ISSN: 2690- 909X

Research Article

Advances in Neurology and Neuroscience

A step by step guidelines to project the COVID-19 cases using a Deep Learning Approach

S Marimuthu[1], Thenmozhi Mani[2], Melvin Joy[3] and L Jeyaseelan[4*]

[1]*Associate Research Officer, Department of Biostatistics, Christian Medical College and Hospital, Vellore, India*

[2]*Senior Demonstrator, Department of Biostatistics, Christian Medical College and Hospital, Vellore, India*

[3]*Research Assistant in Statistics, Translational and Clinical Research Institute, Faculty of Medical Sciences, Newcastle University, UK*

[4]*Professor of Biostatistics, College of Medicine, MBR University of Medicine and Health Sciences, Dubai, UAE*

[*]Corresponding author
L Jeyaseelan, Professor of Biostatistics, College of Medicine, MBR University of Medicine and Health Sciences, Dubai, UAE

Submitted: 28 Jun 2022; **Accepted**: 07 Jul 2022; **Published**: 15 Jul 2022

Citation: *S Marimuthu, Thenmozhi Mani, Melvin Joy and L Jeyaseelan (2022). A step by step guidelines to project the COVID-19 cases using a Deep Learning Approach. Adv Neur Neur Sci. 5(3), 130- 137.*

Abstract

Background: *Numerous mathematical models have been developed to forecast COVID-19 cases and helped to plan effectively by strengthening public health infrastructure and services. Many researchers incorporated the long-short term memory model (LSTM) but they have not clearly explained the workflow and steps involved in this model. Moreover, being relatively new, these models are not yet popular among biomedical researchers due to a lack of expertise. This paper presents such models as a tutorial for easy understanding and appropriate use. This includes Python codes and real-time data with instructions for implementation to forecast pandemics like COVID-19. Data and Methods: Daily cases in India from 1-Dec-2021 to 10-Feb-2022 and, in the UK from 1-May-2021 to 10-Feb-2022 were used to train the models. We used Convolutional-LSTM (CNN-LSTM) model and simple LSTM models to forecast COVID-19 cases. Models were validated using data from 11 to 25-Feb-2022.*

Results: *CNN-LSTM and simple LSTM were fitted very well with R2 0.95 and 0.97 for India. The models were validated with RMSE and it was 9972.81 and 19285.57 for CNN-LSTM and the simple LSTM model. The R2 value of CNN-LSTM and simple LSTM models for UK data were 0.77 and 0.84 respectively. RMSE was 12111.95 for CNN-LSTM and 8935.75 for simple LSTM in the validation.*

Conclusion: *Simple LSTM works better while training whereas the performance of CNN-LSTM was found to be better in validation. Therefore, it is suggested that train various models instead of sticking to one and revise them regularly as the behavior of an epidemic generally changes over time.*

Keywords: COVID-19 Projection; Convolutional Neural Network; Deep Learning; Forecasting; Long-Short Memory model; Recurrent Neural Network

Introduction

The COVID-19 pandemic had greatly impacted social, economic and health care systems across the world. Several countries are experience multiple waves due to the mutation of virus. Variants such as Alpha and Delta caused more severe illness whereas Omicron caused mild disease but rapid spread in the community. The continuous evolution of a virus may lead to it being one step ahead. Therefore, it is essential to have accurate forecasting of the epidemic to ensure effective planning and strengthening of health care facilities.

To obtain a handle on the pandemic, frequently updated reliable data are required. Since the beginning of the pandemic, several models such as compartmental model, logistic growth model and time series model have been widely used for projection. However, the accuracy of prediction and its uncertainties are greatly depends on the availability and quality of the data as well as model assumptions [1]. The results may vary significantly when the assumptions are violated, or there are differences in the given parameters. During COVID-19 pandemic, the availability and quality of data keep improving as the epidemic progress, which makes

predictions uncertain in the early stages and more accurate as the epidemic progress. Moreover, an epidemic may not always behave in the same manner as pathogens are likely to behave differently over time [2].

The susceptible-exposed-infectious-recovered (SEIR), susceptible-infectious-recovered (SIR) models, Agent-based models, Curve-fitting, and Logistic growth model due to the exponential nature of growth of the epidemic are commonly adopted in different biological and social processes [3-8]. Especially, the logistic growth model could be relevant for short term projection while SEIR and SIR models could be useful to estimate the maximum number of active cases and the peak of the epidemic. Malavika et al. (2020) used SIR and logistic growth model to predict the COVID-19 pandemic in India [9]. They validated the model with data from Italy and South Korea and incorporated the correction factor for India data. We have also fitted the logistic growth model and found that it under predicts the cases once the pandemic crosses the point of maximum growth. A few researchers used exponential growth model to predict the number of cumulative cases. But, this model, not being stable, does not have an upper bound and is likely to go on increasing [10]. In this scenario, the logistic growth model is preferred.

