Machine Learning Approaches for COVID-19 Forecasting

Othman Istaiteh[*], Tala Owais[†], Nailah Al-Madi[†], Saleh Abu-Soud[†]
[*]Computer Science Dept.
[†]Data Science Dept.
[*†]Princess Sumaya University for Technology
Amman, Jordan
Email: othmanistaiteh@gmail.com, talaowis@gmail.com, n.madi@psut.edu.jo, abu-soud@psut.edu.jo

***Abstract*—COVID-19 (Coronavirus) pandemic tends to be one of the most global serious issues in the last century. Furthermore, the world did not face any similar experience regarding the spread of the virus and its economic and political impacts. Forecasting the number of COVID-19 cases in advance could help the decision-makers to take proactive measures and plans. This paper aims to provide a global forecasting tool that predicts the COVID-19 confirmed cases for the next seven days in all over the world. This paper applies four different machine learning algorithms; The autoregressive integrated moving average (ARIMA), artificial neural network(ANN), long-short term memory (LSTM), and convolutional neural network (CNN) to predict the COVID-19 cases in each country for the next seven days. The fine-tuning process of each model is described in this paper and numerical comparisons between the four models are concluded using different evaluation measures; mean absolute error (MAPE), root mean squared logarithmic error (RMSLE) and mean squared logarithmic error (MSLE).**

***Index Terms*—COVID-19, Spatial Time-series Forecasting, Deep Learning, ARIMA, ANN, CNN, LSTM.**

I. Introduction

The world nowadays faces the COVID-19 pandemic, which has a very large global spread that affects the economy and more importantly leads to a large number of deaths around the world, more than any other lethal pandemic since the Spanish flue. Moreover, the world faced many other diseases in this century such as Severe Acute Respiratory Syndrome (SARS), Middle East respiratory syndrome-related coronavirus (MERS), and swine flu [1]–[3] As a result, providing an accurate prediction tool of the cases of the disease is very essential for the political and healthcare leaders to make the proper decisions in advance. Time-series forecasting aims to predict future events based on historical data using statistical and machine learning algorithms. Time-series forecasting is well studied in the literature [4] but on the other hand, spatial time-series forecasting which predicts events in the future for several locations or spaces is a special case that has fewer studies. Some existing methods have been applied in the spatial time series forecasting problems such as the hybrid model of autoregressive integrated moving average (ARIMA) and deep belief networks (DBN) to forecast the red tide in two cities [5]. CNN, and LSTM to forecast traffic flow prediction [6], [7].

Machine learning algorithms have been using in many prediction applications, especially for time-series problems. The ARIMA algorithm has been used widely for forecasting linear time-series problems [8], and it consists of an autoregressive expression (AR), a moving average (MA) component, and a differencing term (I). Deep learning (DL) algorithms have been used lately for the task of time-series forecasting and spatial time-series forecasting [9]. One of the widely used algorithms is Long short term memory (LSTM) [10] which is a recurrent neural network (RNN) that extracts the similarities between sequential inputs by using three gates; forget gate, input gate, and output gate to control the relationship between the previous inputs, the current input, and the output. Every LSTM cell consists of those three gates and usually, the LSTM layer contains several LSTM cells.

CNN [11] models are widely used in the literature for image and video classification, but it gained more attention lately in the spatial time-series forecasting because of the CNN filters that group the neighbor records with a window that can be selected, after that a non-linear activation function is applied which helps in extracting the non-linear relations between neighbor records. This can be very useful in time-series in general and more important in spatial time-series.

The ANN [12] model is the simplest form of DL algorithms that extract non-linear relation in the data by applying activation functions between every two layers. ANN consists of one or more hidden layers, input and, output layers; each of them contains a different number of hidden nodes. Although ANN does not apply a grouping of the neighbor records like CNN neither uses the previous values in the same way as LSTM, it provides reasonable results.

The contribution of this paper is investigating the best architecture and hyperparameters of the ARIMA, ANN, LSTM, and CNN models, to come out with the most accurate forecasting model to predict the confirmed cases for COVID-19 of the next seven days based on the historical data. Furthermore, the paper conducts a comparative analysis based on three measurements; mean absolute error (MAPE), root mean squared logarithmic error (RMSLE) and mean squared logarithmic error (MSLE) to compare the four models performance and the same model with different configurations.

This paper is organized as follows. Section II discusses the used models and algorithms in the literature. Section III briefly describes the dataset used. Section IV illustrates and discusses the proposed approaches. Section V illustrates the preprocessing steps and the experimental setup. Section VI presents the results of the four models in addition to the analysis and comparison of the results. Section VII concludes the paper.

II. Literature Review

Spatial-Temporal time series is the field of time series forecasting where the variables vary through time and space. Spatial-Temporal forecasting can be applied to diseases forecasting such as COVID-19 as it analyzes the patterns and provides a good prediction for decision-makers around the world to help them make the right decisions at the right times.

