Covid-19 Spatio-temporal Prediction Using Combined Graph Convolution - LSTM Model

Project submitted
In partial fulfilments of the requirements for the degree of

MASTER OF COMPUTER APPLICATION

By

SANTU BERA
Roll No: **001910503014**
Registration No: **149877 of 2019-20**
Exam Roll No: **MCA226014**

Under the supervision of

DR. ANASUA SARKAR

Department of Computer Science & Engineering
Faculty of Engineering and Technology
Jadavpur University
Kolkata – 7000 032
India

Jadavpur University
Faculty of Engineering and Technology
Department of Computer Science & Engineering

CERTIFICATE

This is to clarify that the project entitled "**Covid-19 Spatio-temporal Prediction Using Combined Graph Convolution - LSTM Model**" has been completed by Santu Bera. This work is carried out under the supervision of Dr. Anasua Sarkar, Assistant Professor, Jadavpur University in partial fulfilment for the award of the degree of Master of Computer Application of the department of Computer Science and Engineering, Jadavpur University, during the session 2019-2022. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the said degree.

Dr. Anasua Sarkar
Project supervisor
Computer Science & Engineering
Jadavpur University

Countersigned:

Prof. Anupam Sinha
Head of the department
Computer Science & Engineering

Jadavpur University

Prof. Chandan Mazumdar
Dean
Faculty of Engineering & Technology
Jadavpur University

Jadavpur University

Faculty of Engineering and Technology
Department of Computer Science & Engineering

CERTIFICATE

This is to certify that the project entitled **“Covid-19 Spatio-temporal Prediction Using Combined Graph Convolution - LSTM Model”** has been submitted by Santu Bera in partial fulfilment of the requirements for the award of the degree of **Master of Computer Application** in the department of **Computer Science & Engineering**, Jadavpur University, during the period 2019-2022 has been carried out under my supervision and that this work has not been submitted elsewhere for obtaining a degree.

EXAMINER:

------------------------------------ --------------------------------

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION OF ORIGINALITY
AND
COMPLIANCE OF ACADEMIC ETHICS

I hereby declare that this project contains original work by the undersigned candidate, as part of his Master of Computer Application (MCA) studies.

All information in this document have been obtained and presented in accordance with academic rules and ethical conduct.

I also declare that, as required by these rules and conduct, I have fully cited and referenced all material results that are not original to this work.

Name: Santu Bera
Roll No: 001910503014
Exam Roll No: MCA226014
Project Title: Covid-19 Spatio-temporal Prediction Using Combined Graph Convolution - LSTM Model.

Signature with date

ACKNOWLEDGEMENTS

On the submission of "Covid-19 Spatio-temporal Prediction Using Combined Graph Convolution - LSTM Model", I would like to convey my sincere gratitude to Dr. Anasua Sarkar, Assistant Professor, Department of Computer Science & Engineering, Jadavpur University for her valuable suggestions throughout the duration of the thesis work. I am really grateful to her for constant support which immensely helped me to fully involve myself in this work and develop new approaches in the field of Spatio-temporal Modelling.

Lastly, I would like to thank all my teachers, classmates, guardians and well-wishers for encouraging and co-operating me throughout the development of this thesis.

Regards,

SANTU BERA
Roll Number: 001910503014
Exam Roll No: MCA226014
Registration Number: 149877 of 2019-20

Signature with date

ABSTRACT

For the past two years, the world has been going through a tough time because of the global pandemic. The COVID-19 virus spread very quickly throughout the world and dismantled our daily lives, healthcare system, economy, and almost everything. At this point in time, we have lost almost 63 million human lives. From the very beginning of this pandemic, researchers have been trying to predict the nature of the COVID-19 virus and its impact on human life. But there was not sufficient data in the early days to analyze the effect of this deadly virus on a large scale. In spite of that, a lot of research work has been done to predict the future possibilities. Many machine-learning models have been proposed and successfully applied for predictions. Here we collected global data from COVID-19 till date, containing daily new cases, new deaths, cumulative cases, and cumulative deaths country-wise. Since there are several machine-learning algorithms that have been used to predict the COVID-19 virus, in this paper we propose a graph neural network model to make predictions on the number of new cases and deaths. In our model, we will utilize the distance between countries worldwide and try to represent the countries as a graph of structured data. Through our model and obtained predictions, our aim is to provide better knowledge about the spread of the COVID-19 virus and better preparation for any pandemic in the future.

CONTENTS

1 INTRODUCTION

1.1 BACKGROUND

The Coronavirus Disease (COVID-19) is a contagious disease caused by the SARS-CoV-2 virus. It was first identified in Wuhan, China in late 2019, but the outbreak has since spread to countries all over the world. The disease's global impact is enormous, with around 528 million cases and 63 million fatalities as of April 2022 [1]. COVID-19 is a highly infectious virus that spreads faster than influenza but slower than measles, one of the most contagious diseases known to humans [2].

To address the disease's rapid spread, nearly every country on the globe has implemented a number of preventative measures based on World Health Organization(WHO) standards. COVID-19 has been discovered to spread predominantly through human contact among people within 6 feet of one another. Therefore, the two main safeguards are social distancing and mask wearing, both of which try to restrict interpersonal encounters. Coughing, sneezing, talking, and other forms of exposure to an infected person's respiratory droplets cause this sort of transmission. Given the ease with which this potentially fatal disease can be transmitted, there is a clear need in society to manage and limit COVID-19's effects. As a result, the ability to foresee COVID-19's spread becomes a critical aspect in the world's ability to prepare a suitable reaction to the pandemic.

1.2 MOTIVATION

The vast majority of studies employing both epidemiological and machine learning models generate predictions on a macro scale, focusing on entire states and countries according to the current level of knowledge on COVID-19 transmission. During the early stages of the pandemic, however, numerous governments throughout the world imposed interstate and international travel restrictions, reducing the danger of infection from other countries.

When anticipating the spread of COVID-19 on a meso-scale, there are many restrictions that can be accommodated on a macro-scale. Each country has had a varied timeline for their COVID-19 outbreaks, as well as their responses to lock-downs and business shutdowns. Furthermore, depending on the severity of the outbreak, different countries imposed different rules, as countries like India is far more severely afflicted than others. Furthermore, with the establishment of travel limitations, communities' compliance with social distancing laws has a significant impact on COVID-19 transmission, as the disease is carried predominantly through interpersonal encounters. As a result, macro-scale modelling of COVID-19 should be able to better reflect the impacts of social distancing when generating predictions. Due to the scarcity of COVID-19 case count data for particular towns and cities compared to states and countries, time-series forecasting at the micro-scale is regarded as a worthwhile undertaking for this study.

1.3 LITERATURE SURVEY

Before beginning the research, it is necessary to first review previous works in the field in order to gain a baseline grasp of the present state of knowledge on this subject. The latest approaches in research on COVID-19 time-series forecasting are summarised in this literature review. The present existing epidemiology models and the current existing machine learning models are the two main areas of attention in this exploratory research, with the goal of being able to identify a potential research need and eventually contribute to this field of study with our own work.

1.3.1 EPIDEMIOLOGICAL MODELS

In the field of disease research, epidemiological models are frequently used to assess infectious disease growth trends and forecast epidemic consequences. The Susceptible-Infected-Removed (SIR) model, for example, calculates the theoretical number of people infected with a certain disease in a closed community over time. The numbers of susceptible (S), infected (I), and recovered (R) people are used to predict the spread of COVID-19 in the early phases of the pandemic in China, as well as in other countries such as South Korea, India, Australia, the United States, and Italy [3] [4].

Other studies have used the Susceptible-Exposed-Infected-Removed (SEIR) paradigm to extend this model even further. This model is similar to the SIR model, except it also considers the exposed (E) population. The SEIR model has also been used to forecast the COVID-19 spread in countries such as China and Italy [5] [6]. In the presence of infection hotspots, the SEIR model has also been used to estimate the effects of contact tracing, testing, and containment strategies [7]. Overall, the researchers are able to offer positive results and forecasts for the countries analysed.

1.3.2 Machine Learning Models

Multiple models have been shown to efficiently forecast the spread of COVID-19 in the field of machine learning. There are numerous works focusing on the task of forecasting the spread of COVID-19 within specific countries using machine learning, ranging from classical models such as the Auto-Regressive Integrated Moving Average (ARIMA) model to more robust deep learning models such as Multilayer Perceptron (MLP) and Long Short-Term Memory (LSTM) [8] [9].

Furthermore, researchers have conducted comparative studies in which they observed and compared the predictions of numerous distinct machine learning models at the same time. Tian et al. [10]compare the results from simple RNN, LSTM, Bidirectional LSTM, gated recurrent unit (GRU), and variational autoencoder (VAE) algorithms, while Zeroual [11]et al. compare the results from simple RNN, LSTM, Bidirectional LSTM, gated recurrent unit (GRU), and variational autoencoder (VAE) algorithms.

Because of their capacity to capture spatial relations inside their models, Graph Neural Networks are becoming more popular due to the nature of this specific time-series forecasting problem. This is important for COVID-19 since the disease's impact differs based on the observed area due to differences in population, health legislation, and other factors. A spatio-temporal graph neural network is[12] used to predict COVID-19 case counts at the county level for the next day, and a Spatio-Temporal Attention Network (STAN) [12]was used to capture geographic and temporal trends to predict COVID-19 case counts at the county level for a fixed period of time in the future [13]. To aid in prediction, these two graph neural networks included mobility data in their models.

1.4 Objective

Using country-level COVID-19 infection data from around the world, we investigate two macro-scale machine learning models using a variety of techniques and architectures that can predict COVID-19 transmission on a macro-scale.

Two neural network models make up our models. These are the Graph Convolutional Network–Long Short Term Memory (GCN-LSTM) and Long Short-Term Memory (LSTM) models, respectively. These will be examined in further depth in the paper's methodology chapter. In addition, the test results and analysis of each of these models will be detailed in the results of the paper and discussion sections.

2 METHODOLOGY

2.1 NEURAL NETWORKS

The neural networks are scientifically inspired by neurons in the brain, which aim to build a map of inputs and outputs comparable to how the brain reacts to various sensory inputs. Neural networks are a type of computer network. This mapping can be learned by mixing forward and backward passes across. There are several input-output pairs. Deep learning models, such as the ones utilised in this study, Feedforward neural networks (FNN) are a key component in this thesis. The FNN tries to optimize a function f that maps an input x to an output y. It does so by feeding information forward through all the inputs until it reaches y, hence the name. A typical FNN can be depicted by the following formula:

$$f(x) = f_1\,(f_2(\ldots\ldots f_{n-1}(f_n)))\;\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \;(1)$$

where f_i is the ith layer in the neural network containing n layers. Each layer in the neural network contains a set of nodes that takes in the output of previous layers as its input. These outputs are then multiplied by a series of weights, which the network will attempt to learn in approximating the target function [14].