Kirbas et al. (2020) compared Auto-Regressive Integrated Moving Average (ARIMA), Non-Linear Auto Regression Neural Network (NARNN) with Long-Short Term Memory (LSTM) models and reported LSTM was the most appropriate model [11]. Rguibi et al. (2020), and Namini et al. (2018) compared LSTM and ARIMA model and concluded that deep learning-based algorithms such as LSTM outperform the traditional ARIMA model [12, 13].

When there is a new phase of the epidemic begins due to new variants of the virus or mutations in the DNA structure, the researchers should validate the models with available data from other countries as it would be too early to project the cases. In such a situation, it would be appropriate to work with a model that depends on limited time points and parameters, if forecasting is the main intention. The limitations of the above models made us to explore more on deep learning models such as Recurrent Neural Network (RNN) and Long Short Term Memory (LSTM).

RNN is designed to recognize the sequential characteristics of data and use patterns to predict the next likely scenario. Unlike other neural networks such as Artificial Neural Network (ANN), internal memory in the RNN integrates previous input which allows it to make decisions by considering current input alongside learning from previous input. It can also effectively handle non-linear time series data by using non-linear activation functions.

However, RNN suffers due to vanishing gradients and exploding gradients problem [14]. Long training time, poor performance, and poor accuracy are the major issues in gradient problems. Therefore, LSTM models are suggested to overcome the issues of vanishing gradient and exploding gradient, which plagues RNN [15]. LSTMs are special kind of RNN, which are capable of learning long term dependencies by remembering information for long periods.

Furthermore, our preliminary work shows that LSTM models are better as compared to previous models. Moreover, the deep learning models would give users the flexibility to decide the hidden layers which could improve their performance. However, they are not popular among medical investigators due to lack of expertise. This paper aims to present these models in a simple way with Python codes, using real time data and adequate instructions, so that more researchers could use these methods to project pandemics like COVID-19.

Methods

Recurrent Neural Network

Rumelhart (1986) introduced the RNN [16]. The unfolded architecture of RNN is depicted in figure1. The initial hidden state (internal memory) is denoted as h0. The hidden state at time step 't' is denoted as ht which is updated by previous time step. There are two weight matrices and a column vector of bias are associated with each RNN layer. A weight matrix and a bias are associated with the output layer [Figure 1].

