed using Eqs. (1) and (2):

$$h_0 = g_h\left(w_x x_0 + w_h h_{0-1} + b_h\right), \tag{3}$$

$$h_0 = g_h\left(w_x x_0 + b_h\right), \tag{4}$$

$$y_0 = g_y\left(w_y h_0 + b_y\right). \tag{5}$$

At time $t = 1$, input is,x_1, h_1 and y_1 are computed using Eqs. (1) and (2)

$$h_1 = g_h\left(w_x x_1 + w_h h_{1-1} + b_h\right), \tag{6}$$

$$h_1 = g_h\left(w_x x_1 + w_h h_0 + b_h\right), \tag{7}$$

$$y_1 = g_y\left(w_y h_1 + b_y\right). \tag{8}$$

At time $t = 2$, input is, x_2, h_2 and y_2 are computed using Eqs. (1) and (2)

$$h_2 = g_h\left(w_x x_2 + w_h h_{2-1} + b_h\right), \tag{9}$$

$$h_2 = g_h\left(w_x x_2 + w_h h_1 + b_h\right), \tag{10}$$

$$y_2 = g_y\left(w_y h_2 + b_y\right). \tag{11}$$

LSTM based Risk Rating

This component takes input as total inflow, outflow of the population and risk density calculated by neural networks explained in "CNN for Flow Detection" and "Back Propagation Neural Network for Risk Density Calculation". To control a certain disease, it is important to investigate that how much a certain disease has already affected the community and how much it can spread further. This component provides a value known as Risk Rating of a hexagonal cell on a scale of 1–10 to represent the prediction of spread of COVID cases in the hexagonal cell, as shown in Fig. 6.

This component has one basic unit called Long Short-Term Memory (LSTM) unit. The main reason of using LSTM in this component is that the previous inputs need to be considered while calculating Risk Rating. For example, at one-point total inflow may be less but it does not imply that the Risk Rating is low. In fact, it should take all previous calculations into consideration before making calculation of risk rating. To perform computation using previous values, the previous rating values should be stored in some memory. LSTM has memory cells which enable it to forget and save some information over a longer period of sequences. LSTM is a special type of Recurrent Neural Network (RNN)

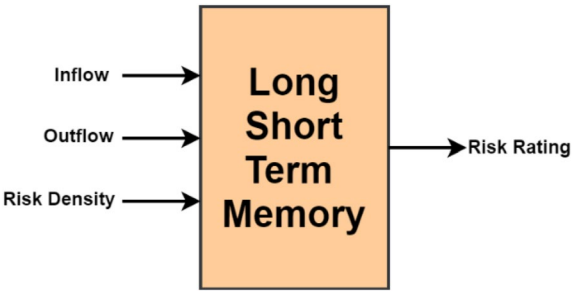


Fig. 6 Component to predict the risk rating

which is used for prediction and classification of temporal sequences. The issues with RNN are vanishing gradient and exploding gradient and these are handled in LSTM networks; therefore, they can remember short sequences for a longer period. Every occurrence of risk rating in a specific geographical cell is cached into LSTM network. A LSTM network architecture is a set of recurrently connected subnets or memory blocks. LSTM block, shown in Fig. 7, consists of self-connected memory cells and three multiplicative units; input gate, output gate and forget gate. These gates regulate the flow of data by learning which information is crucial to keep or discard away. The Algorithm 2 is used to predict the risk rating using LSTM in CASE-CF.

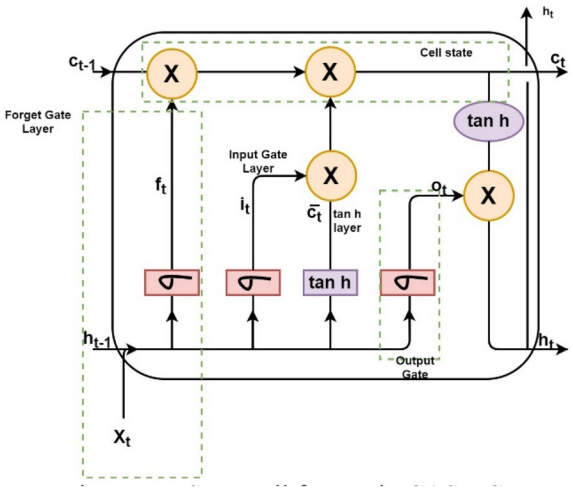


Fig. 7 LSTM cell for RR in CASE-CF

Algorithm 2: LSTM for Predicting Risk Rating of a Hexagonal

1: The initial step is to identify which information is not important and will be thrown away from the cell state. This decision is made by the forget gate layer.

$$f_t = \sigma\big(w_f[h_{t-1}, x_t] + b_f\big), \quad (12)$$

where w_f is the weight of forget gate layer,h_{t-1} is the output from previous time stamp, $x_t$is the new input and b_f is bias.

2: The second step is to identify the important information and to store it in a cell state. This process is completed by sigmoid layer and tanh layer. Sigmoid layer/ input gate layer decides which values needs to be updated. Tanh layer creates a vector of new candidate values, that could be added to the state.

$$i_t = \sigma(w_i[h_{t-1}, x_t] + b_i) \quad (13)$$

$$\tilde{c_t} = \tanh(w_c[h_{t-1}, x_t] + b_c) \quad (14)$$

Where $w_i$is the weight of input layer, x_t is the new input, b_i is bias, $c_t$is the cell state, w_c is weight of cell state, b_c is bias.