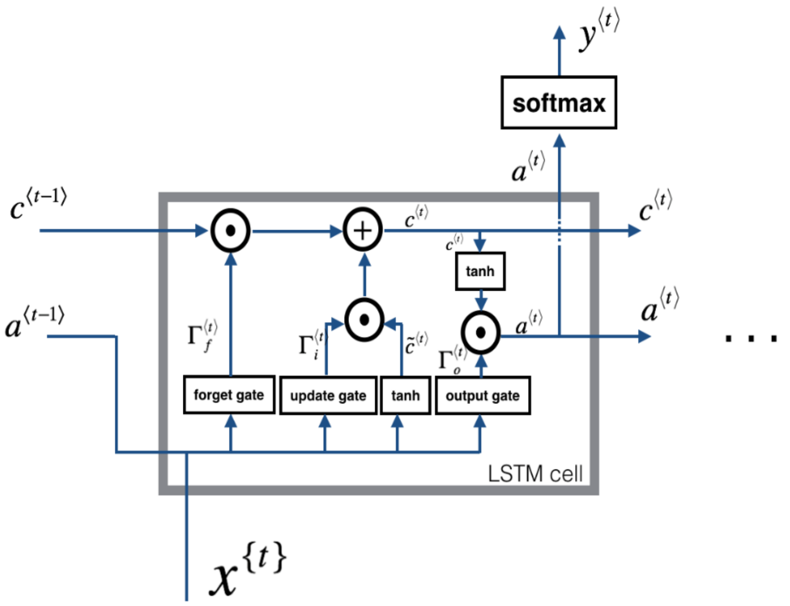
2.1.1 TIME-SERIES FORECASTING MODELS

The problem we're working on is a time-series forecasting challenge. Time-series forecasting, like other machine learning applications, is basically centred on making predictions, with the primary difference being the type of data employed. While most machine learning models reward past observations with roughly equal weights when making predictions, time-series forecasting adds an explicit order dependence between data as an essential feature. Models are fitted to previous data and used to predict future observations in the forecasting process [15]. Because the model has no way of knowing what will happen in the future, all estimates are based on what has already happened. Time-series forecasting has many applications in tracking the spread of diseases and serves as the basis of our machine learning models, the backgrounds of which are outlined below.

2.1.2 LONG SHORT-TERM MEMORY (LSTM)

Due to its widespread use in a variety of domains, the Long Short-Term Memory (LSTM) model is chosen as the first model to be tested as a baseline for this thesis. As its name implies, LSTM is a form of Recurrent Neural Network (RNN) that can better manage long-term memory, and it is a widely used deep learning model for problems involving time-series data. A cell, input gate, output gate, and forget gate are common components of an LSTM unit. The cell remembers values across arbitrary time intervals, and the three gates control the flow of data into and out of it. The LSTM model uses its forget gates to remove bias toward recent events, but the RNN model problems with being biased toward more recent events because to its poor long-term memory.

An LSTM unit will take the previous cell state, which is usually in the form of a vector, and execute different activations, multiplications, and concatenations on it. The forget gate then evaluates how much of the current cell state and input should be passed on and discards the rest [16]. An LSTM cell with its many gates is depicted in the diagram below.



$$\Gamma_f^{\langle t\rangle} = \sigma(W_f[a^{\langle t-1\rangle}, x^{\langle t\rangle}] + b_f)$$
$$\Gamma_u^{\langle t\rangle} = \sigma(W_u[a^{\langle t-1\rangle}, x^{\langle t\rangle}] + b_u)$$
$$\tilde{c}^{\langle t\rangle} = \tanh(W_C[a^{\langle t-1\rangle}, x^{\langle t\rangle}] + b_C)$$
$$c^{\langle t\rangle} = \Gamma_f^{\langle t\rangle} \circ c^{\langle t-1\rangle} + \Gamma_u^{\langle t\rangle} \circ \tilde{c}^{\langle t\rangle}$$
$$\Gamma_o^{\langle t\rangle} = \sigma(W_o[a^{\langle t-1\rangle}, x^{\langle t\rangle}] + b_o)$$
$$a^{\langle t\rangle} = \Gamma_o^{\langle t\rangle} \circ \tanh(c^{\langle t\rangle})$$

Figure 1. LSTM cell used for classification problems [17]

2.2 ACTIVATION FUNCTION

The activation function creates a weighted total and then adds bias to it to determine whether a neuron should be activated or not. We know that the weight, bias, and activation function of neurons in a neural network affect how they work. Based on the output inaccuracy, we would adjust the weights and biases of the neurons in a neural network. This process is known as back-propagation. Activation functions allow back-propagation because the gradients and error are provided at the same time to update the weights and biases. A neural network is nothing more than a linear regression model without an activation function. The activation function is used to deal with non-linearity.

2.2.1 RELU

The Rectified Linear Unit (ReLU) activation function is given by

$$f(x) = \max(0, x) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2)$$

It has a range of $[0, inf]$. It's the most used activation method. Hidden layers of neural networks are primarily used. Because it includes fewer mathematical calculations, ReLu is less computationally expensive than tanh and sigmoid. Only a few neurons are active at a time, making the network sparse and efficient for computation. Simply put, the RELU function learns much more quickly than the sigmoid and Tanh functions.

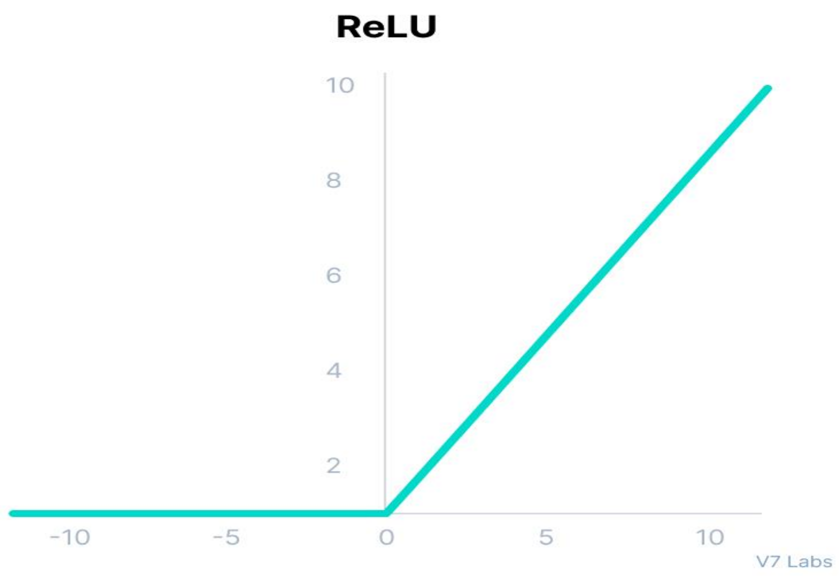


Figure 2. ReLU Activation Function [18]

2.2.2 SIGMOID FUNCTION

The Sigmoid activation function is given by

$$f(x) = \frac{1}{1+e^{-x}} \qquad (3)$$

It has a range of $[0, 1]$. Another widely used activation function that performs well in classification issues is the sigmoid function. One problem with this activation function is that it has a vanishing gradient problem, which means that when the activation approaches the horizontal portion of the curve, the gradient becomes exceedingly small and the activation fails to learn [14]. Typically employed in the output layer of a binary classification, where the result is either 0 or 1. Because the sigmoid function's value is only between 0 and 1, the result can be easily anticipated to be 1 if the value is greater than 0.5, and 0 otherwise.

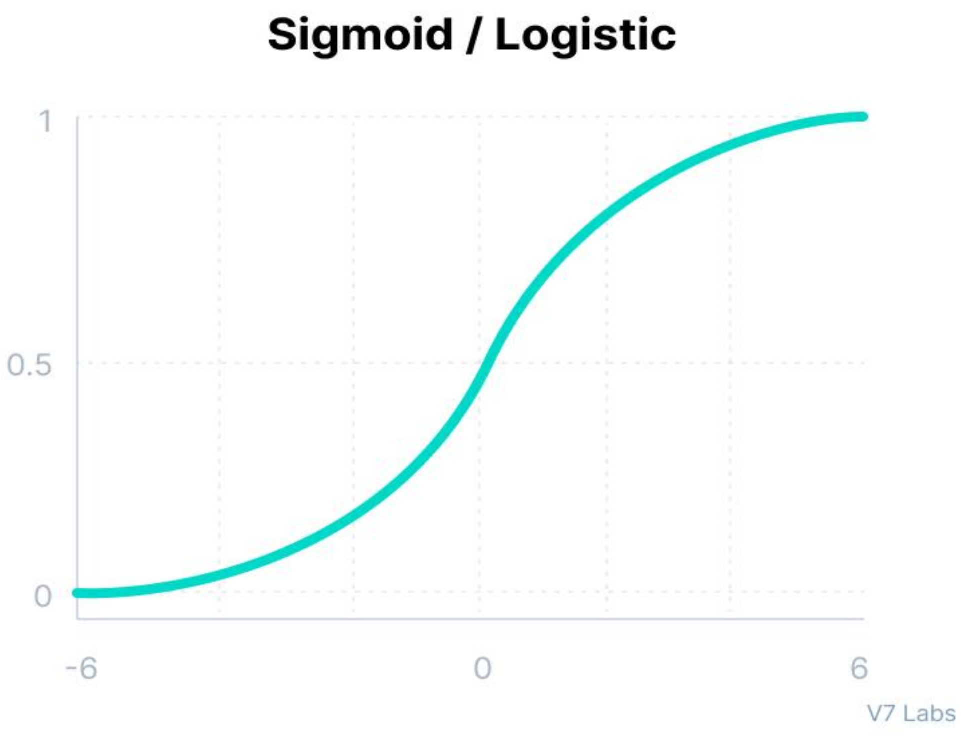


Figure 3. Sigmoid Activation Function[18]

2.3 Loss Function

The loss function indicates how accurate the expected outcome prediction is. This is accomplished by using several comparison functions to compare the expected and observed outputs of the neural network. There are many different loss functions, just like there are many different activation functions, and the choice is based on the sort of problem the neural network is solving.

2.3.1 MSE (MEAN SQUARE ERROR)

Mean squared error (MSE) measures the amount of error in statistical models. It assesses the average squared difference between the observed and predicted values. A higher MSE shows that the data points are widely spread around the central moment (mean), whereas a lower MSE indicates the reverse. A smaller MSE is preferable since it implies that your data points are evenly distributed around the centre moment (mean). It shows your data's concentrated distribution, the fact that it is not skewed, and, most crucially, that it has fewer errors (measured by the dispersion of the data points from its mean).