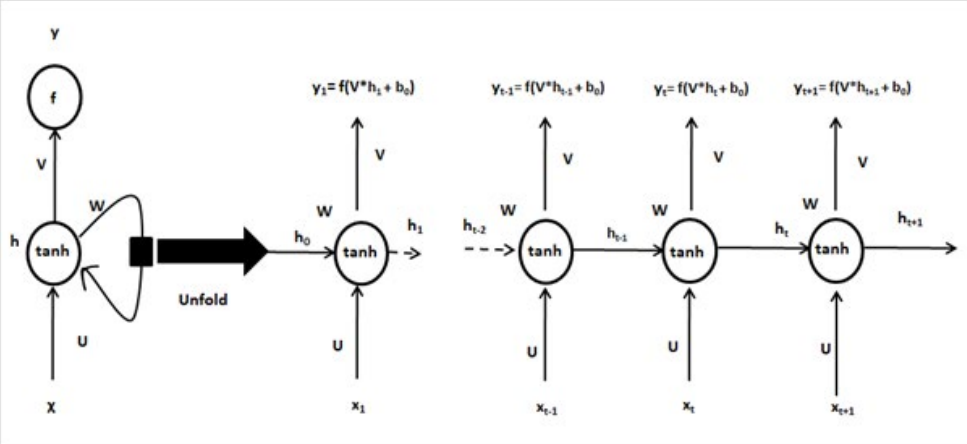


Figure 1: Simple RNN Architecture

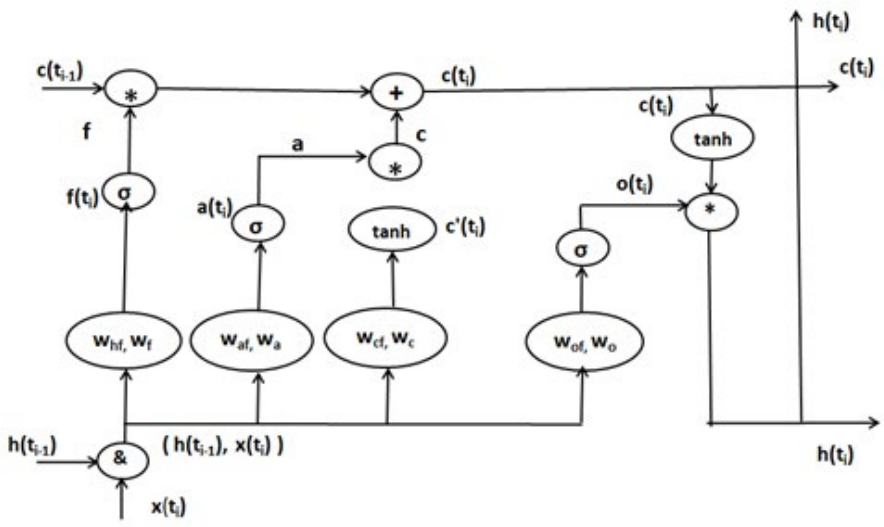
Work flow of RNN

Kavita et al. (2017) investigated the evolution of RNN model in the last three decades [17] RNN is an updated version of feed forward neural networks, improved by the addition of recurrent borders which cover adjoining temporal phases, establishing a concept of time to the design. The hidden state (h_t) is computed from the actual sample (xt) and the previous hidden state (h_{t-1}) in the network i.e $h_t = \tanh(U*x_t + W*h_{t-1} + b_h)$, where U is the weight matrix between input and hidden layer, W is the weight matrix between hidden layer to hidden layer and bh is the bias matrix. Result yt is computed by performing the activation function to state ht and weight V, $f(V*h_t + b_o)$, V is the weight matrix between 'hidden and output layer' and b_o is the bias matrix in the output layer. Hence, at time t – 1, feed x_{t-1} may affect the result y_t at time through such recurrent links.

After computing the outputs (forward pass), calculate the loss / error. Using this loss, calculate the gradient of the loss function for back-propagation. Using the gradient, update the weights (U, W, and V) and bias (b_h, and b_0) in the model accordingly so that future computations with the input data will produce more accurate results. The entire process is called 'back-propagation'. Combined with the forward pass, back-propagation is looped over and again for a given number of epochs, and the optimal values of weight matrix and bias are obtained. Unlike the ANN, the weight matrix is same throughout the process.

LSTM Models

Schmidhuber et al. (1997) proposed the LSTM neural network model [15]. The architecture of LSTM module is given in [Figure 2] which contains forgot gate, input gate, internal state and output gate [18].



Notations:

$x(t_i)$: The input value; $h(t_i)$: The current state output value at time point t_i; $c(t_i)$: Cell state value at time points t_i;

$b = \{b_a, b_f, b_c, b_0\}$ are biases of input gate, forget gate, internal state and output gate.

$\vec{w}_1 = \{w_a, w_f, w_c, w_o\}$ are weight matrices of input gate, forget gate, internal state and output gate and output.

$\vec{w}_2 = \{w_{ha}, w_{hf}, w_{hc}, w_{ho}\}$ are the recurrent weights ;

$\vec{a} = \{a(t_i), f(t_i), c(t_i), o(t_i)\}$ are the results for input gate, forget gate, internal state, and output gate.

σ and **tanh** are activation functions. **&**, **+**, and ***** represent the Concatenation, Addition, and point-wise multiplication respectively.

Figure 2: The architecture of LSTM memory cell (18)

Work Flow of LSTM Module

Wang et al. (2019) described the workflow of the LSTM model [19]. The LSTM first determines the information to be excluded from the state of the unit. This operation is implemented by a 'sigmoid' layer which is also called as forget gate. The forget gate reads the last hidden state h(t_(i-1)) and the current input x(t_i), and gives output value f(t_i) between 0 and 1 i.e.
$f(t_i) = \sigma(w_f x(t_i) + w_{hf} h(t_{i-1}) + b_f)$.

Then, it determines the information to be reserved in the current unit. There are two parts in this operation. The first part uses the sigmoid layer to decide the updating target by using last hidden state $h(t_{i-1})$ and current input $x(t_i)$ that is,
$a(t_i) = \sigma(w_a x(t_i) + w_{ha} ht_{i-1} + b_a)$.