3: Updates the old cell state $c_{t-1}$to new cell state c_t

$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t \quad (15)$$

4: The last step decides the output $o_t$based on the cell state. The cell state is then pushed to an activation function, tanh to squash the output values between -1 and 1. The final output value h_t identifies the category of the infectious disease.

$$o_t = \sigma\,(w_o[h_{t-1}, x_t] + b_o) \text{ and } h_t = o_t * \tanh(c_t) \quad (16)$$

Facility Rating

Facilities available in an area also contributes to the decision that whether to impose lockdown in an area or not. If a hexagonal area has sufficient hospitals and beds to facilitate the infected patients, then relaxation in lockdown can be provided. The prime reason to enable lockdown is the unavailability of medical resources due to overcrowded hospitals. However, if an area has abundant medical resources, authorities should not be worried. Therefore, the CASE-CF considers the facilities available while making the final decision of whether to impose lockdown or not in a region. For calculation of facility rating, the CASE-CF uses back propagation network such as used in Sect. 3.2 and shown in Fig. 8. It has five input neurons which are distance of nearest COVID hospital, number of ICU's bed, Number of PPE kits available, availability of testing facility, and number of ventilators. It has two hidden layers with six neurons each and one output layer with one neuron. After analyzing all the facilities in an area, rating is predicted by neural network known as facility rating. Higher the rating, more is number of facilities available and vice versa. This rating would be used in next stage to mark the sensitivity of an area.

LSTM to Mark Sensitivity of an Area

The last stage of the CASE-CF is a LSTM network for final recommendations. All the previous outputs by various neural networks, act as an input to this network.

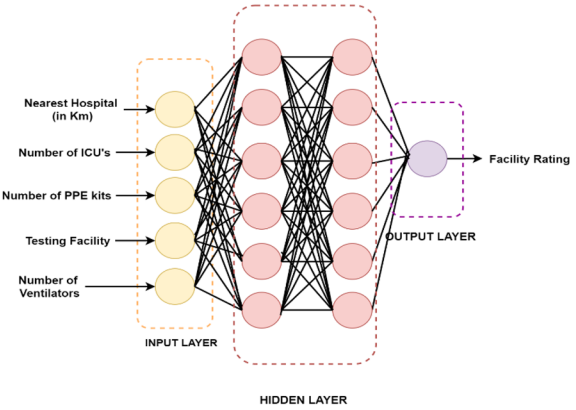


Fig. 8 Back propagation Neural Network for predicting facility rating of an area

Figure 9 represents the complete structure of this component. LSTM is used for all predictions and recommendations in this component. It has one input layer with eight neurons, four hidden layers with ten neurons each and one output layer with nine neurons. It predicts the sensitivity of an area and mark it as red, orange, or green zone depending upon the risk rating of an area, risk rating of neighbors and medical facilities available in an area. Red zone implies that COVID-19 has affected a lot and situation might be out of control, so complete lockdown should be imposed. Orange zone indicates that COVID-19 has affected the area, but partial lockdown can serve the purpose such as opening the stores on alternate days, shutting down the schools and colleges, etc. On the other hand, green zone means either no infection at all or situation completely under control such that it does not require any lockdown, keeping the social distance and personal precautions can help to prevent infection. Furthermore, this component would also recommend to seal borders such as blocking the traffic from neighboring cells depending upon the risk rating of the neighbor. This would efficiently help to regulate the spread of COVID-19. Table 3 shows the inputs and outputs to this neural network. The complete working and algorithm of the LSTM has been explained in "LSTM based Risk Rating".

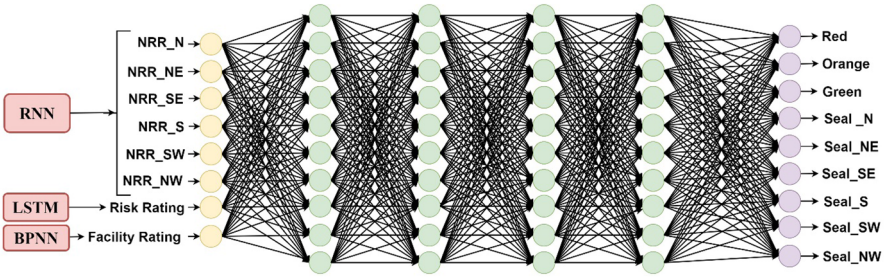


Fig. 9 LSTM to mark the sensitivity of an area

Table 3 Inputs and outputs to the LSTM

Neural Network	Input	Notation of input	Output	Notation of output
Long Short Term Memory	Neighbor Risk Rating from North border	NRR_N	Area marked as red zone	Red
	Neighbor Risk Rating from North East border	NRR_NE	Area marked as orange zone	Orange
	Neighbor Risk Rating from South East border	NRR_SE	Area marked as green zone	Green
	Neighbor Risk Rating from South border	RR_S	Recommendation to seal northern border	Seal_N
	Neighbor Risk Rating from South West border	NRR_SW	Recommendation to seal north eastern border	Seal_NE
	Neighbor Risk Rating from North West border	NRR_NW	Recommendation to seal south eastern border	Seal_SE
	Risk rating of current hexagonal area	Risk rating	Recommendation to seal southern border	Seal_S
	Rating of medical and healthcare facilities in an area	Facility rating	Recommendation to seal south western border	Seal_SW
			Recommendation to seal north western border	Seal_NW

Experiment Evaluation and Performance Analysis

The CASE-CF works in three stages and has multiple inputs and outputs to finally predict the condition of lockdown in an area. After a thorough search over the internet, a complete data set which contains all datapoints was not found. Therefore, it was decided to create a data set from already available data over various websites. A total of 22 hexagons have been chosen with different data value for 10 months. Data is stored every day, so, each hexagon has one data point for each day. This section explains in detail each neural network training as well as the overall prediction results.

As described in Sect. 3.1, for the implementation of counter for inflow and outflow from any road is done using the algorithm and code proposed by [37]. However, the use of this algorithm was not possible, because camera feeds of any COVID infected area was not available over the internet and it was not possible to create one. Therefore, a numeric input was provided for the inflow and the outflow. Data available to COVID was studied from various sources [38, 39] and synthetic data was generated using normal, poison and Gaussian distributions [40] in Python 3.7 framework. Section 4.1 explains the collection of various data values which were clubbed to form an overall data set that is further fed to neural networks for prediction and classification.