The smaller the MSE, the smaller the error, and the better the estimator [14].

The Mean Squared Error is calculated as:

$$MSE = \frac{1}{n}\sum(actual - forcast)^2 \text{ ……………………. } (4)$$

Where, ∑ – a symbol that means "sum", n – sample size, actual – the actual data value, forecast – the predicted data value.

The activation function utilised in this model was the Mean Squared Error (MSE), which is a regularly used loss function for regression problems, due to the nature of the goal of forecasting the spread of COVID-19. The sum of the squared distance between the predicted and expected values is calculated using this loss function. Outliers have a significant impact on this loss function since the mistakes are squared.

2.4 Optimization Algorithms

During backpropagation, optimization methods are employed to update the weights of a neural network. Gradient descent is the most often used optimization algorithm in deep learning. It is a method for gradually changing the values of a function's parameters so that the loss function is minimised as much as feasible. The learning rate determines how quickly these settings are updated. The stochastic gradient descent method is a version of gradient descent that we employ in our models. Gradient descent in this method updates the network's weights after each training sample, resulting in a speedier training process. One of the most difficult aspects of gradient descent is determining the appropriate learning rate. A rate that is too low will not converge, while a rate that is too large would cause too many variations, slowing down the learning process.

2.4.1 Adam Optimizer

Other methods have emerged to handle the difficulty of choosing the proper learning rate, and they are now routinely used in deep learning problems. In this research, the Adam optimizer is employed in the implementation of two deep learning models. Adam optimization is a type of stochastic gradient descent that uses adaptive first-order and second-order moment estimation. It is computationally efficient and suitable for the majority of deep learning situations [19].

3 Proposed Model

Graph Convolutional Network–Long Short-Term Memory (GCN-LSTM)

Graph Neural Networks (GNN) have lately gained popularity as a result of their capacity to function on a graph structure, which allows them to be used in fields such as social networks, recommender systems, and transportation systems. Modern machine learning methods can be applied to graphs to generate predictive models of connected data using certain modifications, such as the Graph Convolutional Network (GCN), employed in our research.

Graph

Vertices (or nodes) and edges are the two components of the graph data structure. Individual entities inside the data, such as members of a social network or, in the case of our study, countries throughout the world, are represented by vertices. Edges reflect the connections between nodes, which in a social network could be friends, co-workers, and so on. Graphs can be either directed or undirected, depending on whether or not the nodes to be connected have directional dependencies.

Two friends would be connected by an undirected edge in a graph depicting a social network, as friendship is a mutually shared relationship between the two entities. To provide extra meaning to the data structure, both nodes and edges can be associated with specific attributes. Returning to the social media example, each node representing a person might also have information such as their name, age, location, and so on, while the edge could indicate the strength of a friendship or the length of time two people have been connected on the network. Graph Neural Networks rely heavily on the ability to assign specific properties to graphs.

Because our research focuses on countries all over the world, it was necessary to first depict the world map as a network in order to employ the GCN-LSTM model for machine learning. The world is depicted in the diagram below as a graph structure, with each of the 236 nodes representing a country in the world and each edge representing the distance between those countries. It is an undirected graph that depicts the freedom of movement in any direction between any two countries.

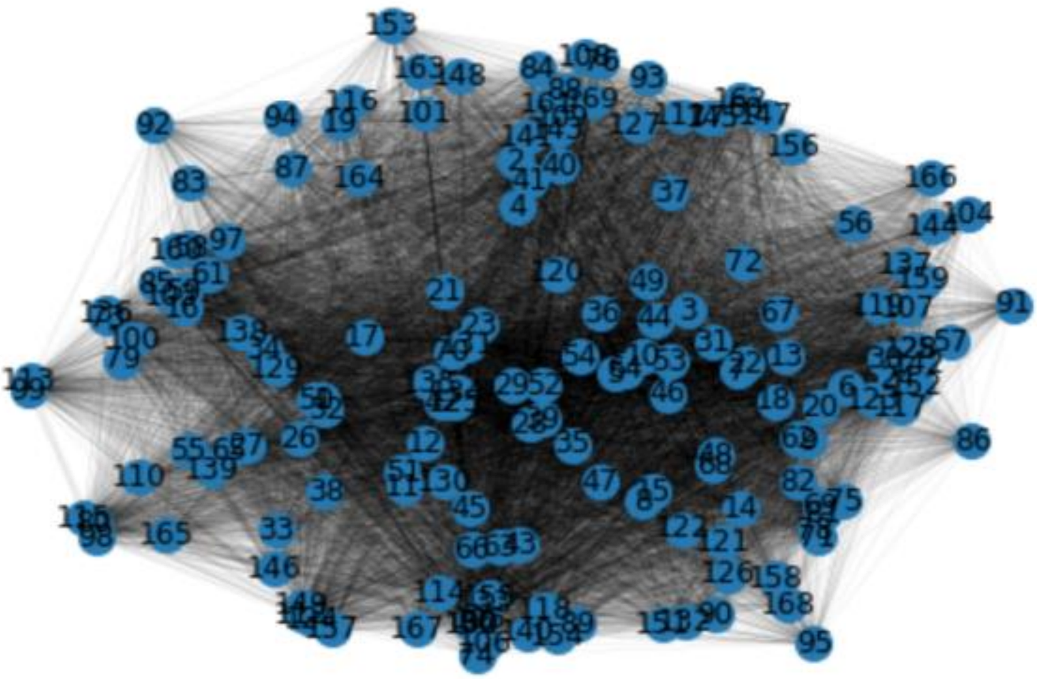


Figure 4. Graph structure showing the connections between the countries worldwide [20]

GCN-LSTM

Graph Convolutional Networks (GCN) [21]can be used to perform machine learning while incorporating network structures as graphs. Node classification is a frequent use of GCNs, in which node properties can be predicted using the graph's relational information. The graph convolutional layer is a newer form of neural network layer used by GCNs, and its architecture is represented in Figure 6. The weight matrix **W** and bias vector **b**, the node features matrix **F**, and the normalised graph adjacency matrix **A'** are the four key trainable parameters in the graph convolutional layer. The graph is encoded into the network by the normalised adjacency matrix **A'**, which is normalised so that each neighbouring node's contribution is proportional to how linked the node is to the rest of the graph. Then, an element-wise non-linear function such as ReLU is applied to the weights. The resulting output matrix can then be used as the input to the next graph convolutional layer.

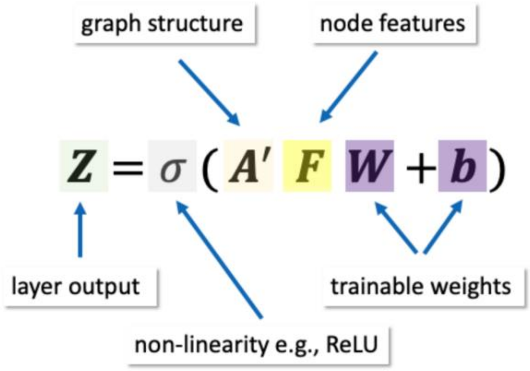


Figure 5: Components of Graph Convolutional layer [21]

The graph convolutional layer contains four main trainable parameters: the weight matrix W and bias vector b, the node features matrix F as its inputs, and the normalized graph adjacency matrix A'. The layer output Z can be used as the input of the next graph convolutional layer.

$$V_t = \sigma(W_v \, [f(A,\, X_t),\; h_{t-1}] + b_v) \ldots\ldots\ldots\ldots\ldots\ldots\ldots (5)$$

$$r_t = \sigma(W_r \, [f(A,\, X_t),\, h_{t-1}] + \; b_r) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (6)$$

$$c_t = ReLU(W_c \, [f(A, X_t), (r_t * \; h_{t-1})] + \; b_c) \ldots\ldots\ldots (7)$$

$$h_t = \; v_t * \; h_{t-1} + (1 - v_t) * c_t \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (8)$$

Where h_{t-1} represents the hidden state at time t-1; x_t represents traffic information at time t; and r_t represents the reset gate, which is used to determine the degree of ignoring the preceding moment's status information. v_t stands for the update gate, which controls the degree to which prior status information is incorporated into the present status; c_t stands for the memory content recorded at time t; and $h_t$stands for the output state at time t.

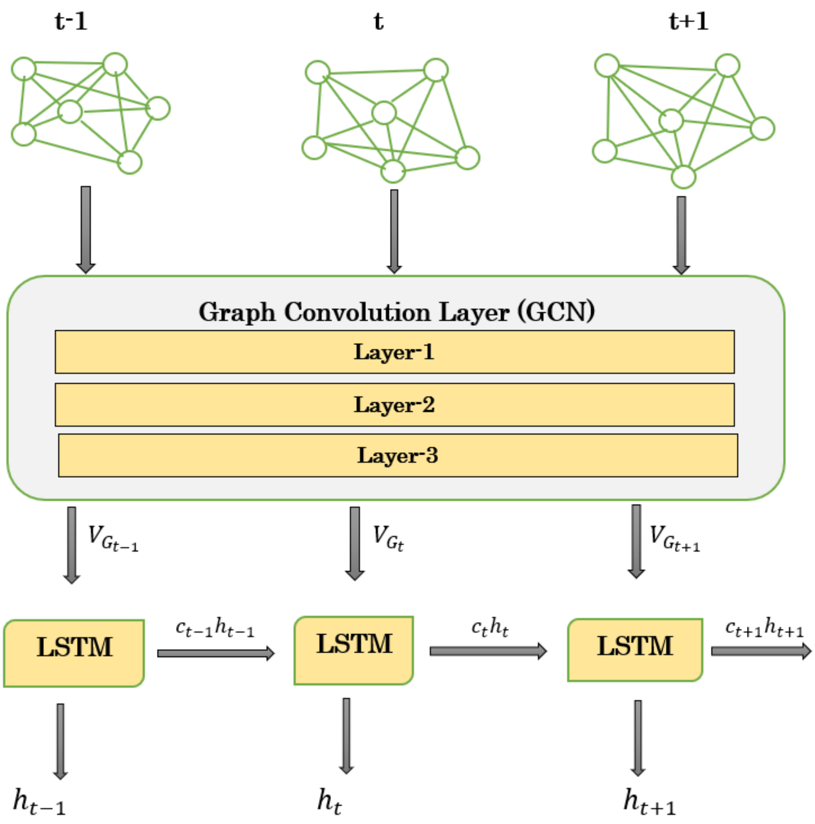


Figure 6: Architecture of GCN-LSTM Model

4 Dataset

As our goal is to predict the spread of COVID-19, the dataset is one of the important key aspects of our work. Here we are using two datasets. The first one contains all the country names, capitals, and latitude and longitude of the respective capitals. In the dataset, there are a total of 236 countries. The second dataset consists of COVID-19 case counts, death counts, and cumulative numbers of cases and deaths every day from January 3rd, 2020 to February 4th, 2022. The COVID-19 datasets are obtained from the official website of the World Health Organization (WHO) [22].