The second part uses the 'tanh' layer to construct the vector
$c'(t_i) = \tanh(w_c x(t_i) + w_{hc} h(t_{i-1}) + b_c)$ and $c'(t_i)$ that will play a role in the subsequent unit status updates. Subsequently, the last cell state c(t_(i-1)) updated in the forget gate will take part in the following calculation to attain the current cell state $c(t_i)$:

$c(t_i) = ft \times c(t_{i-1}) + a_t \times c'(t_i)$

Lastly, the LSTM determines the output for the next moment. The $c(t_i)$ is calculated by the previous step will have two output directions, one is directly connected to the unit at the next moment as its input data, and the other requires a filtering process.

The process first passes the current data x (t_i) and the hidden state from last state $h(t_{i-1})$ into the 'sigmoid' layer as input, and the output is $o(t_i) = \sigma(w_o x(t_i) + w_{ho} h(t_{i-1}) + b_o)$. Then, $\tanh(c(t_i))$ is multiplied with $o(t_i)$, that gives the current hidden state i.e.$h(t_i) = o(t_i) \times \tanh(c(t_i))$. Finally, $h(t_i)$ is passed in the unit of the next moment.

LSTM Models

The following two LSTM models have been demonstrated
(i) Simple LSTM
(ii) Convolutional Neural Network (CNN) LSTM
Steps to construct the LSTM models are given below:

1. Plot Auto Correlation Function (ACF) /Partial Auto Correlation Function (PACF) and choose time step / lag
2. Data pre-processing
3. Reorganize the data as in the format of input to the model
4. Model building
5. Fit the model and evaluate the performance
6. Predict for future

Step 1: Plot ACF/PACF and choose time step / lag

Using time series data, draw auto correlation function (ACF) / partial auto correlation function (PACF) plots. After that choose the parameter (p or q), which defines the lag (number of time steps that the observations are correlated). The lag has significant impact on the performance of time series forecasting and it has to be identified using ACF/PACF plot.

Step 2: Data pre-processing

Unlike the classical time series model, LSTM model requires input (X) and outcome (Y) to train the model which is same as supervised learning such as regression or classification. Therefore, we have to split the data into input (X) and outcome (Y) using the lag or time step.
For instance, {Xi, i=1, 2, 3, …, n} be the daily incident cases. Suppose, lag is three, then the fourth day counts depends on previous three days (third, second and first) and fifth day counts depends on fourth, third, and second day counts, and so on. Therefore, the data should be organized as follows:

$$\begin{matrix} X & & Y \end{matrix}$$

$$\begin{bmatrix} X_1 X_2 X_3 \\ X_2 X_3 X_4 \\ X_3 X_4 X_5 \\ \ldots \end{bmatrix} = \begin{bmatrix} X_4 \\ X_5 \\ X_6 \\ \ldots \end{bmatrix}$$

Step 3: Reorganize the Data as in The Format of Input to The Model

The input data X has the shape [samples, timesteps] and it is reshaped before feed into the model. The input shape differs from each model. The input shape of the simple LSTM model is [samples, timesteps, features], where feature is always one for univariate time series analysis.

But, CNN-LSTM model has one more argument called 'subsequence'. Split the lag into number of sequence and sub-steps. The sub-steps are multiple of the kernel size in the convolutional layer. For example, if lag is six days, split the six into three subsequences and two sub-steps (3*2=6). The input shape of the CNN LSTM model is [samples, subsequences, sub-steps, features].

Step 4: Model Building

Simple LSTM:

Schmidhuber et al. (1997) proposed an LSTM model which is the simple LSTM configuration model compared to other models with one hidden LSTM layer and output layer [15]. But, in our model we used only one LSTM layer followed by four to eight dense layers which were determined based on the model performance. The structure of the LSTM model is presented in [Figure 3].