Synthetic Data Set Generation

Calculation of population inflow and outflow from all six edges of any hexagon is an important aspect. There are various implementations available on the internet which can count the number of cars moving in any given direction from the road. Therefore, collection of this data is not direct part of CASE-CF. To test the framework, direct input is used for calculation of population inflow and outflow from any hexagon. Poisson distribution is used to randomize the inflow and outflow for all 22-hexagon taken to study the performance of the CASE-CF. Poisson distribution is used to simulate the population for probability of number of events in a specified time frame. It is discreet event simulation distribution with an argument called arrival rate. Whenever the arrival rate is fixed it is homogenous Poisson process, on the contrary, when arrival rate varies as a function of time it is non-homogenous Poisson process. For our simulation, a constant arrival rate is taken which is 1500 which leads to homogenous Poisson process. Poisson distribution is used, where calling population size is exceptionally large and probability of having an event is small. In our case both the conditions are satisfied, population of state (Haryana) is huge and people who can enter any boundary is small. Therefore, Poisson distribution is well suited for this task.

This distribution is also used to calculate number of inflow and outflow from all the directions of a hexagon. For the calculation of risk density of any hexagon, three input features are used in a backpropagation neural network, as shown in Fig. 1. This data was taken for 22 districts that was displayed on various websites

	District Name	hexagon_no.	Total Inflow	Total Outflow	IN_N	IN_NE	IN_SE	IN_S	IN_SW	IN_NW	...	ICU_BED	TESTING_LABS	NO_VENTILATOR	LABEL
0	Ambala	1	8441	11356	772	1737	462	1699	1428	2343	...	120	2.0	16	Green
1	Bhiwani	2	11066	8878	1679	1011	2938	322	2934	2182	...	180	NaN	24	Green
2	Charkhi Dadri	3	8931	7613	2254	239	2022	2645	1036	735	...	30	NaN	4	Green
3	Faridabad	4	7645	10100	1393	2718	780	683	212	1859	...	90	4.0	12	Red
4	Fatehabad	5	7999	7050	429	2716	204	2188	1133	1329	...	30	NaN	4	Green
5	Gurugram	6	10794	6759	2672	313	2814	1186	2261	1548	...	60	13.0	8	Red
6	Hissar	7	10859	5809	1090	1560	411	2168	2842	2788	...	75	3.0	10	red
7	Jhajjar	8	7555	11413	274	1521	586	2335	642	2197	...	165	1.0	22	Green
8	Jind	9	9913	6969	2463	2396	1926	1206	1021	901	...	75	1.0	10	Green
9	Kaithal	10	7004	11834	1235	2297	349	962	604	1557	...	30	NaN	4	Green

Fig. 10 Snapshot of raw data containing different attributes in the data set collected

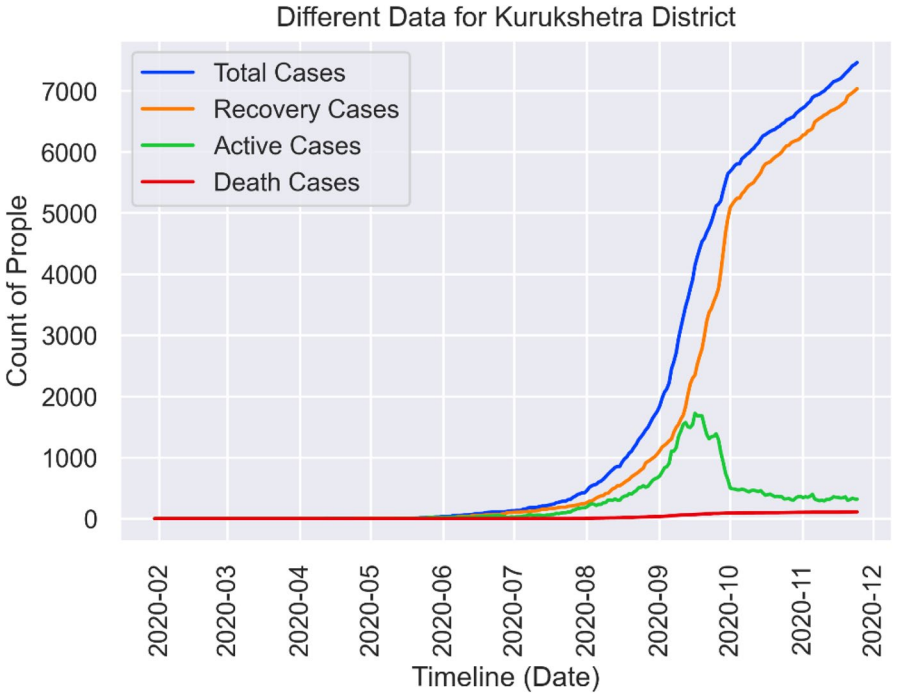


Fig. 11 Number of different cases in Kurukshetra District of Haryana

of Haryana government. Most of the data such as number of hospitals, bed, labs, active cases, recovery has been taken from covidindia.org website [41]. Figure 10 shows the snapshot of the available data for the month of November 2020. Similar data was collected from January 2020 to November 2020. Figure 11 shows the trend of number of cases reported, recovered and death in the Kurukshetra district of Haryana starting from January 2020 to November 2020. Similarly, data was