Authors in [13] applied the ARIMA model to John Hopkins University COVID-19 data [14] from January 20, 2020, to February 10, 2020, to forecast the Covid-19 cases around the world for the next two days. The authors have used the autocorrelation function (ACF) and partial autocorrelation (PACF) graphs to choose the best parameters for the model. Authors of [15] used LSTM to predict the COVID-19 trend in China, they have used the dataset for the 2003 SARS epidemic statistics and used additional factors that are related to COVID-19 such as the probability of transmission, incubation rate, the probability of recovery or death, and contact number to build the model.

Authors in [16] applied multi-input CNN to predict the confirmed cases of COVID-19 in China cities from January 20, 2020, to March 23, 2020. The authors have compared the performance of CNN with other algorithms and figured out that CNN outperformed the other algorithms. In [17]The authors have applied the LSTM model algorithm on a dataset from 30th January 2020 to 4th April 2020 to predict the confirmed cases for the next 30 days in India. Authors in [18] have used LSTM to build a real-time forecasting model to predict COVID-19 cases in Canada for the two successive days, they based their findings on data until March 31, 2020. Paper [19] employed LSTM for time series forecasting for COVID-19 confirmed cases in Iran. The data used in the model was collected from February 19, 2020, to March 22, 2020, at the provincial level based on the Iranian Ministry of Health and Medical Education, It has also compared LSTM with Seasonal ARIMA, exponential smoothing, and moving average approaches and derived that LSTM has performed better than other algorithms.

Authors in [20] used LSTM and ARIMA algorithms to build the prediction model for COVID-19 confirmed cases in four countries; the US, Italy, Spain, and Germany. The collected data was until May 25, 2020. They applied multiple variations of the LSTM like vanilla LSTM, stacked LSTM, and Bidirectional LSTM. The authors have concluded that LSTM models perform equivalently as the ARIMA model pointing that each model has advantages and disadvantages. As for [21], the authors have employed LSTM and RNN to forecast COVID-19 confirmed cases over 100 countries. The data used to train the models was collected from after January 22, 2020, until May 1, 2020. The authors have chosen the best settings for the models heuristically. LSTM had the least Root Mean Square Error and it outperformed the RNN.

Unlike the papers [13], [15]–[20] where they applied the prediction models on specific regions, this paper aims to employ spatial-temporal forecasting for 189 countries around the world with the focus on applying the models used in the literature to predict the COVID-19 confirmed cases. This paper also spots the lights on using ANN for spatial Temporal Forecasting problems. This paper compares the four most used algorithms in the literature to predict the COVID-19 cases in the next seven days using three evaluation measurements; MAPE, RMSLE, and MSLE.

III. Dataset

The dataset used in this paper is a Novel COVID-19 dataset which was developed by the Johns Hopkins University Center for Systems Science and Engineering (JHU CSSE) [14]. It consists of 266 records and 165 columns, a sample of five records is shown in Table I, where the first and second columns indicate the country and the Province/State for some countries that are divided into Provinces/States. Moreover, the lat and long of every record are provided followed by the date and number of accumulated confirmed cases on that day. The dataset covers the period from the 22nd of Jan 2020 until the 30th of June 2020, where the total number of countries is 189 until that date. The JHU CSSE collects data from reliable resources such as the WHO (World Health Organization) and provides a daily update to the data. The data includes no missing values, and the data was preprocessed by aggregating cases of countries with multiple states because the paper focuses on predicting the COVID-19 cases country's wise.

IV. Proposed Approaches

This section gives a detailed explanation of the models applied in this paper. The models are categorized into two main categories, the first category contains the statistical model -ARIMA- while the second category includes the DL methods; ANN, LSTM, and CNN. Every Category is explained in a separate subsection.

A. *Autoregressive Integrated Moving Average (ARIMA)*

ARIMA is a statistical method used in literature to perform time-series prediction. ARIMA was applied in various fields such as business, and stock markets [8]. This paper applies ARIMA on Virus confirmed cases prediction matter to inspect how valid this approach could be in prediction and forecasting for such matters. Because we are working on spatial-temporal data as the dataset contains the confirmed cases cumulative rates for 189 countries for the time frame January 22, 2020, until June 30, 2020. Each country time-series data is input to a separate ARIMA model and Grid Hyper Parameters tuning was used to get the best p,d,q parameter combination for ARIMA [22] which in turn will specify the confirmed cases

TABLE I
A SAMPLE OF COVID-19 DATASET FOR FIVE RECORDS OUT OF 266 RECORDS OF ONE DAY OUT OF 161 DAYS

Country/Region	Province/State	Lat	Long	Date	Confirmed
Antigua and Barbuda	-	17.0608	-61.7964	20/3/2020	1
Argentina	-	-38.4161	-63.6167	20/3/2020	128
Armenia	-	40.0691	45.0382	20/3/2020	136
Australia	Australian Capital Territory	-35.4735	149.0124	20/3/2020	6
Australia	New South Wales	-33.8688	151.2093	20/3/2020	353