	CountryName	CapitalName	CapitalLatitude	CapitalLongitude
0	Afghanistan	Kabul	34.516667	69.183333
1	Albania	Tirana	41.316667	19.816667
2	Algeria	Algiers	36.750000	3.050000
3	American Samoa	Pago Pago	-14.266667	-170.700000
4	Andorra	Andorra la Vella	42.500000	1.516667

Figure 7: Dataset containing country capitals and their latitude and longitude respectively.

	Date_reported	Country_code	Country	WHO_region	New_cases	Cumulative_cases	New_deaths	Cumulative_deaths
0	2020-01-03	AF	Afghanistan	EMRO	0	0	0	0
1	2020-01-04	AF	Afghanistan	EMRO	0	0	0	0
2	2020-01-05	AF	Afghanistan	EMRO	0	0	0	0
3	2020-01-06	AF	Afghanistan	EMRO	0	0	0	0
4	2020-01-07	AF	Afghanistan	EMRO	0	0	0	0

Figure 8:Dataset containing daily and cumulative no of cases and deaths per country

5 Data Pre-processing

5.1.1 DATA NORMALIZATION (MIN-MAX SCALING)

While the model employed the basic COVID-19 data set with case counts, the GCN-LSTM model had extra data requirements due to its more complicated architecture. As previously stated, this particular model also includes graph data, as proposed in the model chapter. Figure 4 shows a network structure to represent all the countries in the whole world. The spatial properties of the entire universe are captured by modelling the graph as an adjacency matrix. Each node in the GCN-LSTM model's network represented a distinct country, and each edge represents distances between the country capitals. The distances between each country and the other nodes are represented. This essentially creates a map of the entire environment that the model can use as a variable. The adjacency matrix is also normalised so that the contribution of each neighbouring node is proportionate to how linked the node was to the rest of the graph. The following equation iss used to do this:

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} \qquad (9)$$

All edge distances are rescaled to fall within the range of [0, 1] when the equation is applied to the adjacency matrix. After all of the changes, the final adjacency matrix had a 236 X 236 matrix for each relevant country around the world, totalling 55,696 total distances, with the distance between each country and the remaining 235 countries being recorded. The GCN-LSTM model can learn a lot of spatial information because of this.

0.0	0.21746041188480902	0.29376190710294475	0.6841523582652514	0.2907384696844293	0.3811817716580969	0.615663281892647
0.21746041188480902	0.0	0.0767991264185626	0.8444059431867978	0.07607148336827496	0.2816553065303924	0.40614221343753426
0.29376190710294475	0.0767991264185626	0.0	0.8744418538597695	0.03272699638561722	0.2596661196702229	0.3362295675439572
0.6841523582652514	0.8444059431867978	0.8744418538597695	0.0	0.8417941677942601	0.8729954954338338	0.6181320386542164
0.2907384696844293	0.07607148336827496	0.03272699638561722	0.8417941677942601	0.0	0.2923580531765443	0.3301039152429073
0.3811817716580969	0.2816553065303924	0.2596661196702229	0.8729954954338338	0.2923580531765443	0.0	0.4456625164999988
0.615663281892647	0.40614221343753426	0.3362295675439572	0.6181320386542164	0.3301039152429073	0.4456625164999988	0.0
0.6160948688048625	0.4050173264172429	0.3341884604685624	0.6234918632385285	0.3290816293942197	0.4380734981633199	0.008846212701134129
0.7669217389149706	0.5829447542021549	0.5104382365625902	0.5529572386893464	0.5279128754925229	0.3909301178794216	0.29530574048966574
0.11343964587168331	0.10411680907801357	0.18032862719192733	0.7796302947369648	0.17849159513962692	0.31718356088198135	0.5073075649156843
0.6640202797972563	0.45531347370412995	0.3850858587145928	0.5754565505096386	0.37927715239923815	0.4761832895705805	0.04917329096158114

Figure 9: Adjacency matrix of distances between the capitals of all countries.

Now, as we are working with number of new daily COVID-19 cases, we process the dataset containing case and death counts. We process the data in a format where there are 236 columns for the countries denote the number of daily new cases and 704 rows denoting the dates from March 3rd, 2020 to February 4th, 2022. After processing we get the data as follows:

	2020-03-03	2020-03-04	2020-03-05	2020-03-06	2020-03-07	2020-03-08	2020-03-09	2020-03-10	2020-03-11	2020-03-12	...	2022-01-26	2022-01-27	2022-01-28	2022-01-29	2022-01-30	2022-01-31	2022-02-01	2022-02-02	2022-02-03	2022-02-04
0	0	0	0	0	3	0	0	0	3	0	...	356	456	301	49	608	474	610	970	470	537
1	0	0	0	0	0	0	2	0	0	1	...	2156	1562	1549	1615	1341	939	522	1533	1164	0
2	2	3	4	5	0	0	3	0	0	5	...	2521	2162	2130	1870	1742	1464	1343	1403	1365	0
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
4	1	0	0	0	0	0	0	0	0	0	...	1676	327	0	528	0	0	402	0	357	155

Figure 10: Covid-19 dataset after processing.

6 EXPERIMENTS

6.1 MODEL ARCHITECTURE

The architecture of our Graph Convolutional Network–Long Short-Term Memory (GCN-LSTM) model is shown in Figure 11. The StellarGraph Machine Learning Library, which is written in Python, was used to create this model. And all the model is implemented on Google colab online platform. StellarGraph's Temporal Graph Convolutional Network (T-GCN) [23]architecture inspired the GCN-LSTM model. The graph convolutional network (GCN), the long short-term memory (LSTM) model, and a dropout and dense layer make the model.

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 236, 5)]          0

 tf.expand_dims (TFOpLambda)  (None, 236, 5, 1)        0

 reshape (Reshape)           (None, 236, 5)            0

 fixed_adjacency_graph_convo  (None, 236, 12)          55992
 lution (FixedAdjacencyGraph
 Convolution)

 fixed_adjacency_graph_convo  (None, 236, 12)          56076
 lution_1 (FixedAdjacencyGra
 phConvolution)

 fixed_adjacency_graph_convo  (None, 236, 10)          56052
 lution_2 (FixedAdjacencyGra
 phConvolution)

 reshape_1 (Reshape)         (None, 236, 10, 1)        0

 permute (Permute)           (None, 10, 236, 1)        0

 reshape_2 (Reshape)         (None, 10, 236)           0

 lstm (LSTM)                 (None, 10, 12)            11952

 lstm_1 (LSTM)               (None, 10, 12)            1200

 lstm_2 (LSTM)               (None, 12)                1200

 dropout (Dropout)           (None, 12)                0

 dense (Dense)               (None, 236)               3068

=================================================================
Total params: 185,540
Trainable params: 18,452
Non-trainable params: 167,088
```

Figure 11: GCN-LSTM model summary

6.2 Training and Testing

Several critical parts of the training procedure had to be examined with the shared goal of creating more accurate COVID-19 transmission predictions in the world. All three models used 80% of the data as their training set and the remaining 20% as their prediction test set for the predicted time frame. The test data will be predicted for about 5 months as a result of this. A total of 563 days are employed as the training period, with the remaining 146 days serving as the prediction period.

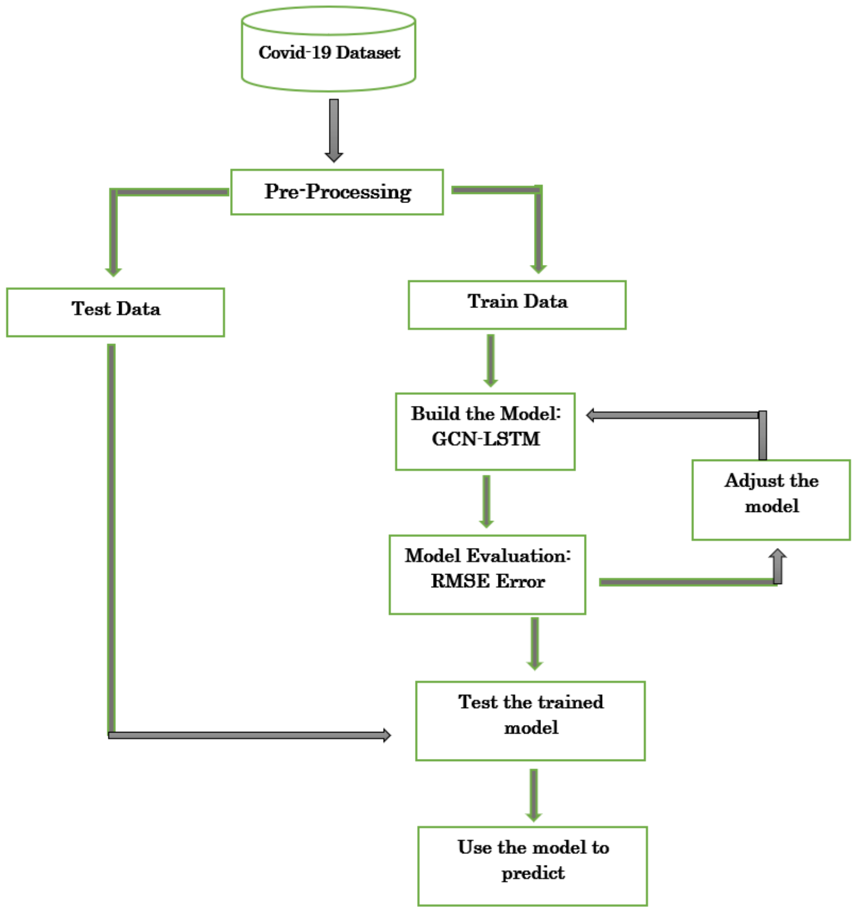


Figure 12: Work-flow of GCN-LSTM Model

6.3 Hyperparameters

Hyperparameter adjustment is an important part of each model's training process. A hyperparameter is a parameter, usually in the form of a variable, whose value is used to impact the learning process in machine learning. Different hyperparameters are required depending on the type of machine learning model utilised, whether deep learning or classical machine learning. The hyperparameters employed in our work's model are described in this section. Although many different hyperparameter combinations are tested, the following discussion will focus on the ones that produced the best outcomes in our model research.

The more sophisticated architecture of the GCN-LSTM model necessitate the usage of more hyperparameters. The sequence length, prediction length, graph convolutional activation function, lstm activation function, optimizer, learning rate, loss function, epochs, and batch size are the hyperparameters as listed in Figure 13.

The sequence length in this model refers to the number of prior 5 days that will be utilised to predict the next additional day, which is defined by the prediction length. As a result, the GCN-LSTM model uses the previous five days to forecast the following one. The ReLU and Sigmoid activation functions for the GCN and LSTM networks, respectively, are used in this model. The Adam optimizer is used in the GCN-LSTM model, with a default learning rate of 0.001. The Loss Function is set to Mean Squared Error because it is the statistical function that we are trying to reduce in our experiments. The number of epochs is set to 100, and the amount required to make the loss curve converge is determined. Finally, the stochastic gradient descent algorithm's batch size is set to 10.

Hyperparameter	**Value**
Sequence Length	5
Prediction Length	1
GC Activation Function	ReLU
LSTM Activation Function	Sigmoid
Optimizer	Adam
Learning Rate	0.001
Loss Function	MSE
Epochs	100
Batch Size	10

Figure 13: Hyperparameters of GCN-LSTM Model

7 RESULTS

7.1 RUNTIME

The runtime is another crucial part of the training process. Although the training process's duration is highly dependent on hardware and processing speeds, training times using a personal laptop are provided as a guide. The personal computer used for this experiment has the following specification: processor: Intel(R) Core(TM) i3-7020U CPU @ 2.30GHz 2.30 GHz; RAM size: 4.00 GB; System type: 64-bit operating system, x64-based processor.

Runtimes are critical to assess because they impact how soon a model can estimate case counts and how much time communities have to prepare in the event of a pandemic. For this reason, the model with the shorter runtime would be favoured between two models with identical accuracies. When comparing the model's results, this is a crucial consideration to keep in mind. For our GCN-LSTM model the runtime on a personal laptop is given as:

$$Runtime = 2m\ 27s$$

$$= 147\ seconds$$

The GCN-LSTM is by far the most sophisticated model, combining a neural network with a graph data structure, and it takes the longest to train. These runtimes and complexity, on the other hand, may not always have an impact on the model's overall performance, which will be examined in the next chapter.

7.2 MODEL PERFORMANCE

As discussed previously, we are using the MSE as our loss function. Using the country-wise daily COVID-19 data, we are able to predict the number of new cases over the whole world. Using COVID-19 case counts from March 3rd, 2020 to September 17th, 2021 as training data, our model predicts disease transmission on test data from September 18th, 2021 to February 4th, 2022. We use the Adam optimizer in our model, as previously mentioned. When we are training the model with COVID-19 data, we also calculate the training and testing accuracy of the model. For 100 epochs, the model calculated the MSE error over train and test data. After calculation, every time it readjusts the model to optimise the loss, and so on. The MSE errors and train and test accuracy of last 15 epochs are shown in the Figure 14.