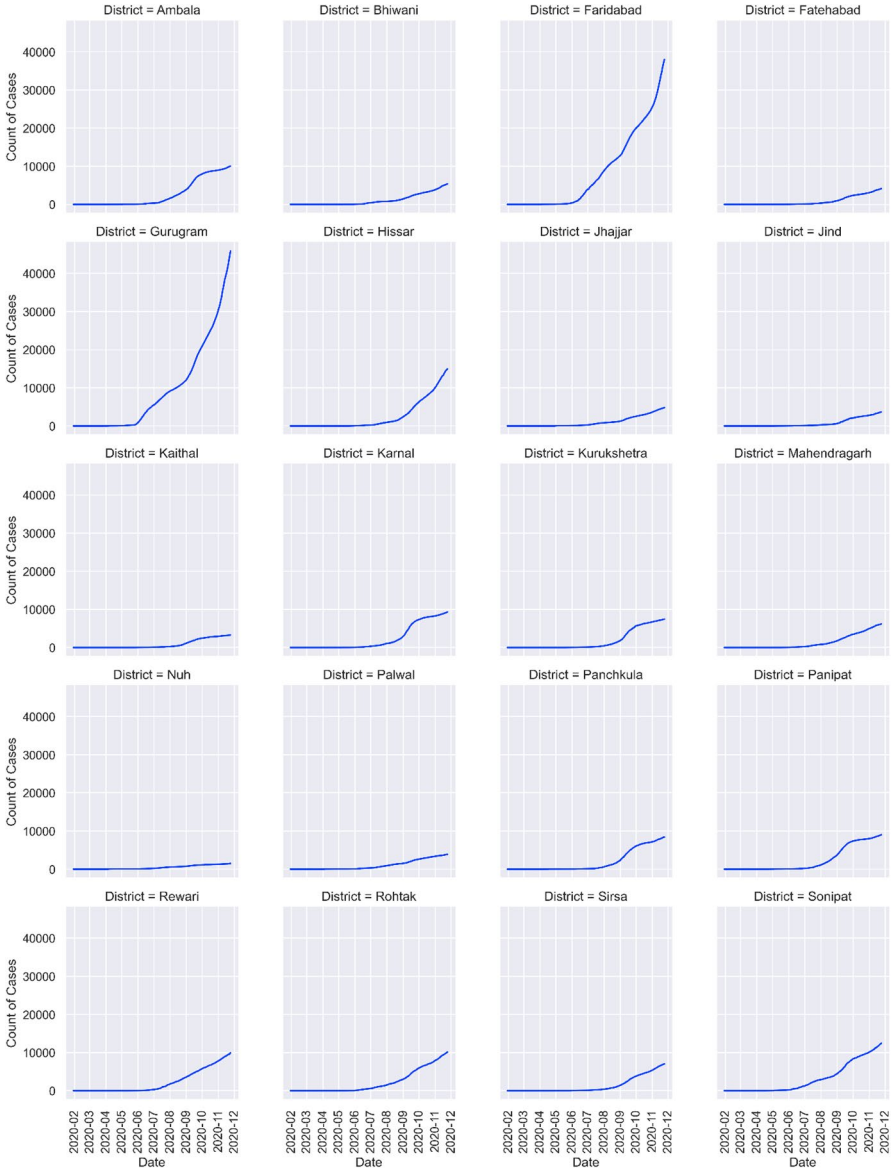


Fig. 12 Trend of increase in Total number of cases in all 22 districts of Haryana

collected for all the 22 districts of Haryana but only one district graph is shown in the paper for its brevity. Furthermore, Fig. 12 shows the total number of cases for every district of Haryana in a single graph for the comparison also. Figure 13 shows the heatmap of number of people moving in and out of a hexagon from each side.

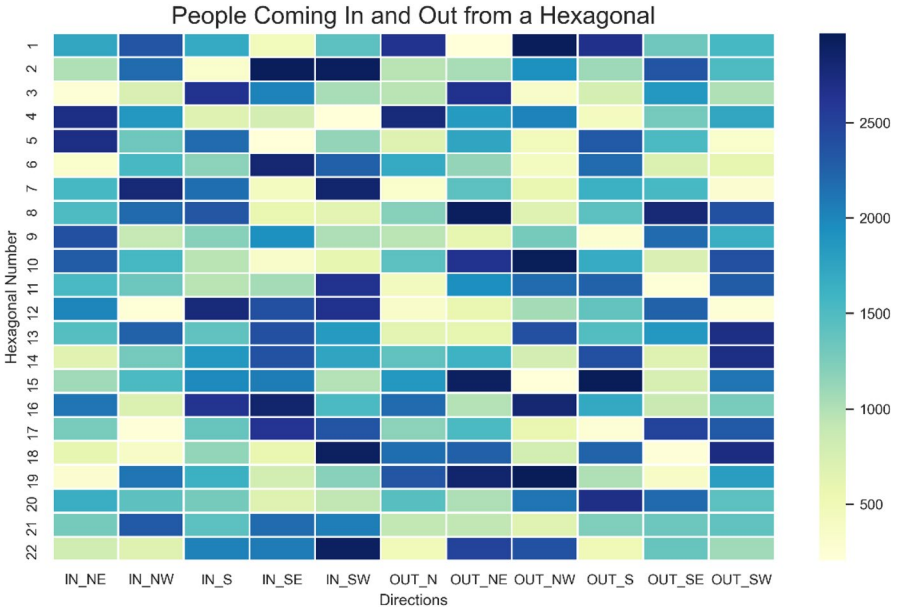


Fig. 13 Heatmap for different traffic coming from different edges of each hexagon

Neural Network Training and Data Set Snapshots

Different neural networks are trained in sequence to find the training loss and accuracy of 22 hexagons. The parameter used in every neural network are listed in Table 4 and their respective graphs for training and accuracy measures in Fig. 14. Furthermore, Fig. 15 provides the color mapping of different districts of Haryana used in the experimental evaluation. It is point to be noted that the sequence of districts presented in Fig. 15 does not reflect their actual placement on the ground.