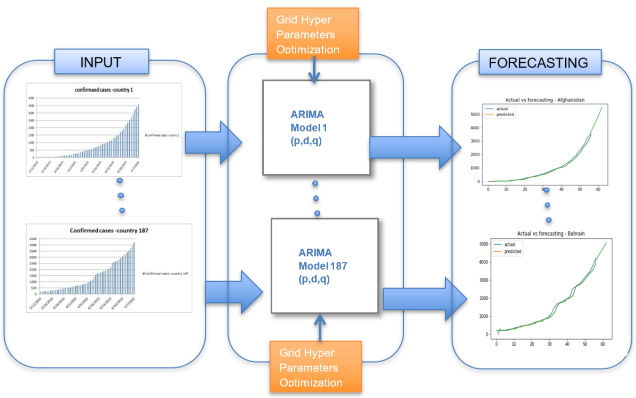

Fig. 1. The flow diagram of the ARIMA model

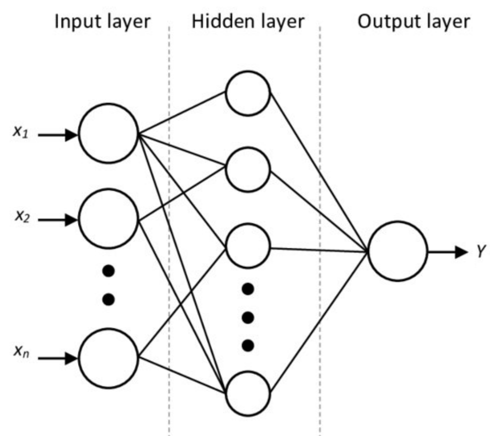

Fig. 2. ANN Architecture

time lags, the differencing factor which helps in smoothing the time series line to make it stationary and the error term lags in the prediction process. Figure 1 shows the flow diagram used.

B. Deep Learning Models

The DL models tend to outperform statistical methods in extracting non-linearities. This paper applied ANN, LSTM, and CNN. Each model is explained in a subsection.

1) Artificial Neural Network (ANN): A single-layer neural network shown in Figure 2 was selected in this study to extract the nonlinearities in the input data. The input layer consists of the COVID-19 confirmed cases in nine days period for one country followed by one layer of hidden units each layer consists of 512 nodes and the output layer predicts the number of cases on the tenth day. ReLU activation function was used between layers due to the fast computation and as it overcomes the problem of vanishing gradient, and a 0.3 dropout was applied to avoid the problem of overfitting. Furthermore, the number of inputs, hidden layers, and nodes per hidden layers were fine-tuned and will be discussed in the experiment setup section.

2) Long Short Term Memory (LSTM): A single-layer LSTM network was used to predict the COVID-19 cases, where LSTM is the most used type of RNN and it is widely used in time series forecasting [23]. The input of each LSTM unit is the output of the previous LSTM as shown in Figure 3,

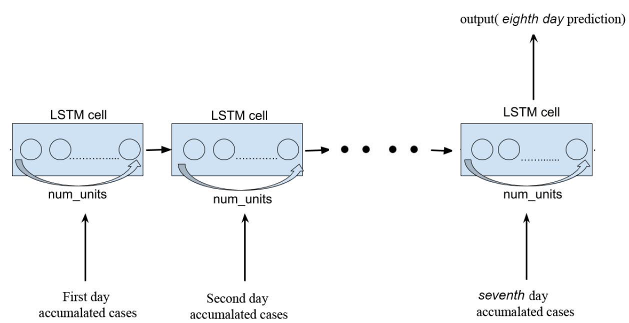

Fig. 3. LSTM Architecture

for that LSTM tends to be very useful because the current step depends on the last steps and it is the case in time-series data, where the current value depends on the previous historical values. The hyperparameters of the LSTM were selected using fine-tuning which will be discussed in the experiment setup section. Every instance consists of 7 inputs enter sequentially to 1024 nodes LSTM layer and the final output is the prediction of the eighth day. Moreover, ReLU was used as the activation function and a 0.2 dropout was applied to increase the generalization of the model.

3) Convolutional Neural Network (CNN): The convolutional neural network tends to be very useful in image clas-

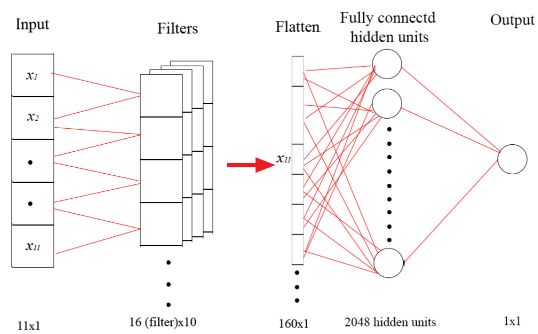

Fig. 4. CNN Architecture

TABLE II
ERROR MEASURES

Mean Absolute Percentage Error	MAPE = $\frac{100\%}{n}\sum_{i=1}^{n}\left\lvert\frac{Y_a - Y_p}{Y_a}\right\rvert$
Root Mean Squared Logarithmic Error	RMSLE = $\sqrt{\frac{1}{n}\Sigma_{i=1}^{n}\Big(\log(Y_a) - \log(Y_p)\Big)^2}$
Mean Squared Logarithmic Error	MSLE = $\frac{1}{n}\sum_{i=1}^{n}\Big(\log(Y_a) - \log(Y_p)\Big)^2$