```
Epoch 85/100
56/56- 1s 23ms/step - loss: 0.0017 - MSE: 0.0017 - accuracy: 0.9373 - val_loss: 0.0038 - val_MSE: 0.0038 - val_accuracy: 0.8582
Epoch 86/100
56/56- 1s 22ms/step - loss: 0.0018 - MSE: 0.0018 - accuracy: 0.9320 - val_loss: 0.0038 - val_MSE: 0.0038 - val_accuracy: 0.8582
Epoch 87/100
56/56- 1s 22ms/step - loss: 0.0019 - MSE: 0.0019 - accuracy: 0.9491 - val_loss: 0.0038 - val_MSE: 0.0038 - val_accuracy: 0.8582
Epoch 88/100
56/56- 1s 22ms/step - loss: 0.0015 - MSE: 0.0015 - accuracy: 0.9391 - val_loss: 0.0038 - val_MSE: 0.0038 - val_accuracy: 0.8582
Epoch 89/100
56/56- 1s 22ms/step - loss: 0.0015 - MSE: 0.0015 - accuracy: 0.9320 - val_loss: 0.0037 - val_MSE: 0.0037 - val_accuracy: 0.8582
Epoch 90/100
56/56- 1s 23ms/step - loss: 0.0015 - MSE: 0.0015 - accuracy: 0.9373 - val_loss: 0.0037 - val_MSE: 0.0037 - val_accuracy: 0.8582
Epoch 91/100
56/56- 1s 22ms/step - loss: 0.0015 - MSE: 0.0015 - accuracy: 0.9391 - val_loss: 0.0037 - val_MSE: 0.0037 - val_accuracy: 0.8582
Epoch 92/100
56/56- 1s 22ms/step - loss: 0.0014 - MSE: 0.0014 - accuracy: 0.9391 - val_loss: 0.0036 - val_MSE: 0.0036 - val_accuracy: 0.8582
Epoch 93/100
56/56- 1s 22ms/step - loss: 0.0016 - MSE: 0.0016 - accuracy: 0.9473 - val_loss: 0.0037 - val_MSE: 0.0037 - val_accuracy: 0.8582
Epoch 94/100
56/56- 1s 23ms/step - loss: 0.0015 - MSE: 0.0015 - accuracy: 0.9356 - val_loss: 0.0036 - val_MSE: 0.0036 - val_accuracy: 0.8582
Epoch 95/100
56/56- 1s 22ms/step - loss: 0.0013 - MSE: 0.0013 - accuracy: 0.9373 - val_loss: 0.0036 - val_MSE: 0.0036 - val_accuracy: 0.8582
Epoch 96/100
56/56- 1s 22ms/step - loss: 0.0014 - MSE: 0.0014 - accuracy: 0.9356 - val_loss: 0.0037 - val_MSE: 0.0037 - val_accuracy: 0.8582
Epoch 97/100
56/56- 1s 23ms/step - loss: 0.0015 - MSE: 0.0015 - accuracy: 0.9358 - val_loss: 0.0037 - val_MSE: 0.0037 - val_accuracy: 0.8582
Epoch 98/100
56/56- 1s 23ms/step - loss: 0.0015 - MSE: 0.0015 - accuracy: 0.9320 - val_loss: 0.0037 - val_MSE: 0.0037 - val_accuracy: 0.8582
Epoch 99/100
56/56- 1s 22ms/step - loss: 0.0015 - MSE: 0.0015 - accuracy: 0.9391 - val_loss: 0.0037 - val_MSE: 0.0037 - val_accuracy: 0.8582
Epoch 100/100
56/56- 1s 22ms/step - loss: 0.0014 - MSE: 0.0014 - accuracy: 0.9320 - val_loss: 0.0036 - val_MSE: 0.0036 - val_accuracy: 0.8582
```

Figure 14: Training & testing accuracies and MSE errors

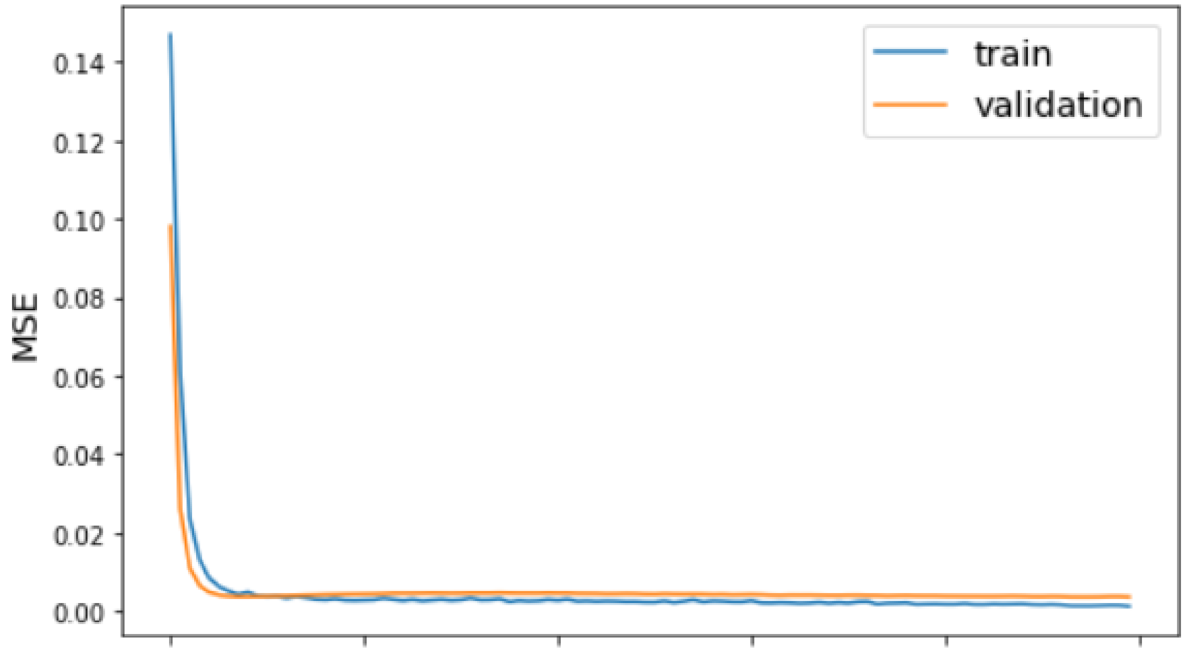


Figure 15: Comparison of MSE between train and test data for new cases

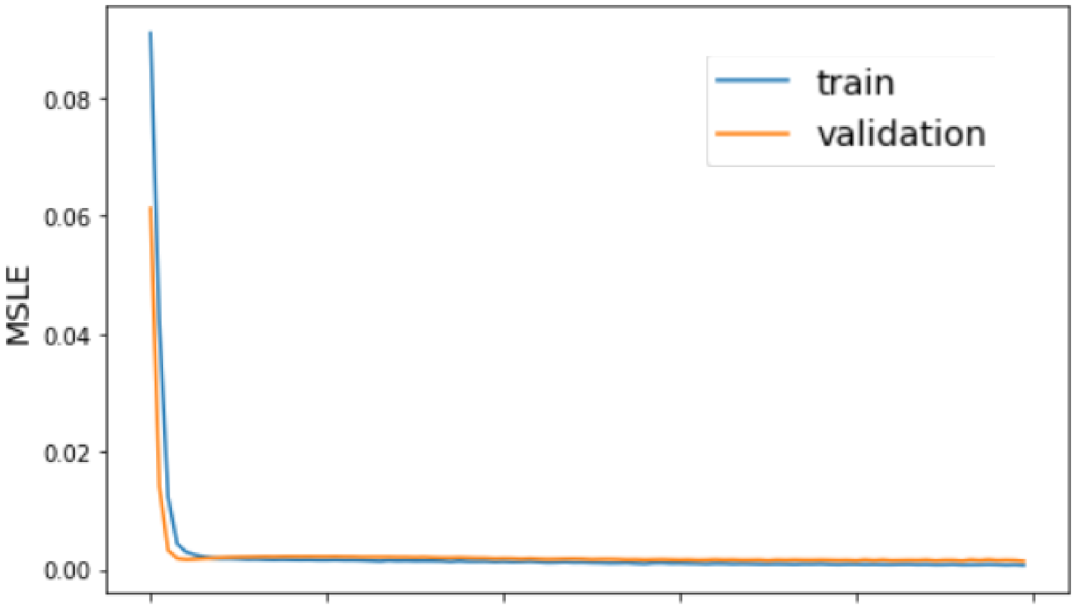


Figure 16: Comparison of MLSE between train and test data for new cases

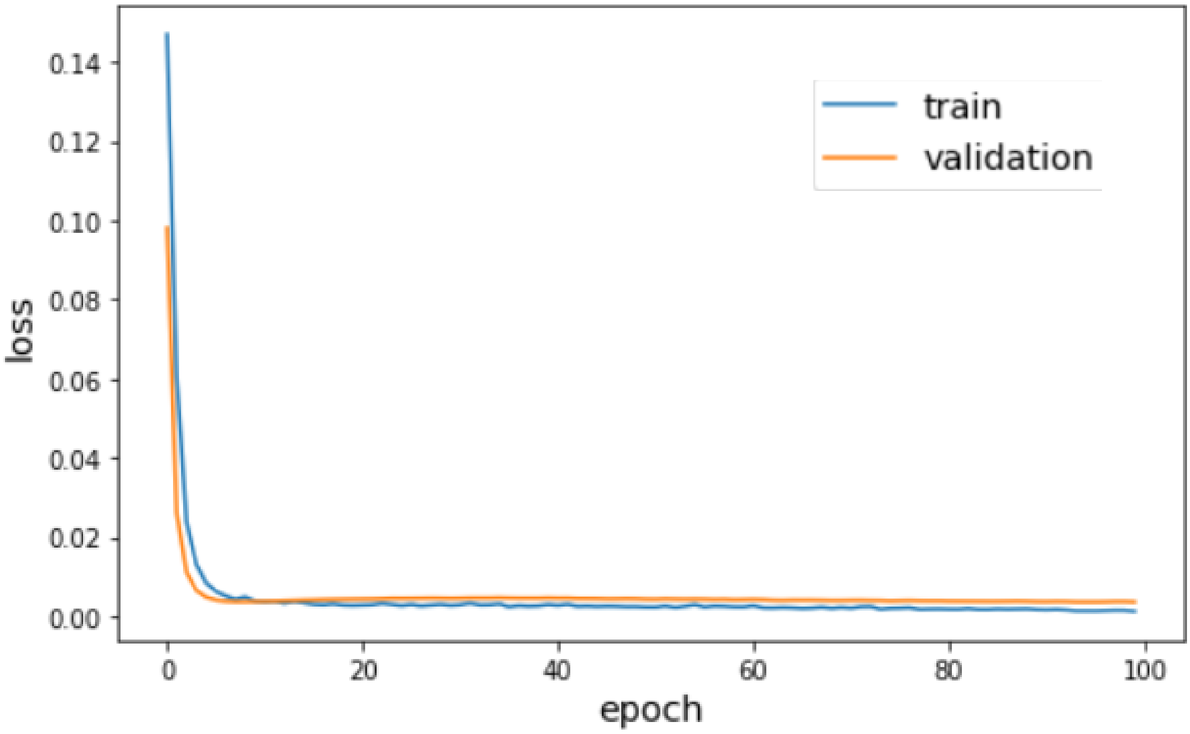


Figure 17: Comparison of loss between train and test data for new cases

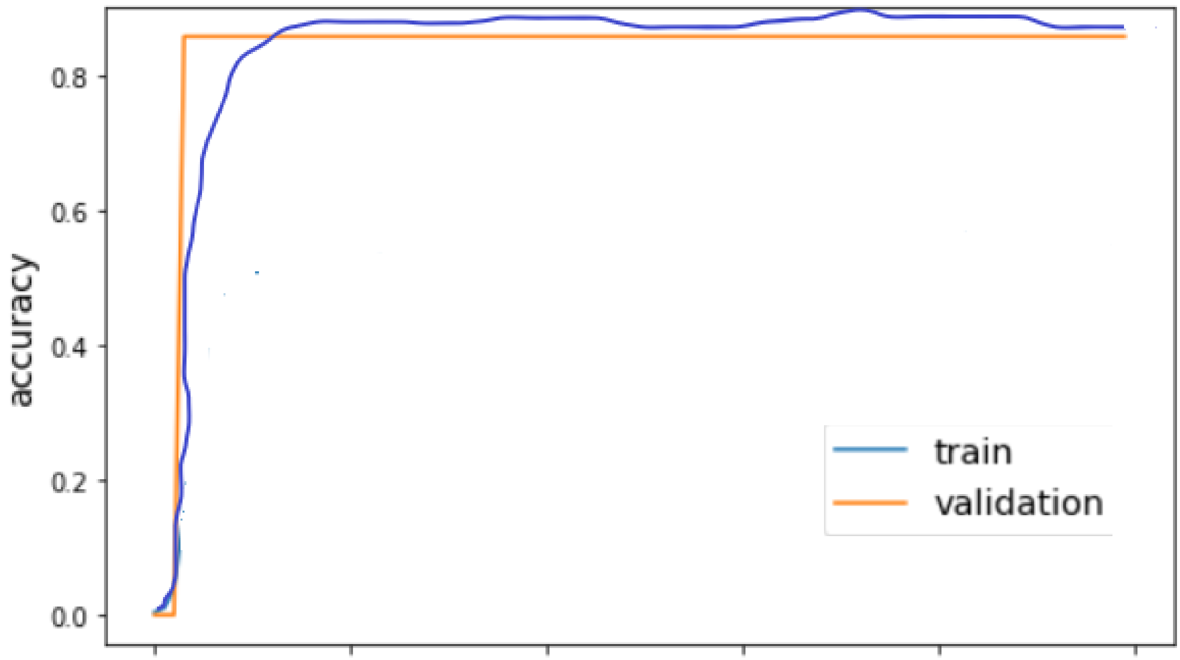


Figure 18: Comparison of train & test accuracy for new cases

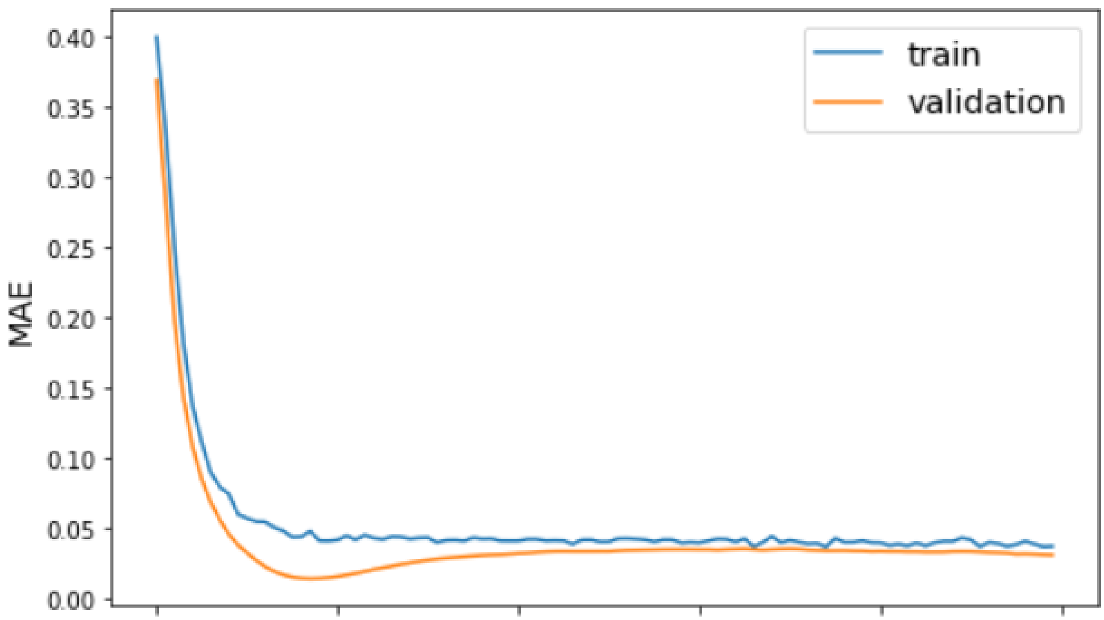


Figure 19: Comparison of MAE between train and test data for new cases

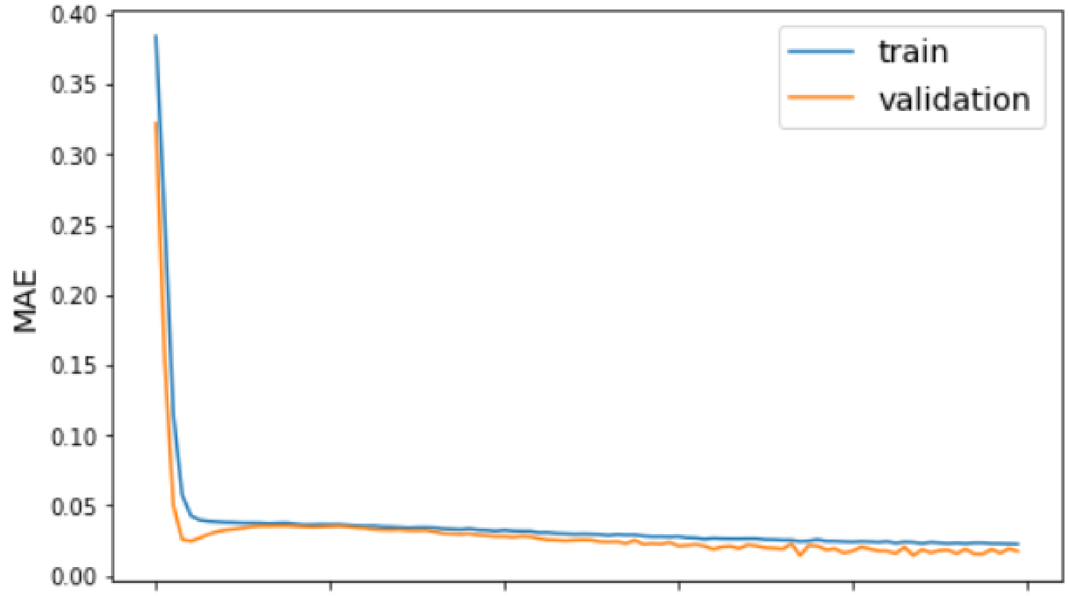


Figure 20:Comparison of MAE between train and test data for new deaths

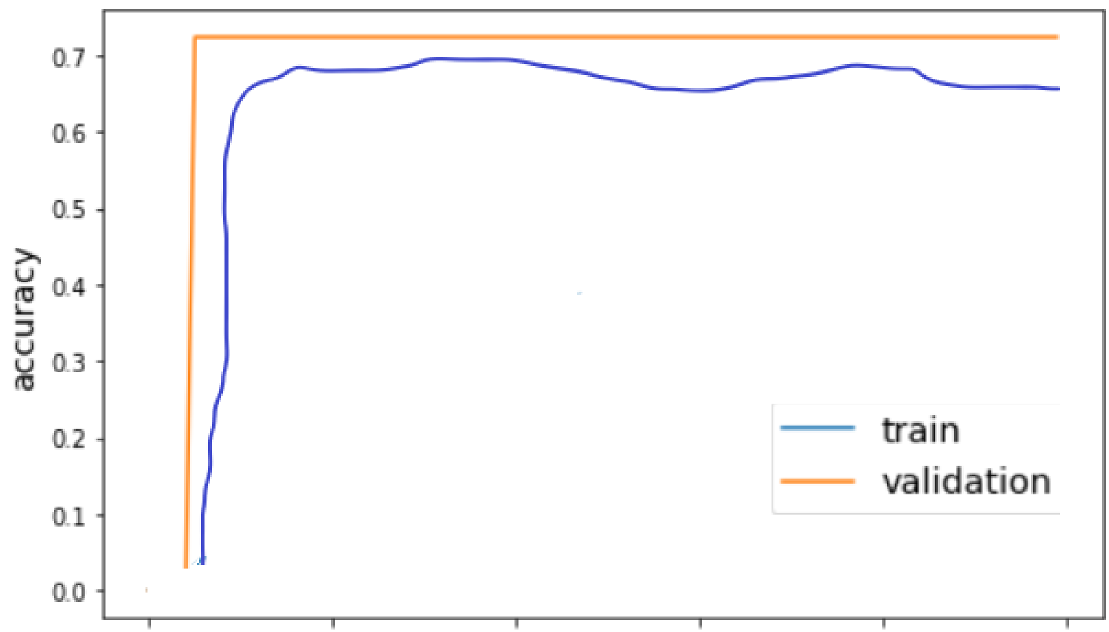


Figure 21: Comparison of train & test accuracy for new deaths

7.3 Evaluation Metric

7.3.1 RMSE (Root Mean Squared Error)

Before discussing the models' actual prediction outcomes, it's crucial to first go over the assessment metric that was used to assess their performance: the Root Mean Squared Error (RMSE). The RMSE is given by:

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(y_i' - y_i)}{n}} \dots\dots\dots\dots\dots\dots\dots\dots (10)$$

Where y' represents the anticipated value, y represents the observed value, and n is the number of observations. The mean squared error (MSE) stated in Section 2.3 is quite comparable to this statistic. The total of the squared distance between the observed and predicted values is calculated using this evaluation metric. It then takes the square root of this sum, effectively equating the RMSE to the residuals' standard deviation, or prediction errors. As a result, a smaller RMSE denotes improved performance and more precise forecasts.