Where *RD* risk density, *NRR* neighbor risk rating, *RR* risk rating, *FR* facility rating, *BP* back propagation, *NN* neural network, *RNN* recurrent neural network, *LSTM* long short-term memory, *MAE* mean absolute error, *MCE* multiclass cross entropy

A back propagation neural network is used for calculating the risk density of each hexagon based on the synthetic data set developed in Sect. 4.1. Table 4 shows different parameters used in the experimental calculation of risk density. Tensorflow framework is used for the implementation of all the neural networks of the CASE-CF. Training error rate and accuracy of different neural networks proposed in this framework are shown in Fig. 14. Table 5 provides the value of various performance parameters of the proposed framework. The proposed framework does not have class imbalance problem which is the reason that all performance evaluation parameters achieved good score. Final accuracy is 85% with precision, recall and *f*-measure values above the score of 0.8. Figure 15 shows the result of the proposed deep neural network in this paper using different inputs as mentioned in testbed and various intermediate values. Figure 15 also shows the final recommendation of proposed neural network using data collected from January 2020 till November 2020. Most of the hexagon's are marked as green which

Table 4 Risk density neural network parameters

S. no	Parameter	RD	NRR	RR	FR	Final
1	Type	Dense BP NN	Dense RNN	Dense LSTM	Dense BP NN	Dense LSTM
2	Model	Sequential	Sequential	Sequential	Sequential	Sequential
3	Number of Input Neuron	3	18	3	5	8
4	Number of Output Neuron	1	6	1	1	9
5	Number of Hidden Layers	2 (4 Neuron Each)	4	1	2 (6 Neuron Each)	4 (10 neuron Each)
6	Activation Function in Hidden Layer	ReLu	ReLu	ReLu	ReLu	ReLu
7	Output Layer Activation Function	Leaky ReLu	Softmax	Leaky ReLu	Sigmoid	Softmax
8	Optimization	Short Adam	Adam	Short Adam	Short Adam	Adam
9	Regularization Technique	None	Dropout (0.2)	None	None	Dropout (0.2)
10	Mini-Batch Size	None	None	None	None	None
11	Loss Function	MAE	MCE	MAE	MAE	MCE
12	Metric	Accuracy	Accuracy	Accuracy	Accuracy	Accuracy
13	Return type [State or Sequence]	None	State	Sequence	None	State

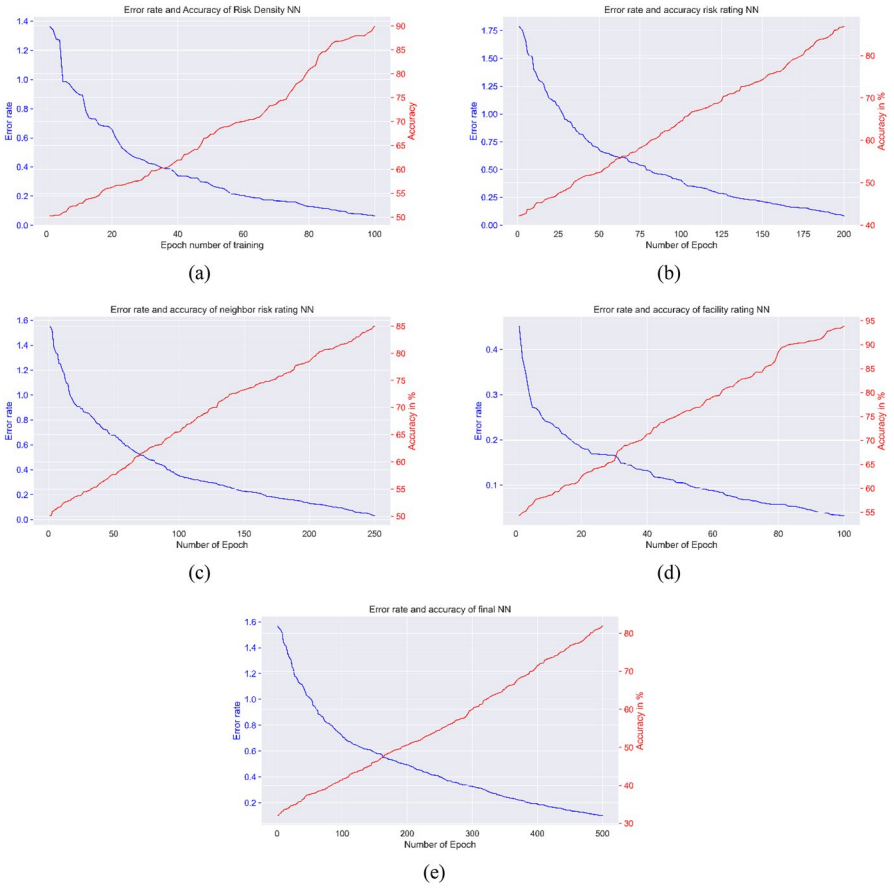


Fig. 14 Training error rate and accuracy over different number of epochs for **a** risk density neural network, *b* risk rating neural network, **c** neighbor risk rating neural network, **d** facility rating neural network and **e** final sensitive area prediction and seal border recommendation generation neural network

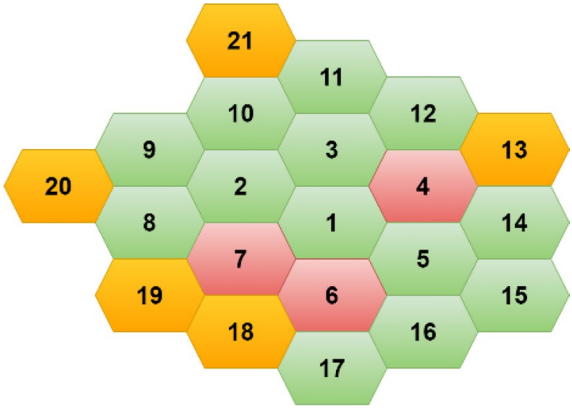


Fig. 15 Final color marks of hexagons

Table 5 Evaluation parameter values for all neural networks of the proposed framework

S. no	Parameter	RD	NRR	RR	FR	Final
1	Accuracy	0.91	0.85	0.87	0.93	0.83
2	Precision	0.92	0.86	0.88	0.93	0.84
3	Recall	0.94	0.90	0.92	0.96	0.88
4	F-Measure	0.93	0.88	0.90	0.95	0.86

are like actual marking of Haryana government very few districts of Haryana government were red districts in November 2020 which was perfectly reflected by the CASE-CF results.

Evaluation Metrics

Neural networks just like any other supervised learning can be evaluated using various parameters. The parameters used to evaluate the proposed framework are the accuracy, precision, recall and *f*-measure [42]. This section explains these terms in context of prediction and classification. The basic to understand these parameters is the confusion matrix which provides the number of instances correctly classified for each category, as shown in Fig. 16 taken from [42].