sifications. However, it has shown very good results in other tasks such as time series forecasting [24]. The most common type of CNN in the literature is 2D CNN but in this paper, 1D CNN filters as shown in Figure 4 were used to extract the non-linear relations between every two successive days by grouping them and applying the ReLU activation function, the number of filters was fine-tuned and chosen to be 16, as well as the other hyperparameters and it will be discussed in details in the experiment setup section. The number of input cases was chosen to be the cases of a period of 11 days because it provided the best results with the output of the 1D CNN filter enters a hidden layer that consists of 2048 nodes with ReLU activation function as well and the output is fed to one neuron output which is the prediction value of the 12th day. Moreover, a value of 0.2 dropout was used to overcome the problem of the overfitting since not all connections are activated which significantly increases the generalization of the model.

V. EXPERIMENT SETUP

This section is divided into two subsections, the first one discusses ARIMA setup and the other one discusses the DL algorithms setup. Error measures such as MAPE, RMSLE, and MSLE are used for both the validation and the testing steps. Table II shows the equations of the three measures.

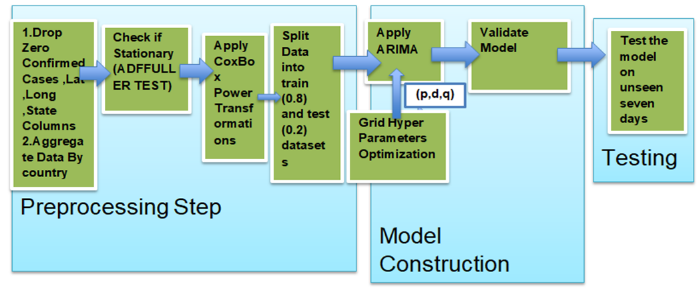

Fig. 5. The flow diagram of the ARIMA model Preprocessing and Construction steps

A. ARIMA Preprocessing and Setup

The ARIMA model performs the forecasting using the prediction equation (1) which considers the past time lags and errors. *Xti* represents predicted value at time t which is calculated by taking the difference between previous time lags and the past errors as shown in equation (1). ARIMA assumes that error lags are independently distributed with a mean of zero and a constant variance. Figure 5 shows the flow diagram of the ARIMA model preprocessing and construction steps.

$$\begin{aligned} X_{t_i} \;=\;& a_{t_i} + \Phi_{1_i} X_{t-1_i} + \Phi_{2_i} X_{t-2_i} + \ldots + \Phi_{pi} \\ & -\theta_{1_i}\epsilon_{t-1_i} - \theta_{2_i}\epsilon_{t-2_i} - \ldots - \theta_{q_i}\epsilon_{t-q_i} \end{aligned} \tag{1}$$

As expected when dealing with a time series forecasting problem for a new virus, there was a lack of diagnosis in the first days of disease which caused many zero values in the confirmed cases numbers in the data set for the early days. To avoid those zeros, confirmed cases up to March 12, 2020, have been dropped since those zero values can affect ARIMA in a bad way. Lat, Long, and state columns have been dropped because they are not useful while using ARIMA because it is purely designed to predict linear time series problems and it only accepts time series as its input. Data is then aggregated by country to get the time series data by country. In the second stage, we have checked if the time series data for each country is stationary because ARIMA is capable of making good predictions for stationary data and unit root non-stationary data –as there are many types of non-stationary data [22] which are resolved through the differencing feature available in ARIMA. To avoid the bad prediction results of non-stationary data Box & Cox Power Transformations [25] have been applied to each country's time-series dataset.

Box & Cox (1964) [25] proposed a parametric power transformation technique to reduce anomalies such as non-additivity, non-normality, and heteroscedasticity. The Box & Cox transform is a configurable data transform method that supports both square root and log transform, as well as a suite of related transforms. More than that, it can be configured to evaluate a suite of transforms automatically and select the best fit. The resulting series from Box & Cox may be more linear and the resulting distribution more Gaussian or Uniform [25]. Then data has been shifted by constant -1, so that the zero confirmed cases do not affect the log transformation if it was selected as the best fit transformation by Box & Cox. For preparing the data, each country's time series data has been split into 7 days for testing, and the days from March 12, 2020, until June 24, 2020, were used as %80 for training and %20 for validation. The validation set was used to improve the model performance and to avoid the problem of overfitting on the training set.

For setting the best parameters (AR (p), I(d), MA (q)) for each ARIMA model, the Grid Hyperparameter Optimization approach has been employed to get the ARIMA Model with the least Akaike Information Criterion (AIC), the goodness of fit criteria. The least the AIC the better fit of the model. Different criteria such as MAPE, RMSE have been tried in this experiment. By trial and error, it has proved that the least the AIC is the best model fit. Grid Search has been used to search the best parameters in the range 0-2 as this range has proved to be the least computationally expensive and it also achieves the best ARIMA AIC results. After ARIMA has been applied, the model has to be validated by examining the residual plot thus that it must satisfy the mean of zero and constant variance to get the best results.