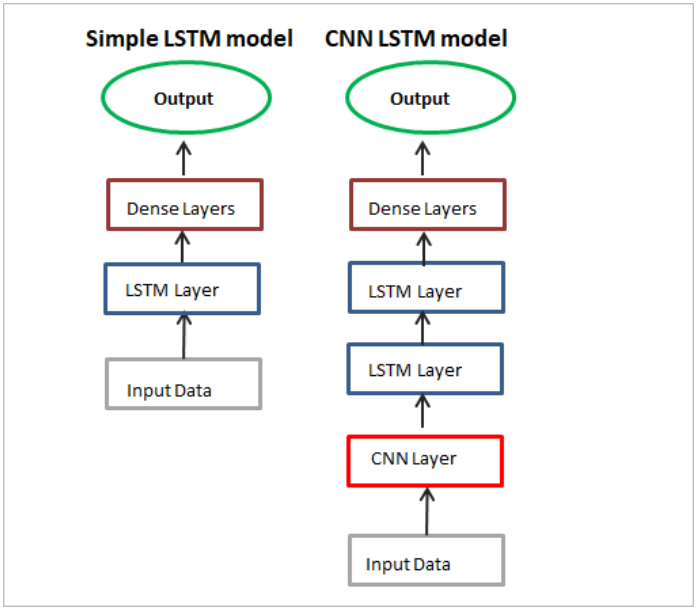


Figure 3: Structure of LSTM Models

CNN LSTM Model

LeCun et al. (1998) developed CNN model which is a type of neural network developed to work with two-dimensional image data [20]. The CNN can be very effective on automatically extracting and learning features from one-dimensional sequence data such as univariate time series data. A CNN model can also be used with an LSTM as a hybrid model where the CNN is used to interpret subsequences of input that together are provided as a sequence to an LSTM model to interpret [21]. This hybrid model is called a CNN-LSTM.

The time distributed wrapper function is used to read the subsequence of data separately. The CNN model first has a convolutional layer for reading across the subsequence that requires a number of filters and a kernel size to be specified. The number of filters is the number of reads or interpretations of the input sequence. The kernel size is the number of time steps included in each 'read' operation of the input sequence. The convolution layer is followed by a max pooling layer that distils the filter maps down to 1/2 of their size that includes the most salient features. Subsequently, these structures are flattened down to a single one-dimensional vector to be used as a single input time step to the LSTM layer.

Our CNN-LSTM model has one dimensional CNN layer followed by max-pooling layer and flatten layer. Then it has two LSTM layers with 500 neurons followed by four to eight dense layers.

Step 5: Fitting and Evaluating the Model Performance

Fit the model using the train data and evaluate the performance. The Mean Squared Error (MSE) / Mean Absolute Error (MAE) can be used as the metrics to evaluate model performance of both train and test data.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \widehat{Y}_i\right)^2 \text{ and } MAE = \frac{1}{n}\sum_{i=1}^{n}\left|Y_i - \widehat{Y}_i\right|$$

Step 6: Future Prediction

In order to predict the future value, we have to feed past data as the input. In the absence of past value, we can use our predicted value of the corresponding day as the input. The Python code for all the above steps are given in the link for the future work.

Data

Daily data of confirmed COVID-19 cases in India were taken from https://covid19tracker.in/. Data from 1st December 2021 to 10th February 2022 were used to train the models. Everyday COVID-19 cases in UK were extracted from https://covid19.who.int/info and data from 1st May 2021 to 10th February 2022 were used to train the models. The model was validated with the data from 11th February to 25th February 2022. The analysis was done using python in Google colab platform. The codes and data are available in (Figure 3).

Results

India: For the modelling of India, the COVID-19 cases reported by the Government of India from 1st December 2021 to 10th February 2022 were used. Figure 4 indicates the observed and projected COVID-19 cases in India by using simple LSTM and CNN-LSTM models. Since early December the cases started to increase and reached a peak of 3.4 lakh cases/day on 20th January. It decreased steadily after that and reached about 58,000 cases on 10th February 2022. The CNN-LSTM model shows that the cases would decrease steadily and reach about 17,000 cases on 25th February 2022. The actual reported case on same day was about 11,000. In the same manner, simple LSTM model shows a decreasing trend but the figure is slightly higher than CNN LSTM model which is about 30000 cases on 25th February 2022. The goodness of fits statistics is presented in [Table 1]. Table 2 presents the observed and projected cases of validation for both India and UK. The R square values were 0.95 and 0.97 for CNN and simple LSTM model respectively. The RMSE and MAE of CNN LSTM model were 24440.04and 15027.58 respectively. RMSE and MAE were 19252.12and 13237.22 for simple LSTM model respectively. On the other hand, for the test data, MAE of CNN and simple LSTM models were 8993.33 and 18870.87. MSE was 9972.81 and 19285.57 for CNN-LSTM and simple LSTM model respectively. The simple LSTM model fits very well as compared to CNN LSTM model. However, the performance of CNN LSTM model was quite good in the test data.