Accuracy can be calculated as the ration of number of correctly classified instance with total number of instances which can be written using Eq. 17 with reference to Fig. 16.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{FP} + \text{TN}}. \tag{17}$$

However, accuracy can be deceptive when there is class imbalance problem in the data set [42] for that recall and precision parameters are calculated. Precision provides the fraction of correctly classified positive classes as compared to total classes classified as positive. Higher the precision value better the classification system is. On the other hand, recall is defined as the fraction of correctly classified positive classes and total actual positive classes. Formulas to calculate both these measures are in Eqs. 18 and 19. Detail for these measures can be studied from [42].

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{18}$$

Fig. 16 Confusion matrix [42]

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a (TP)	b (FN)
	Class=No	c (FP)	d (TN)

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{19}$$

Performance Comparison with Variations

As discussed in Sect. 4, a standard data set was not available over the internet for the evaluation purposed of the proposed framework. Therefore, a data set was created by combining data from various sources. Similarly, no framework and model were found in the literature which can justify the performance benefits of the proposed framework. The proposed framework cannot be considered good without comparison with similar frameworks or models. Therefore, to perform the comparison a few variations were designed in the proposed framework, and it is compared with its own variations.

The main topic of study of the proposed framework is the use of multiple and different frameworks of DNNs based on the type and other factors of collected context parameters. The hypothesis as discussed in Sect. 1 was multiple layer of DNNs frameworks would better analyze the data related to COVID pandemic. Therefore, following two variations are performed on the proposed framework.

- **Variation 1 (Single Neural Network):** In this variation, single neural network was considered which takes all the inputs at once and predict the marking and lockdown condition for the hexagons. However, the single neural network used is LSTM and its hyperparameters values are listed in Table 6.
- **Variation 2 (only BPNN):** In this variation, all neural networks proposed in the framework were replaced by back propagation neural network. This means there is no RNN or LSTM network in the series of neural networks. Table 6 provides the hyper parameter used for BPNN in this variation.

Same data set with slight changes was used in the variations and results were recorded which are shown in Fig. 17. The proposed framework achieved higher score in all the parameters as compared to its both the variations. Single neural network only achieved accuracy of 48%, whereas multiple neural network framework with all neural networks achieved 64% accuracy. Similar is the case with other evaluation parameters.

Discussion of Performance Valuation and Simulation Results

In the proposed framework, context parameters are:

- Geographical location for which hexagonal structure is used.
- Infection density
- Population density
- Inflow and outflow density

Table 6 Risk density neural network parameters

S. no.	Parameter	Variation 1	Variation 2				
			RD	NRR	RR	FR	Final
1	Type	Dense LSTM	Dense BP NN	Dense BP NN	Dense BP NN	Dense BP NN	Dense BP NN
2	Model	Sequential	Sequential	Sequential	Sequential	Sequential	Sequential
3	Number of Input Neuron	28	3	18	3	5	8
4	Number of Output Neuron	9	1	6	1	1	9
5	Number of Hidden Layers	6 (15 neuron Each)	2 (4 Neuron Each)	2 (4 Neuron Each)	2 (4 Neuron Each)	2 (6 Neuron Each)	2 (4 Neuron Each)
6	Activation Function in Hidden Layer	ReLu	ReLu	ReLu	ReLu	ReLu	ReLu
7	Output Layer Activation Function	Softmax	Leaky ReLu	Leaky ReLu	Leaky ReLu	Sigmoid	Softmax
8	Optimization	Adam	Short Adam	Short Adam	Short Adam	Short Adam	Short Adam
9	Regularization Technique	Dropout (0.2)	None	None	None	None	None
10	Mini-Batch Size	None	None	None	None	None	None
11	Loss Function	MCE	MAE	MAE	MAE	MAE	MAE
12	Metric	Accuracy	Accuracy	Accuracy	Accuracy	Accuracy	Accuracy
13	Return type [state or sequence]	State	None	None	None	None	None

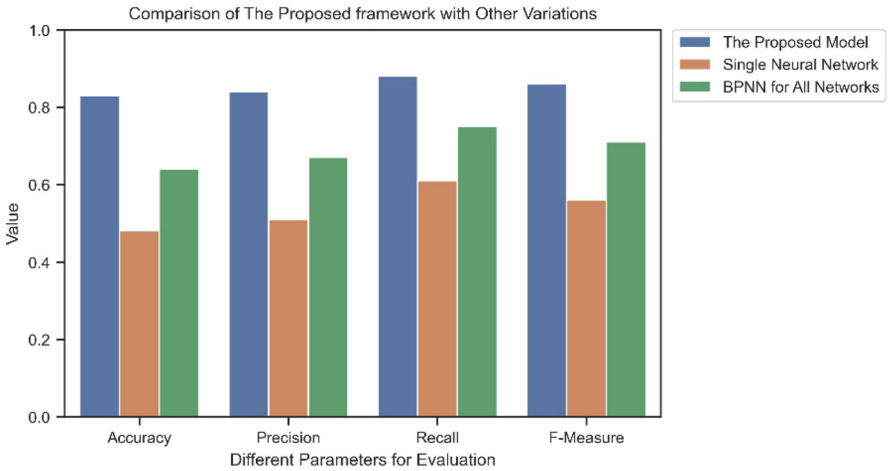


Fig. 17 Comparison of the proposed framework with its variation

- Medical facilities

The above said context parameters are different for each geographical location represented as hexagonal. However, it is not claimed that only these parameters can or should be used. The proposed framework provides method to analyse these context values using DNN models which results in better epidemic control. A user can change these context parameters depending on the requirement of the epidemic and available data collection sources.