B. Deep Learning Model Preprocessing and Setup

The preprocessing stage is a very important stage in time-series, where it is very uncommon to input the whole data at once without time-series splitting, and it is even worse in this COVID-19 data case because the data belongs to different countries and some of those countries have a separate row for each state. The number of accumulated cases per day was aggregated for the states that belong to the same country, furthermore, lat, long, and other population-related data was dropped because by applying them on the three deep learning models, the performance of the models significantly decreased which indicates that those columns are not related to the number of COVID-19 confirmed cases.

The resulted data after the previous preprocessing are the number of accumulated cases for each country, which is then split to windows with n days where n is the size of the window that was considered as a hyperparameter and it was fine-tuned for each model, where 9 days window was the best for ANN, 7 days window for CNN, and 11 days window for LSTM. Each input record of the DL models represents a window that consists of n columns as input where every column represents the accumulated cases in a single day of a single country and the output represents the accumulated cases in the n+1 day. For example in the case of ANN, the input is the cases of the first 9 days and the output is the prediction of the 10th-day, the second input consists of the accumulated cases from the 2nd-day to the 10th-day inclusively and the output for the 2nd record is the cases of the 11th and so on.

Finally, the dataset was split into 3 sets, the first set is the training set which includes the first 147 days, where the validation set includes the next 7 days after the training days, and finally, the test set includes the last seven days. The selection of seven days for both validation and testing set was goal-driven because the goal of the paper is to predict the COVID-19 confirmed cases in the next 7 days based on historical data. Shuffling, or K-fold validation is not common as it provides underperformance because the train data should be earlier than the tested data. The whole preprocessing stage in addition to the model's construction is shown in Figure 6. The validation set is used to come up with the best hyperparameters of the DL model, while the test set is used to measure the performance of the final model on predicting the COVID-19 cases of all countries for the seven next days without being trained or validated on that data which simulates the real-life scenarios. The validation and testing were done using MAPE, RMSLE, and MSLE measuring metrics.

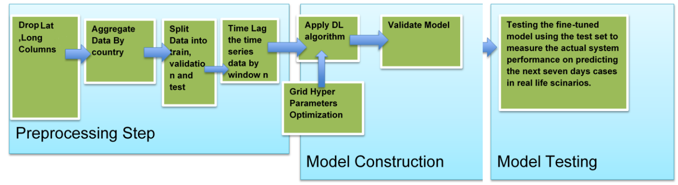

Fig. 6. The flow diagram of the DL models preprocessing and construction steps

The hyperparameters fine-tuning and the experiment setup varies between the three DL models, for that every model will be discussed in a separate subsection.

1) ANN Setup and Validation: The ANN tends to be the simplest deep learning algorithm where all neurons in every two adjacent layers are connected. After extracting the time series windows, a grid hyperparameters optimization was applied to choose the best parameter combination. Adam [26] optimization algorithm was selected since it showed great results in the literature in similar tasks. The training record contains nine days of cases for a single country and the output is the tenth day. Because of the noticeable differences in the measuring metrics values when training a model with the same hyperparameters several times which happens because of the random weight initialization, the experiment was done ten times on each hyperparameter combination and the mean was taken. Table III shows the best hyperparameters combination. Increasing the number of layers badly affects the performance of the system. The optimum number of nodes is 512 nodes, where increasing the number of nodes leads to overfitting, and decreasing them leads to underfitting. The best batch size is 8192 instances per batch which is the largest between the used DL models, and the best number of epochs is 80 which is the lowest between the DL models which can be justified by the simplicity of the ANN compared to other DL models. Moreover, the best dropout is 0.3 which gives a good generalization without leading to underfitting. The window size of nine days provided the best measures. Mean absolute percentage error was used as the loss function because it gives the same weight for countries with a large number of cases as well as countries with a small number of cases.

2) LSTM Setup and Validation: The first stage of LSTM setup and validation was conducted by applying a time series splitting as mentioned in the preprocessing stage. The window size is chosen to be one of the hyperparameters where the best measurements were obtained with window size equals 7. In the LSTM model, every cell output is determined by the input of the cell and output of the previous cell. Each record consists of only the accumulated number of the cases in a single country for seven days and the output indicates the prediction value of the eighth day. Adam optimization was applied to obtain the model with the best three measurements mentioned in the previous section. The best hyperparameters are shown in Table IV, where increasing the number of hidden layers tends to harm the performance on the validation set, while the optimum number of nodes is 1024. Mean absolute percentage error was used as the loss function for the same reason as ANN.