7.3.2 MAE (Mean Absolute Error)

The average difference between calculated and real values is calculated using Mean Absolute Error. Because it calculates inaccuracy in observations taken on the same scale, it's also known as scale-dependent accuracy. It's a statistic for evaluating regression models in machine learning. It calculates the differences between actual and model-predicted values. It is used to forecast the machine learning model's accuracy.

The Mean Absolute Error is calculated as:

$$MAE = \frac{1}{n} \sum |y_i - x_i| \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (11)$$

Where y_i is the actual value of the i-th observation and x_i is the calculated value of i-th observation.

7.3.3 MSLE (Mean Squared Logarithmic Error)

The mean squared logarithmic error (MSLE) is a measurement of the difference between true and anticipated values. MSLE now only cares about the relative difference between the true and predicted values, or in other words, the percentual difference between them.

This means that MSLE will consider tiny discrepancies between true and predicted values in the same way as large disparities between true and anticipated values are treated.

$$MSLE(y, \hat{y}) = \frac{1}{n}\sum_{i=0}^{n}(\log(y_i + 1) - \log(\hat{y}_i + 1))^2 \ldots\ldots\ldots (12)$$

All the results obtained from our experiment are shown in the below table:

Error Metric	New_cases		New_deaths	
	Train	Validation	Train	Validation
RMSE	48.6342	36.8964	64.7832	62.9867
MAE	654.5477	712.2435	875.3759	813.4573
Accuracy	0.8554	0.8276	0.7136	0.7268
MSLE	0.5267	0.5062	0.6738	0.5937

Table 1: Results of our experiments

7.4 VISUALIZATION

In the table below, we compare the actual and predicted numerical values of Covid-19 new cases all over the world.

Date	**Actual Cases**	**Predicted Cases**
18/09/21	98,732	88,818
19/09/21	98,492	88,812
20/09/21	98,379	88,458
………..	……….	……..
………..	……….	………
02/02/22	98,673	88,752
03/02/22	97,613	89,298
04/02/22	98,432	88,659

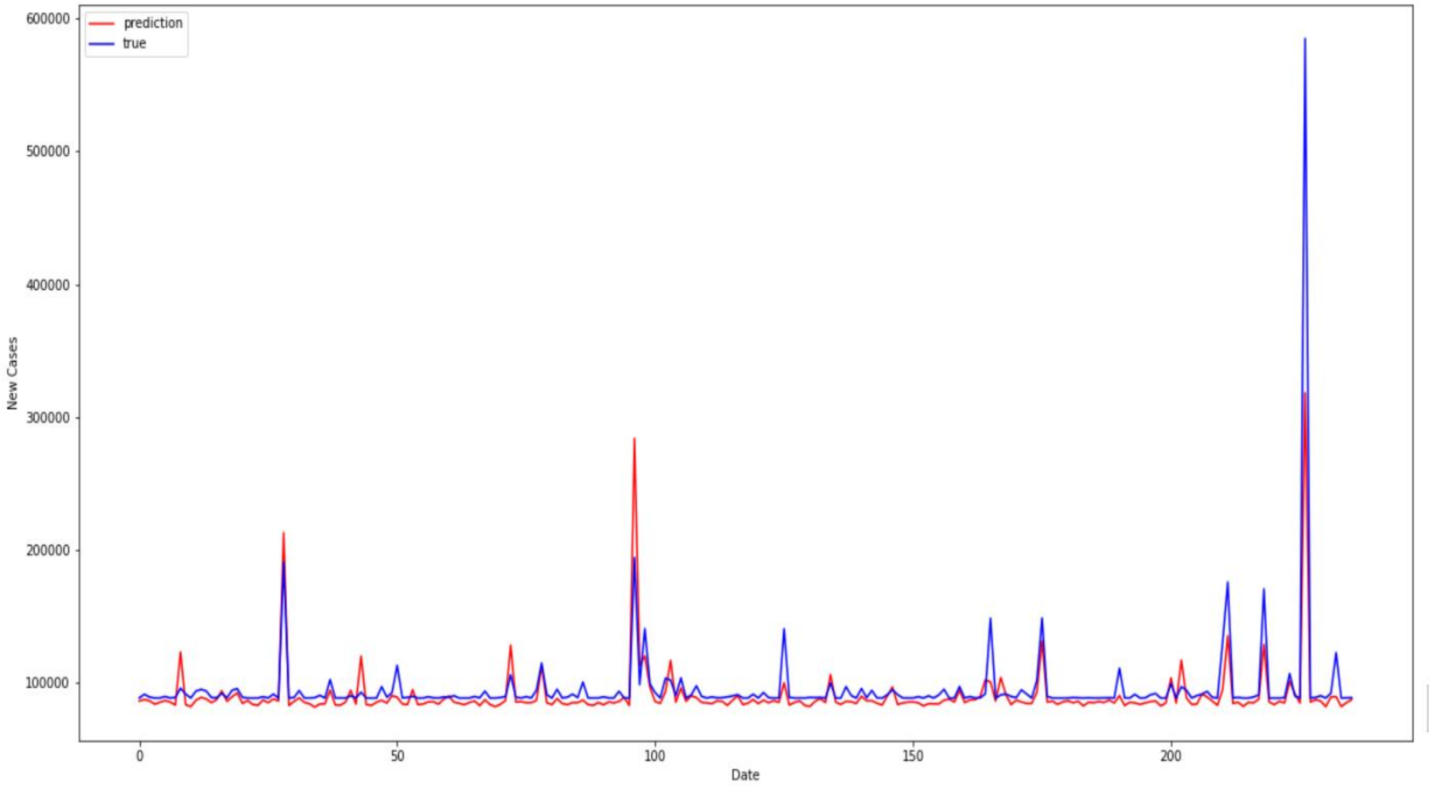


Figure 22: Globally Covid-19 new case prediction. The blue line indicates actual value and the red line indicates predicted value

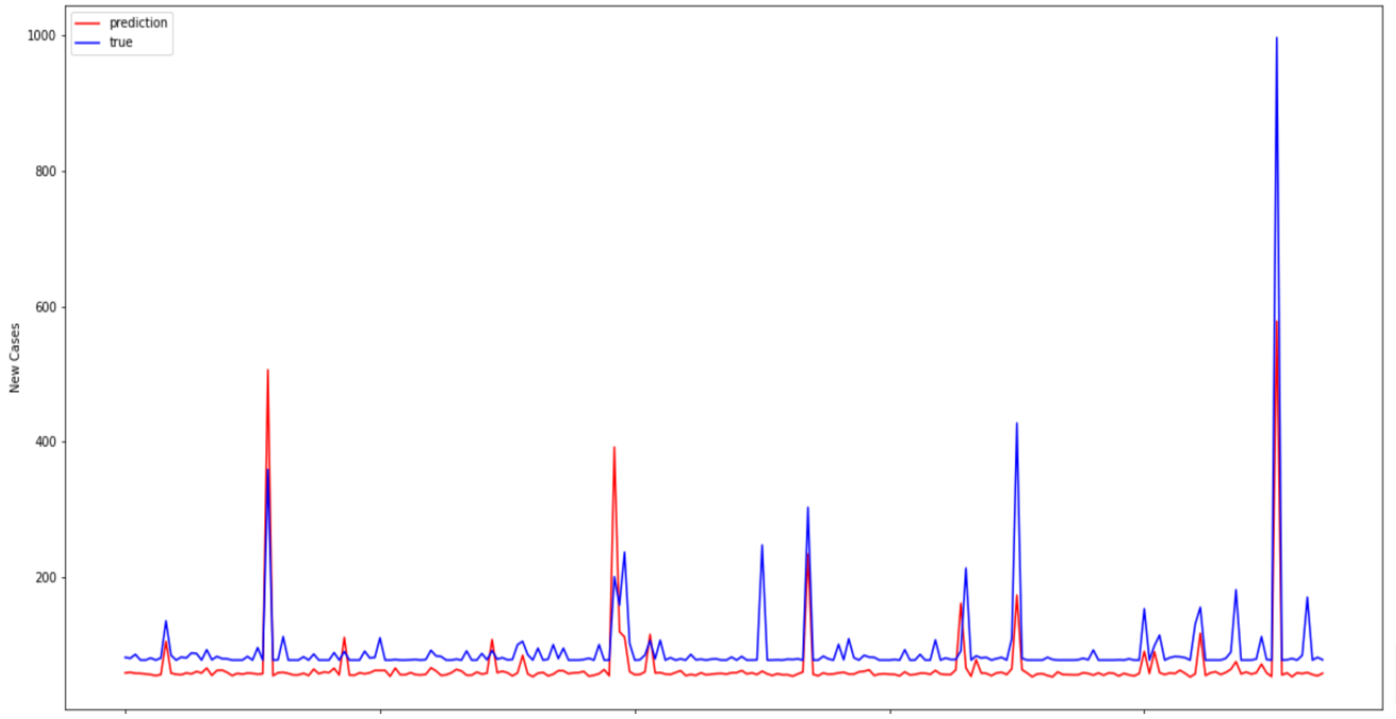


Figure 23: Globally Covid-19 new deaths prediction. The blue line indicates actual value and the red line indicates predicted value

8 DISCUSSION

8.1 PERFORMANCE ANALYSIS

The proposed model is one of the very uncommon machine learning algorithms as it combines a time-series forecasting model and a graph neural network model. The GCN-LSTM model is an extension of the baseline LSTM and is expected to perform better due to the model's integration of the geography of the world in making its predictions. The GCN-LSTM's sophisticated architecture had an impact on the result we obtained. Looking at the graph data structure of the countries all over the world shown in Figure 4, there are a total of 55,696 edges connecting the 236 countries, which certainly played a role in its long training time. It is a very complicated network to learn. It performs comparatively well in predicting the transmission of COVID-19 on a macro-scale. Our research reveals several insights about the pandemic.

Whereas other studies mentioned in Section 1.3.2 that used Graph Neural Networks in their macro-scale studies are able to incorporate mobility data into their neural networks, mobility data is not available in a format that could be used by our GCN-LSTM model. The integration of mobility data among the towns could possibly improve the results of this model.

Furthermore, the GCN-LSTM model may have not been used to its full potential due to the absence of mobility data in the Graph Convolutional Network. Though it may not have been the best choice for this particular macro-scale time-series forecasting problem, it is an extremely robust model that is certainly worth exploring, given its potential to incorporate spatial features into its learning.

8.2 FUTURE WORKS