The proposed framework was simulated on exceedingly small data set of only 22 hexagons those were created to represent 22 districts of Haryana state in India. It may seem that it would not reflect the actual performance if the proposed framework would be implemented throughout the nation. However, the number of resources used to conduct these experiments and the accuracy of the result produce provide the exact evidence that can be implemented throughout a nation without any major resource issue. The proposed framework is self-contained, because it automatically collects most of the data, perform the feature extraction and at the end generate the recommendations. Most of the neural networks, for example, risk rating calculation and risk density calculation were able to achieve the desired accuracy and loss with less than 100 epochs of training only. However, the last neural network which need to predict the sensitive area marking as well as the recommendation to seal some borders had an accuracy close to 83%. All other neural networks accuracy was close to 90% and one neural network also achieved an accuracy of 95%. The accuracy of 85% is not less and it does not reflect any kind of error generated by the system. If the government were able to predict the sensitive areas within accuracy of 83% during COVID-19 pandemic, it would have helped to decline COVID-19 cases in an area and a lot of businesses might have been saved.

Furthermore, the comparison of the proposed framework with its two variations also proved the viability of the proposed hypothesis that multiple neural networks

should be used and based on the context data specific DNN framework should be implemented. The proposed framework achieved 35% more accuracy as compared to single neural network and 19% more accuracy when same type of neural network was used at each point. Similar was the case with other evaluation parameters such as precision, recall and *f*-measure as shown in Fig. 16.

Conclusion

Deep neural networks have capabilities to handle multiple data inputs and process it to make recommendations which can benefit everyone, and COVID-19 is not any expectation in this regard. COVID-19 pandemic has opened various spheres of controlling pandemic like it and resulted in large number of publications. Accurate prediction of lockdown with respect to areas and date has been proved to be first line of defence for pandemics such as COVID-19. In this paper a real-time context aware framework is proposed that takes multiple inputs and predicts the lockdown areas with different colour marking schemes as well as predict which roads to be blocked for traffic. Hexagon based geo-mapping is proposed, because it eliminates the overlapping and underlapping of areas. For experimental setup and performance evaluation, real-time data was crawled from the web for 22 districts of Haryana state of India. Results proved the effectiveness of proposed framework.

High load and latency on the centralized servers such as cloud computing may hinder the implementation of the proposed framework in any nation. Therefore, in the future work Fog computing based decentralized computing framework may be designed for better utilization of local resources and fast results. Furthermore, data set created in this paper still needs to be updated and more features need to be added. Data set upgradation and using the proposed framework on different data set is also the future work. The proposed framework can be extended with more number of neural network with new parameters related to COVID.

References

1. Guo, Y.R., et al.: The origin, transmission and clinical therapies on coronavirus disease (COVID-19) outbreak—an update on the status. Mil. Med. Res. (2020). https://doi.org/10.1186/s40779-020-00240-0
2. Lau, H., et al.: The positive impact of lockdown in Wuhan on containing the COVID-19 outbreak in China. J. Travel Med. (2020). https://doi.org/10.1093/jtm/taaa037
3. WHO | Disease outbreaks. https://www.who.int/emergencies/diseases/en/. Accessed Dec 07, 2020
4. Sacks, J.A., et al.: Introduction of mobile health tools to support Ebola surveillance and contact tracing in Guinea. Glob. Heal. Sci. Pract. **3**(4), 646–659 (2015). https://doi.org/10.9745/GHSP-D-15-00207
5. Sandhu, R., Sood, S.K., Kaur, G.: An intelligent system for predicting and preventing MERS-CoV infection outbreak. J. Supercomput. **72**(8), 3033–3056 (2016). https://doi.org/10.1007/s11227-015-1474-0
6. Meegan, J.M.: Yellow fever. In: Handbook of Zoonoses, Second Edition, Section B: Viral Zoonoses. (2017)