3) CNN Setup and Validation: Several CNN configurations are presented in the literature such as Resnet [27], AlexNet [28], and visual geometry group (VGG) [29]. However, after trying those networks and other networks, it was found that those complicated networks do not serve the purpose of this paper. By trial and error, it tends that a simple 1D CNN network performs very well and does not overfit the training data as the complicated networks. CNNs usually consist of CNN filters, followed by a pooling layer to decrease the dimensionality, after that flatten layer is applied, and finally, the output of the flatten layer enters a fully connected ANN hidden layers followed by the output layer. By applying several experiments it was clear that the pooling layer significantly harms the performance as well as adding multiple ANN or CNN filters layers. The final network consists of a 1D CNN filters layer followed by a flatten layer and finally, a hidden fully connected ANN layer ended with a single output node. The input consists of 11 values each indicates the number of accumulated cases of a single country in 11 successive days, and the output is the prediction of the 12th day. The hyperparameters were obtained by testing models with all different combinations on the validation set using a grid search method. The combination with best MAPE, RMSLE, and MSLE on the validation set was picked as the final model. Table V shows the hyperparameters that have shown the least error measures. ADAM optimizer and MAPE loss function were used for the same reason mentioned in the ANN section. One layer of 16 CNN filters followed by 2048 connected hidden nodes tends to provide the best performance. Increasing the number of filters did not affect the performance while the number of connected hidden nodes highly affect the performance as increasing the number of nodes leads to overfitting and decreasing them leads to an underfit model. A value of 0.2 dropout plays a significant rule in increasing the generalization while increasing the dropout harms the model by underfitting. The number of epochs is 120 which is very reasonable and a batch size with 1024 instances provided the best performance on the validation set and speed up the computation compared to smaller batch sizes.

VI. Results and Analysis

This section consists of two parts, the first part discusses each model separately where the second part provides a comparison between the four models. The results and graphs mentioned in this section were obtained by testing the final model. The testing was done on the COVID-19 confirmed cases between June 24, 2020, and June 30, 2020, where this data was not used for either the training or the validation to provide a real-life scenario where the cases are the next week and we do not have the future values in real-life scenarios.

A. Results

The results of each model are described in a separate subsection.

1) ARIMA Results: ARIMA model provided promising results in predicting accumulated cases especially with the countries that have linear cases curve. However, it did not perform well with the non-linear curves, and one other draw-back of the ARIMA that every country needs a separate ARIMA model which is computationally expensive and time-consuming. Figure 7 shows the prediction of China. Where the ARIMA performed very well. On the other hand, Figure 8 shows the prediction results of Eritrea. where he ARIMA model failed to predict precisely the number of cases.

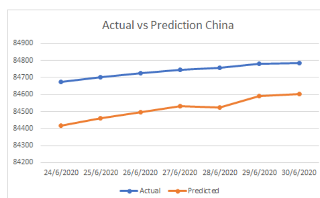

Fig. 7. Good prediction of ARIMA

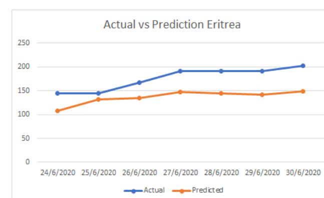

Fig. 8. Poor predictions of ARIMA

2) ANN Results: ANN provides very good results in fore-casting the accumulated COVID-19 cases with a very simple network consists of only one layer. Moreover, the training time on an average personal computer is about one second per epoch, which is very reasonable for the powerful forecasting it provides. Figure 9 shows how accurate the prediction of COVID-19 accumulated cases in a period of 7 days for Austria. While Figure 10 shows the prediction for Malawi where the ANN model failed to predict the cases with the same accuracy as Austria.

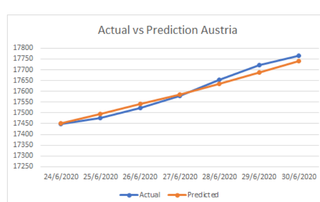

Fig. 9. Good prediction of ANN

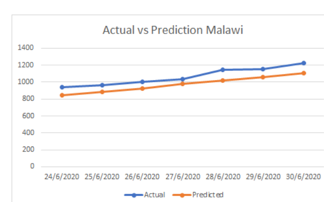

Fig. 10. Poor predictions of ANN

TABLE III
DL MODELS (CNN, LSTM, AND ANN) HYPERPARAMETERS

Model	Layers	Filters	Nodes	Window	Dropout	batch size	epochs
ANN	1	-	512	9	0.3	8192	80
LSTM	1	-	1024	7	0.2	1024	100
CNN	1	16	2048	11	0.2	1024	120

3) LSTM Results: One layer LSTM model provided a very good prediction in countries such as Brunei as shown in Figure 11, however, it did not perform well in predicting the number of cases in some countries such as South Africa as shown in Figure 12.

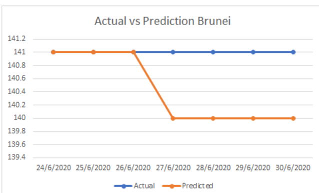

Fig. 11. Good prediction of LSTM

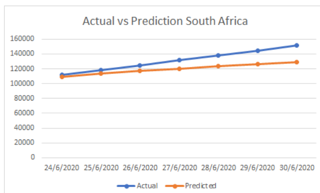

Fig. 12. Poor predictions of LSTM

4) CNN Results: Adding a single one-dimensional convolutional layer to the ANN gave a noticeable improvement by capturing the non-linear component between every two successive days and it surprisingly outperformed the LSTM which is known to perform better in similar tasks. The CNN model performed well in predicting the cases of many countries such as Egypt as shown in Figure 13, but it also provided a bad performance in some countries such as Croatia as shown in Figure 14.