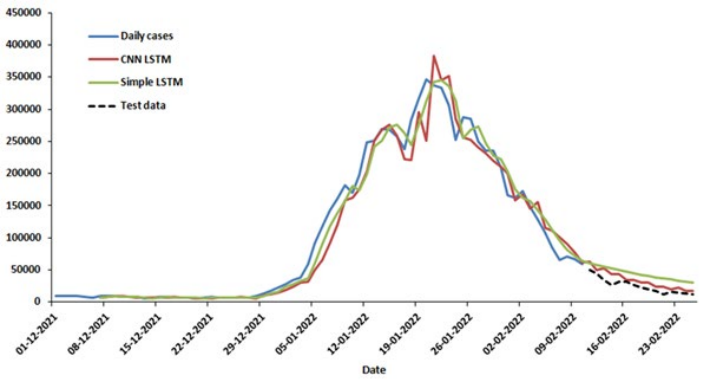


Figure 4: COVID-19 Projection for India

Table 1: Fits Statistics for India and UK

	India				The UK			
	Train data		Validation data		Train data		Validation data	
	CNN LSTM	Simple LSTM	CNN LSTM	Simple LSTM	CNN LSTM	Simple LSTM	CNN LSTM	Simple LSTM
R^2	0.95	0.97	-	-	0.77	0.84	-	-
MAE	15027.58	13237.22	8993.33	18870.87	12163.53	9722.01	10060.9	7824.821
RMSE	24440.04	19252.12	9972.81	19285.57	21287.39	17830.48	12111.95	8935.75

Note: CNN- Convolutional Neural Network; LSTM – Long Short Term Memory; RMSE - Root Mean Squared Error; MAE - Mean Absolute Error

Table 2: COVID-19 projected cases for India and UK

	India			The United Kingdom		
Date	Validation data	CNN LSTM	Simple LSTM	Validation data	CNN LSTM	Simple LSTM
11-02-2022	50399	62620	60087	66638	80751	71995
12-02-2022	45129	50337	57063	58316	51152	72275
13-02-2022	34493	52038	54483	45500	44257	53853
14-02-2022	26052	43681	51869	40844	57774	48056
15-02-2022	30987	42871	49357	41170	53938	29048
16-02-2022	30907	34424	46965	54665	65601	50484
17-02-2022	26274	34286	44696	52453	76969	37778
18-02-2022	22042	30296	42544	51520	48643	55745
19-02-2022	19753	30031	40492	44670	40035	46578
20-02-2022	16678	23535	38539	33217	39333	41775
21-02-2022	12207	23244	36679	34642	47027	45896
22-02-2022	15479	19639	34910	37667	43537	41407
23-02-2022	14484	22669	33226	45089	24988	30627
24-02-2022	13083	17453	31623	39115	50025	42111
25-02-2022	11601	17344	30098	36442	36793	40812
Note: CNN- Convolutional Neural Network; LSTM – Long Short Term Memory;						

The United Kingdom (UK): Figure 5 indicates the observed and projected COVID-19 cases in UK. The pattern of UK is found to be different from India. In the beginning of May 2021, the cases were increasing slowly, and remaining steady from August 2021 till December 2021. After that, the cases increased rapidly and reached a peak of 2.7 lakh cases on 06th January 2022, subsequently it decreased rapidly. In the beginning of February 2022, UK experienced an average of 77,000 cases every day. Simple LSTM model showed that the cases would decrease slowly and reach about 40,000 cases on 25thFebruary 2022. CNN-LSTM model showed that the cases would be decreasing a little faster than what was suggested by the simple LSTM model and reach about 36,000 cases on 25th February 2022 which is close to the reported cases on the same day. The R square value of CNN and simple LSTM models were 0.77 and 0.84 respectively. The RMSE was 21287.39for CNN LSTM and it was17830.48for simple LSTM. The MAE for CNN LSTM was 12163.53 and 9722.01 for simple LSTM model. For the test data, MAE for CNN and simple LSTM model were 10060.92 and 7824.82 respectively. The MSE values were 12111.95 and 8935.75 for CNN and simple LSTM model respectively.

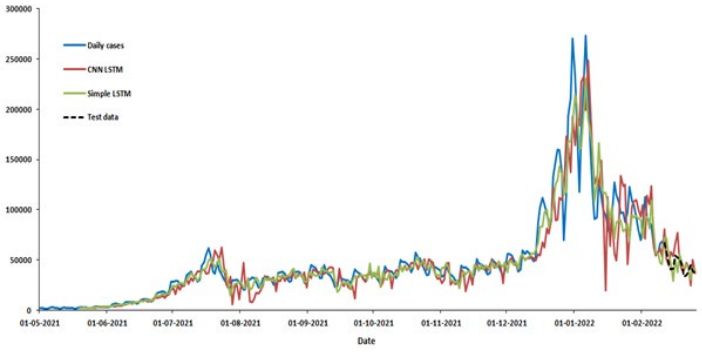


Figure 5: COVD-19 Projection for the United Kingdom

Discussion

Short- and long-term forecasting of the number of cases in any epidemic or pandemic is a useful tool for government, public health

agencies, programme officials and policymakers for better panning and preparedness of health system. Given the rapid increase of COVID-19 cases worldwide, it is imperative that these tools are widely made available to estimate the required health care resources including the size of health care workers, number of hospital beds, ventilators, and personal protective equipment (PPE). These would in turn be directly reflected in better outcomes for patients as well as the number of lives saved due to a fatal disease [22].