We have the opportunity to do a fundamental analysis utilising the GCN-LSTM machine learning models to create predictions as part of our time-series forecasting research on COVID-19 transmission over the world. These are based simply on the case numbers and distances of the 236 different countries throughout the world. We evaluate the model performance by reducing the Root Mean Squared Error (RMSE) with the goal of constructing a model that can best forecast the spread of COVID-19 (RMSE). However, certain areas of our research can be built upon to increase the breadth of our findings. These include further regional analysis, population analysis, and a chronology analysis to assess the various stages of the worldwide pandemic.

When studying the transmission of a disease, there are various aspects to examine in addition to the RMSEs of each machine learning model. One such consideration is the location of countries in relation to neighbouring countries, as well as the distance between them. This has ramifications for future research topics relating to COVID-19 macro-scale time-series forecasting. One strategy to improve a machine learning model's outcome is to reduce its size even more, focusing on specific towns or cities within a country. We can see from the data from the previous two years that the pandemic may have affected different countries in different ways. Unfortunately, this has not been investigated as part of our research, but it may be in the future.

One of the other areas of future research is a timeline study of the epidemic. The time-series data would be separated by the several phases of reopening in this research. Every country takes precautions from the start of the pandemic by enforcing a lockdown or curfew. We can inherently capture the social distance rules, business openings, and other restrictions for the appropriate timeframe within our trained models by separating the phases in our time-series data and training our models on the phases independently. All of these characteristics would be established by the reopening phase being assessed. While it's unclear how much of an influence this timeline analysis will have on reducing the RMSE of our machine learning models, it's a promising step for the research's future.

9 Conclusion

The COVID-19 epidemic has presented economies and healthcare systems around the world with unprecedented challenges. The virus quickly spread over the world, destroying our daily lives, healthcare systems, economies, and practically everything else. We have lost about 63 million human lives at this point in history. Researchers have been trying to forecast the nature of the COVID-19 virus and its influence on human existence since the beginning of the pandemic. However, there was not enough data available in the early days to assess the impact of this lethal virus on a big scale. Despite this, a significant amount of research has been conducted in order to forecast future possibilities. Many machine-learning models for prediction have been suggested and successfully implemented. We compiled global statistics from COVID-19 to the present day, including daily new cases and deaths per country. While current research has looked at the virus's propagation at a meso-scale, which includes towns and cities within a state, previous models have not looked at COVID19 transmission at the macro-scale. We propose one deep learning model to investigate this topic using COVID-19 case-count data from all nations around the world: Long Short-Term Memory Graph Convolutional Network (GCN-LSTM). Though the findings of this study are simply a starting point for the types of insights that may be gained from meso-scale time-series forecasting, we expect that with further improvements to our models, we can construct a foundation of evaluation and preparedness for future pandemics.

10 REFERENCES

[1] W. H. O. (WHO), "WHO Coronavirus (Covid-19) Dashboard," 04 2022. [Online]. Available: https://covid19.who.int/.

[2] C. f. D. C. a. P. (CDC), "Frequently Asked Questions," 04 2022. [Online]. Available: https://www.cdc.gov/coronavirus/2019-ncov/faq.html.