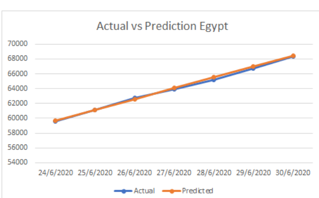

Fig. 13. Good prediction of CNN

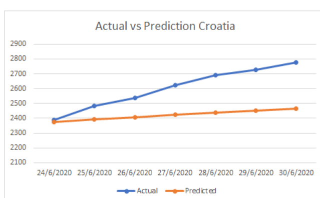

Fig. 14. Poor predictions of CNN

B. Analysis

Table VI shows the most commonly used error measures: root mean squared logarithmic error (RMSLE), mean absolute percentage error (MAPE), and mean squared logarithmic error (MSLE) of the fine-tuned models. The three DL methods provide similar results regarding the three used measures, while the ARIMA model provides worse performance by a big margin, for that it is not preferred to use ARIMA by itself because it is not capable of capturing the non-linearity in data in opposite to DL methods which can capture a large order of non-linearity due to using of non-linear activation functions such as ReLU which is used by the three deep learning models. Moreover, DL methods used only a single deep learning model to predict the accumulated cases of all countries, while the ARIMA method requires one model for each country which is very time-consuming in the process of training and fine-tuning.

TABLE IV
PREDICTION MODELS RESULTS

Prediction Model	MAPE	RMSLE	MSLE
ARIMA	14.14	0.17	0.06
ANN	3.23	**0.13**	**0.02**
LSTM	4.14	**0.13**	**0.02**
CNN	**3.13**	**0.13**	**0.02**

One-layer ANN tends to be the simplest deep learning algorithms but it provides the same RMSLE, and MSLE as LSTM and CNN, and it provides close MAPE to the best model CNN, using a single hidden layer of 512 hidden units, which is considered very small compared to the average personal computer capabilities nowadays. Secondly, the LSTM model with one hidden layer and 1024 LSTM units provided slightly worse MAPE which is considered the main measuring metric because it lowers the weights of outliers and gives the same weight for all countries. Moreover, the LSTM needs much more time in the training stage, which is around one hour on an average personal computer, which is a very reasonable time and negligible predicting time in seconds in the predicting stage. The fine-tuning of the LSTM needs much more time than the training time because the model needs to be tested on several hyperparameters combinations which can take few days on an average personal computer and would be better if it is done using GPUs. Moreover, CNN provides slightly better results than LSTM and ANN with very short training time and the fine-tuning time is also responsible. Finally, the three deep learning models provided similar performance in forecasting the COVID-19 cases for the next seven days and provided very low error measures compared to the statistical ARIMA model.

VII. CONCLUSION

COVID-19 pandemic tends to be one of the most global serious issues this century. This paper aims to apply four models (ARIMA, ANN, LSTM, and CNN) to predict the COVID-19 cases for the next week based on historical data. Moreover, this paper discussed the best configuration o f the four models by fine-tuning them using a validation set to figure out the best hyperparameters combination and the effect of each hyperparameter. Finally, A comparison between the four models was done using a test set to test the final fine-tuned model in a real-life scenario to predict the COVID-19 cases for the next seven days.

MAPE, RMSLE, and MSLE error measurements were used in the validation and testing stage. The four models were tested on a test set that was not used in the training or validation processes. The deep learning models outperformed the ARIMA model by a big margin regarding the three error measures. However, one dimensional CNN outperformed the other two deep learning models by a small margin followed by ANN and LSTM came third with very good results compared to ARIMA.

The deep learning models discussed in this paper are helpful and can be used by global organizations for the forecasting problem of COVID-19 as error measures show that they are reliable but yet; the high error rates for some countries need to be considered to make the models reliable for country-wise predictions. Because of the close results of deep learning models and high errors for some countries' predictions; implementing a hybrid deep learning model is a future direction to try to get the advantages of the three deep learning models in addition to using other countries-related data such as the lockdown periods and the schools' closures. Moreover, it would be more helpful to implement a model with a longer prediction period than seven days. Some promising machine learning techniques such as ensemble methods and clustering could be also taken into consideration.

References

[1] N. Zhong, B. Zheng, Y. Li, L. Poon, Z. Xie, K. Chan, P. Li, S. Tan, Q. Chang, J. Xie *et al.*, "Epidemiology and cause of severe acute respiratory syndrome (sars) in guangdong, people's republic of china, in february, 2003," *The Lancet*, vol. 362, no. 9393, pp. 1353–1358, 2003.

[2] R. J. de Groot, S. C. Baker, R. S. Baric, C. S. Brown, C. Drosten, L. Enjuanes, R. A. Fouchier, M. Galiano, A. E. Gorbalenya, Z. A. Memish *et al.*, "Commentary: Middle east respiratory syndrome coronavirus (mers-cov): announcement of the coronavirus study group," *Journal of virology*, vol. 87, no. 14, pp. 7790–7792, 2013.