Many researchers incorporated LSTM models in their work and compared it with other models. Barman (2020) forecasted cumulative COVID-19 cases using LSTM model and compared it with ARIMA [22]. They found relatively less errors in the prediction using LSTM models compared to ARIMA models. Shyam et al. (2020) analyzed the performance of LSTM model and compared it with two parameter reduced LSTM model [23]. Chimmula et al. (2020) evaluated the key features of predicting the trends of COVID-19 cases and possible end of current COVID-19 outbreak in Canada and around the world in 2020 by using long short-term memory (LSTM) networks [24]. Hawas et al. (2020) predicted the daily infection cases in Brazil using LSTM and GRU (Gated Recurrent Unit) model [25]. Kafieh et al. (2021) applied multilayer perceptron, random forest, and various LSTM models for nine countries and forecasted the epidemic [26]. However, most of the articles were published in technical journals not in medical or epidemiology journals. Moreover, none of them explained the steps involved in the LSTM models in a simple way. Therefore, we did not compare LSTM with other models as many studies have indicated that it is better than other classical time series models. Besides, we aimed to train young researchers who are interested in this field but have little capacity in using this method. This manuscript provides the details of using LSTM model using Python codes. The codes are made available for the readers in such a way that they can understand the hyper parameters and the steps easily.

One of the important steps in LSTM models is to fix dense layers and hyper parameters such as number of lag and epochs, however, there is no clear guidance of choosing lag. Some researchers training the models with various time step values and choose the optimal one which gives the better performance. However, we do not know exactly how many times points the future data would depends on. Therefore, based on our experience, we choose the time step (lag) by plotting ACF / PACF plot. However, when the pattern of the epidemic changes very frequently, we recommend to revise the model either weekly or once in a couple of weeks. Invariably, we used four to eight dense layers in our model. The number of dense layers will be adjusted based on model performance. Except the output layer, we used the "he_uniform" distribution to initialize the weights of the layers and adopt L2 kernel regularization to avoid over fitting. Rectified Linear Unit (ReLU) activation function was used in both LSTM and all the dense layers. In addition, Adam optimizer technique was used to update the weights. The 'call backs' technique was used to decrease the learning rate when there is no improvement in the past epochs.

We used India and UK data for modelling. However, UK data showed a fluctuating pattern and after reaching the peak, a sharp decrease was also recorded. In both cases, LSTM models work well with an R2 value of about 0.96 for India and about 0.80 for UK. This statistic suggested that the model performs well at various trend of the epidemic curves.

Limitations

One of the concerns is that the strategies and polices of the government may change based on the virulence and transmissibility of the virus. Sudden changes in the testing policy may mislead predictions. Therefore, the model may over or underestimate the cases. The time series analysis and forecasting presented in this study are restricted to short term forecasts. The users should have minimal knowledge of time series analyses and able to estimate the Auto Regression parameter (p) and Moving Average parameter (q) using autocorrelation function and partial auto-correlation function. The model needs sufficient data to map the pattern of the epidemic. We carried out all the projections using python code and, therefore the users of this method need to know the basics of python codes.

Conclusion

Simple LSTM model works better than CNN-LSTM model while training. Nonetheless, the performance of CNN is better in the validation. Therefore, our suggestion for the researcher is to, train various LSTM models instead of depending on a single model and forecast the future with reporting the range of values. Moreover, we recommend revising the model frequently, either weekly or once in a couple of weeks, as the behaviour of an epidemic may rapidly change over time.