[3] Z. Liu, "Predicting the cumulative number of cases for the COVID-19 epidemic in China from early data," arXiv preprint arXiv:2002.12298, 2020.

[4] C. Ian, A. Mondal and C. G. Antonopoulos, "A SIR model assumption for the spread of COVID-19 in different communities.," Chaos, Solitons & Fractals, vol. 139, p. 110057, 2020.

[5] Wu, J. T, K. Leung, Leung and G. M., "Nowcasting and forecasting the potential domestic and international spread of the 2019-nCoV outbreak originating in Wuhan, China: a modelling study.," The Lancet, vol. 395, no. 10225, pp. 689-697, 2020.

[6] Godio, Alberto, F. Pace and A. Vergnano, "SEIR modeling of the Italian epidemic of SARS-CoV-2 using computational swarm intelligence," International journal of environmental research and public health 17, vol. 10, p. 3535, 2020.

[7] L. Lorch, W. Trouleau, S. Tsirtsis, A. Szanto, B. Schölkopf and M. Gomez-Rodriguez, "A spatiotemporal epidemic model to quantify the effects of contact tracing, testing, and containment.," arXiv, 2020.

[8] T. Kufel, "ARIMA-based forecasting of the dynamics of confirmed Covid-19 cases for selected European countries," Equilibrium. Quarterly Journal of Economics and Economic Policy, vol. 16, no. 2, 2021.

[9] Chimmula, V. K. Reddy and L. Zhang, "Time series forecasting of COVID-19 transmission in Canada using LSTM networks.," Chaos, Solitons & Fractals, vol. 135, no. 109864, 2020.

[10] Tian, Yuan, I. Luthra and X. Zhang, "Forecasting COVID-19 cases using Machine Learning models," MedRxiv, 2020.

[11] Zeroual, Abdelhafid, F. Harrou, A. Dairi and Y. Sun, "Deep learning methods for forecasting COVID-19 time-Series data: A Comparative study.," Chaos, Solitons & Fractals, vol. 140, no. 110121, 2020.

[12] L. Zhu, T. Li and S. Du, "TA-STAN: A Deep Spatial-Temporal Attention Learning Framework for Regional Traffic Accident Risk Prediction," 2019 International Joint Conference on Neural Networks (IJCNN), 2019, pp. 1-8, doi: 10.1109/IJCNN.2019.8852212.

[13] K. Amol, X. Ben, L. Liu, B. Perozzi, M. Barnes, M. Blais and S. O'Banion, "Examining covid-19 forecasting using spatio-temporal graph neural networks," arXiv preprint arXiv, no. 03113, 2020.

[14] J. Dammen, "End-to-end deep learning for," Norwegian University of Science and Technology.

[15] J. Brownlee, "What Is Time Series Forecasting?," Machine Learning Mystery, 2 December 2016. [Online]. Available: https://machinelearningmastery.com/time-series-forecasting/.

[16] S. Hochreiter and J. Schmidhuber, "LONG SHORT-TERM MEMORY," Neural Computation, vol. 9, no. 8, pp. 1735-1780, 1997.

[17] O. B. Sezer, "LSTM_RNN_Tutorials_with_Demo," Github, 03 Jan 2019. [Online]. Available: https://github.com/omerbsezer/LSTM_RNN_Tutorials_with_Demo#References.

[18] P. Baheti, "Activation Functions in Neural Networks [12 Types & Use Cases]," V7, 15 June 2022. [Online]. Available: https://www.v7labs.com/blog/neural-networks-activation-functions.

[19] Keras, "Adam," 2021. [Online]. Available: https://keras.io/api/optimizers/adam/.

[20] N. Choi, "A Deep Learning Model for Predicting COVID-19 Transmission in Connecticut," University of Connecticut, 2021.

[21] P. Elinas, "Knowing Your Neighbours: Machine Learning on Graphs," stellargraph, 06 June 2019. [Online]. Available: https://medium.com/stellargraph/knowing-your-neighbours-machine-learning-on-graphs-9b7c3d0d5896.

[22] WHO, "WHO Coronavirus (COVID-19) Dashboard," [Online]. Available: https://covid19.who.int/data.

[23] StellarGraph, "StellarGraph Machine Learning Library," 30 June 2020. [Online]. Available: https://pypi.org/project/stellargraph/#description.

[24] O. B. Sezer, "LSTM_RNN_Tutorials_with_Demo," Github, 03 Jan 2019. [Online]. Available:https://github.com/omerbsezer/LSTM_RNN_Tutorials_with_Demo#References.

[25] P. Elinas, "Knowing Your Neighbours: Machine Learning on Graphs," stellargraph, 06 June 2019. [Online]. Available: https://medium.com/stellargraph/knowing-your-neighbours-machine-learning-on-graphs-9b7c3d0d5896.