7. Ibarra-Vega, D.: Lockdown, one, two, none, or smart. Modeling containing COVID-19 infection. A conceptual model. Sci. Total Environ. **730**, 138917 (2020). https://doi.org/10.1016/j.scitotenv.2020.138917
8. Perera, C., Qin, Y., Estrella, J.C., Reiff-Marganiec, S., Vasilakos, A.V.: Fog computing for sustainable smart cities: a survey. ACM Comput. Surv. (2017). https://doi.org/10.1145/3057266
9. Kharel, J., Reda, H.T., Shin, S.Y.: An architecture for smart health monitoring system based on fog computing. J. Commun. **12**(4), 228–233 (2017). https://doi.org/10.12720/jcm.12.4.228-233
10. Jeong, J.S., Kim, M., Yoo, K.H.: A content oriented smart education system based on cloud computing. Int. J. Multimed. Ubiquitous Eng. **8**(6), 313–328 (2013). https://doi.org/10.14257/ijmue.2013.8.6.31
11. Aggarwal, C.C.: Neural networks and deep learning—a textbook. Springer International Publishing, Cham (2018)
12. Asano, Y.: Learning OpenCV 3 computer (2012)
13. OpenCV: OpenCV Tutorials. Web. http://docs.opencv.org/3.0.0/d9/df8/tutorial_root.html (2016). Accessed Jun 24, 2021
14. Chollet, F.: Keras: The python deep learning library. *Keras.Io* (2015)
15. Nelli, F. and Nelli, F.: Deep learning with TensorFlow. In: Python Data Analytics. (2018)
16. Coronavirus disease-19: quarantine frame-work for travelers entering Korea. Osong Public Heal. Res. Perspect. **11**(3), 133–139. (2020). https://doi.org/10.24171/j.phrp.2020.11.3.04
17. Park, Y.J., et al.: Development and utilization of a rapid and accurate epidemic investigation support system for COVID-19. Osong Public Heal. Res. Perspect. **11**(3), 118–127 (2020). https://doi.org/10.24171/j.phrp.2020.11.3.06
18. Won Sonn, J., Lee, J.K.: The smart city as time-space cartographer in COVID-19 control: the South Korean strategy and democratic control of surveillance technology. Eurasian Geogr. Econ. (2020). https://doi.org/10.1080/15387216.2020.1768423
19. Yigitcanlar, T., Butler, L., Windle, E., Desouza, K.C., Mehmood, R., Corchado, J.M.: Can building 'artificially intelligent cities' safeguard humanity from natural disasters, pandemics, and other catastrophes? An urban scholar's perspective. Sensors (Switzerland) **20**(10), 2988 (2020). https://doi.org/10.3390/s20102988
20. Sonn, J.W., Kang, M., Choi, Y.: Smart city technologies for pandemic control without lockdown. Int. J. Urban Sci. **24**(2), 1–3 (2020). https://doi.org/10.1080/12265934.2020.1764207
21. Hong, Z., et al.: Telemedicine during the COVID-19 pandemic: experiences from Western China. J. Med. Internet Res. **22**(5), e19577 (2020). https://doi.org/10.2196/19577
22. Sun, F., et al.: Smartphone-based multiplex 30-minute nucleic acid test of live virus from nasal swab extract. Lab Chip **20**(9), 1621–1627 (2020). https://doi.org/10.1039/d0lc00304b
23. Rahman, M.S., Peeri, N.C., Shrestha, N., Zaki, R., Haque, U., Hamid, S.H.A.: Defending against the Novel Coronavirus (COVID-19) Outbreak: How Can the Internet of Things (IoT) help to save the World? Heal. Policy Technol. (2020). https://doi.org/10.1016/j.hlpt.2020.04.005
24. Podpora, M., Gardecki, A., Beniak, R., Klin, B., Vicario, J.L., Kawala-Sterniuk, A.: Human interaction smart subsystem—extending speech-based human-robot interaction systems with an implementation of external smart sensors. Sensors (Switzerland) **20**(8), 2376 (2020). https://doi.org/10.3390/s20082376
25. Kummitha, R.K.R.: Smart technologies for fighting pandemics: the techno- and human- driven approaches in controlling the virus transmission. Govern. Inf. Quart. **37**(3), 101481 (2020). https://doi.org/10.1016/j.giq.2020.101481. (**Elsevier Ltd**)
26. Yasaka, T.M., Lehrich, B.M., Sahyouni, R.: Peer-to-peer contact tracing: development of a privacy-preserving smartphone app. JMIR mHealth uHealth **8**(4), e18936 (2020). https://doi.org/10.2196/18936
27. Armbruster, B., Brandeau, M.L.: Contact tracing to control infectious disease: when enough is enough. Health Care Manag. Sci. **10**(4), 341–355 (2007). https://doi.org/10.1007/s10729-007-9027-6
28. Abeler, J., Bäcker, M., Buermeyer, U., Zillessen, H.: Covid-19 contact tracing and data protection can go together. JMIR Mhealth Uhealth **8**(4), e19359 (2020). https://doi.org/10.2196/19359. (**JMIR Publications**)
29. Sujath, R., Chatterjee, J.M., Hassanien, A.E.: A machine learning forecasting model for COVID-19 pandemic in India. Stoch. Environ. Res. Risk Assess. (2020). https://doi.org/10.1007/s00477-020-01827-8

30. Zhang, Y., Yu, X., Sun, H., Tick, G.R., Wei, W., Jin, B.: Applicability of time fractional derivative models for simulating the dynamics and mitigation scenarios of COVID-19. Chaos Solit. Fractals **138**, 109959 (2020). https://doi.org/10.1016/j.chaos.2020.109959
31. Parbat, D., Chakraborty, M.: A python based support vector regression model for prediction of COVID19 cases in India. Chaos Solit. Fractals **138**, 109942 (2020). https://doi.org/10.1016/j.chaos.2020.109942
32. Zhang, X., Ma, R., Wang, L.: Predicting turning point, duration and attack rate of COVID-19 outbreaks in major Western countries. Chaos Solit. Fractals (2020). https://doi.org/10.1016/j.chaos.2020.109829
33. Tuli, S., Tuli, S., Tuli, R., Gill, S.S.: Predicting the growth and trend of COVID-19 Pandemic using machine learning and cloud computing. Internet Things **11**, 100222 (2020). https://doi.org/10.1016/j.iot.2020.100222
34. Melin, P., Monica, J.C., Sanchez, D., Castillo, O.: Analysis of spatial spread relationships of coronavirus (COVID-19) Pandemic in the world using self organizing maps. Chaos Solit. Fractals **138**, 109917 (2020). https://doi.org/10.1016/j.chaos.2020.109917
35. Salgotra, R., Gandomi, M., Gandomi, A.H.: Time series analysis and forecast of the COVID-19 Pandemic in India using genetic programming. Chaos Solit. Fractals (2020). https://doi.org/10.1016/j.chaos.2020.109945
36. Ozturk, T., Talo, M., Yildirim, E.A., Baloglu, U.B., Yildirim, O., Rajendra Acharya, U.: Automated detection of COVID-19 cases using deep neural networks with X-ray images. Comput. Biol. Med. (2020). https://doi.org/10.1016/j.compbiomed.2020.103792
37. Rosebrock, A.: OpenCV People counter—PyImageSearch. https://www.pyimagesearch.com/2018/08/13/opencv-people-counter/ (2018). Accessed Jun 03, 2020
38. GitHub - CSSEGISandData/COVID-19: novel coronavirus (COVID-19) cases, provided by JHU CSSE. https://github.com/CSSEGISandData/COVID-19. Accessed Jun 04, 2020
39. Dong, E., Du, H., Gardner, L.: An interactive web-based dashboard to track COVID-19 in real time. Lancet Infect. Dis. **20**(5), 533–534 (2020). https://doi.org/10.1016/S1473-3099(20)30120-1. (**Lancet Publishing Group**)
40. Schwarz, G.: Estimating the dimension of a model. Ann. Stat. (1978). https://doi.org/10.1214/aos/1176344136
41. Covid19India: Coronavirus Outbreak in India—covid19india.org. https://www.covid19india.org/. (2020) Accessed Feb 01, 2021
42. Tan, P.N., Steinbach, M., Kumar, V., et al.: Introduction to Data Mining, First Edition (2014)