[3] B. J. Coburn, B. G. Wagner, and S. Blower, "Modeling influenza epidemics and pandemics: insights into the future of swine flu (h1n1)," *BMC medicine*, vol. 7, no. 1, pp. 1–8, 2009.

[4] G. Mahalakshmi, S. Sridevi, and S. Rajaram, "A survey on forecasting of time series data," in *2016 International Conference on Computing Technologies and Intelligent Data Engineering (ICCTIDE'16)*. IEEE, 2016, pp. 1–8.

[5] M. Qin, Z. Li, and Z. Du, "Red tide time series forecasting by combining arima and deep belief network," *Knowledge-Based Systems*, vol. 125, pp. 39–52, 2017.

[6] Z. Zhao, W. Chen, X. Wu, P. C. Chen, and J. Liu, "Lstm network: a deep learning approach for short-term traffic forecast," *IET Intelligent Transport Systems*, vol. 11, no. 2, pp. 68–75, 2017.

[7] Y. Wu and H. Tan, "Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework," *arXiv preprint arXiv:1612.01022*, 2016.

[8] E. M. de Oliveira and F. L. C. Oliveira, "Forecasting mid-long term electric energy consumption through bagging arima and exponential smoothing methods," *Energy*, vol. 144, pp. 776–788, 2018.

[9] M. Längkvist, L. Karlsson, and A. Loutfi, "A review of unsupervised feature learning and deep learning for time-series modeling," *Pattern Recognition Letters*, vol. 42, pp. 11–24, 2014.

[10] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," 1999.

[11] K. Fukushima, "Neural network model for a mechanism of pattern recognition unaffected by shift in position-neocognitron," *IEICE Technical Report, A*, vol. 62, no. 10, pp. 658–665, 1979.

[12] X. Yao, "Evolving artificial neural networks," *Proceedings of the IEEE*, vol. 87, no. 9, pp. 1423–1447, 1999.

[13] D. Benvenuto, M. Giovanetti, L. Vassallo, S. Angeletti, and M. Ciccozzi, "Application of the arima model on the covid-2019 epidemic dataset," *Data in brief*, p. 105340, 2020.

[14] E. Dong, H. Du, and L. Gardner, "An interactive web-based dashboard to track covid-19 in real time," *The Lancet infectious diseases*, vol. 20, no. 5, pp. 533–534, 2020.

[15] Z. Yang, Z. Zeng, K. Wang, S.-S. Wong, W. Liang, M. Zanin, P. Liu, X. Cao, Z. Gao, Z. Mai *et al.*, "Modified seir and ai prediction of the epidemics trend of covid-19 in china under public health interventions," *Journal of Thoracic Disease*, vol. 12, no. 3, p. 165, 2020.

[16] C.-J. Huang, Y.-H. Chen, Y. Ma, and P.-H. Kuo, "Multiple-input deep convolutional neural network model for covid-19 forecasting in china," *medRxiv*, 2020.

[17] A. Tomar and N. Gupta, "Prediction for the spread of covid-19 in india and effectiveness of preventive measures," *Science of The Total Environment*, p. 138762, 2020.

[18] V. K. R. Chimmula and L. Zhang, "Time series forecasting of covid-19 transmission in canada using lstm networks," *Chaos, Solitons & Fractals*, p. 109864, 2020.

[19] M. Azarafza, M. Azarafza, and J. Tanha, "Covid-19 infection forecasting based on deep learning in iran," *medRxiv*, 2020.

[20] A. Barman, "Time series analysis and forecasting of covid-19 cases using lstm and arima models," *arXiv preprint arXiv:2006.13852*, 2020.

[21] N. Yudistira, "Covid-19 growth prediction using multivariate long short term memory," *arXiv preprint arXiv:2005.04809*, 2020.

[22] H. Salpekar *et al.*, "Design and implementation of mobile application for crop yield prediction using machine learning," in *2019 Global Conference for Advancement in Technology (GCAT)*. IEEE, 2019, pp. 1–6.

[23] A. Sagheer and M. Kotb, "Time series forecasting of petroleum production using deep lstm recurrent networks," *Neurocomputing*, vol. 323, pp. 203–213, 2019.

[24] S. Selvin, R. Vinayakumar, E. Gopalakrishnan, V. K. Menon, and K. Soman, "Stock price prediction using lstm, rnn and cnn-sliding window model," in *2017 international conference on advances in computing, communications and informatics (icacci)*. IEEE, 2017, pp. 1643–1647.

[25] R. M. Sakia, "The box-cox transformation technique: a review," *Journal of the Royal Statistical Society: Series D (The Statistician)*, vol. 41, no. 2, pp. 169–178, 1992.

[26] K. Da, "A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[27] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," *arXiv preprint arXiv:1602.07261*, 2016.

[28] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size," *arXiv preprint arXiv:1602.07360*, 2016.

[29] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.