Acknowledgments

Nil

Funding

None

Conflict of Interest

The authors have no conflicts of interest to declare

References

1. Moran, K. R., Fairchild, G., Generous, N., Hickmann, K., Osthus, D., Priedhorsky, R., ... & Del Valle, S. Y. (2016). Epidemic forecasting is messier than weather forecasting: the role of human behavior and internet data streams in epidemic forecast. The Journal of infectious diseases, 214(suppl_4), S404-S408.
2. World Health Organization. (2018). Managing epidemics: key facts about major deadly diseases. World Health Organization.
3. Roda, W. C., Varughese, M. B., Han, D., & Li, M. Y. (2020). Why is it difficult to accurately predict the COVID-19 epidemic?. Infectious disease modelling, 5, 271-281.
4. Arti, M. K., & Bhatnagar, K. (2020). Modeling and predictions for COVID 19 spread in India. ResearchGate, DOI: DOI, 10.
5. Singh, R., & Adhikari, R. (2020). Age-structured impact of social distancing on the COVID-19 epidemic in India. arXiv preprint arXiv:2003.12055.

6. Pandey, G., Chaudhary, P., Gupta, R., & Pal, S. (2020). SEIR and Regression Model based COVID-19 outbreak predictions in India. arXiv preprint arXiv:2004.00958.
7. Ivorra, B., Ferrández, M. R., Vela-Pérez, M., & Ramos, A. M. (2020). Mathematical modeling of the spread of the coronavirus disease 2019 (COVID-19) taking into account the undetected infections. The case of China. Communications in nonlinear science and numerical simulation, 88, 105303.
8. Mandal, S., Bhatnagar, T., Arinaminpathy, N., Agarwal, A., Chowdhury, A., Murhekar, M., ... & Sarkar, S. (2020). Prudent public health intervention strategies to control the coronavirus disease 2019 transmission in India: A mathematical model-based approach. The Indian journal of medical research, 151(2-3), 190.
9. Malavika, B., Marimuthu, S., Joy, M., Nadaraj, A., Asirvatham, E. S., & Jeyaseelan, L. (2021). Forecasting COVID-19 epidemic in India and high incidence states using SIR and logistic growth models. Clinical Epidemiology and Global Health, 9, 26-33.
10. Rai, B., Shukla, A., & Dwivedi, L. K. (2020). COVID-19 in India: predictions, reproduction number and public health preparedness. MedRxiv.
11. Kırbaş, İ., Sözen, A., Tuncer, A. D., & Kazancıoğlu, F. Ş. (2020). Comparative analysis and forecasting of COVID-19 cases in various European countries with ARIMA, NARNN and LSTM approaches. Chaos, Solitons & Fractals, 138, 110015.
12. Rguibi, M. A., Moussa, N., Madani, A., Aaroud, A., & Zine-Dine, K. (2022). Forecasting covid-19 transmission with arima and lstm techniques in morocco. SN Computer Science, 3(2), 1-14.
13. Siami-Namini, S., Tavakoli, N., & Namin, A. S. (2018, December). A comparison of ARIMA and LSTM in forecasting time series. In 2018 17th IEEE international conference on machine learning and applications (ICMLA) (pp. 1394-1401). IEEE.
14. Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. IEEE transactions on neural networks, 5(2), 157-166.
15. Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.
16. Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1985). Learning internal representations by error propagation. California Univ San Diego La Jolla Inst for Cognitive Science.
17. Kavita, D., Saxena, A., & Joshi, J. Using of Recurrent Neural Networks (RNN) Process.
18. Abbasimehr, H., Shabani, M., & Yousefi, M. (2020). An optimized model using LSTM network for demand forecasting. Computers & industrial engineering, 143, 106435.
19. Wang, H., Song, Y., & Tang, S. (2019). LSTM-based Flow Prediction. arXiv preprint arXiv:1908.03571.
20. LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324.
21. Lu, W., Li, J., Li, Y., Sun, A., & Wang, J. (2020). A CNN-LSTM-based model to forecast stock prices. Complexity, 2020.
22. Barman, A. (2020). Time series analysis and forecasting of covid-19 cases using LSTM and ARIMA models. arXiv preprint arXiv:2006.13852.
23. Shyam Sunder Reddy, K., Padmanabha Reddy, Y. C. A., & Mallikarjuna Rao, C. (2020). Recurrent neural network based prediction of number of COVID-19 cases in India.
24. Chimmula, V. K. R., & Zhang, L. (2020). Time series forecasting of COVID-19 transmission in Canada using LSTM networks. Chaos, Solitons & Fractals, 135, 109864.
25. Hawas, M. (2020). Generated time-series prediction data of COVID-19′ s daily infections in Brazil by using recurrent neural networks. Data in brief, 32, 106175.
26. Kafieh, R., Arian, R., Saeedizadeh, N., Amini, Z., Serej, N. D., Minaee, S., ... & Haghjooy Javanmard, S. (2021). COVID-19 in Iran: forecasting pandemic using deep learning. Computational and mathematical methods in medicine, 2021.