Simultaneous Time Series Forecasting on the World's COVID-19 Daily Vaccinations

[Extended Abstract]

Konstantinos Georgiou
The University of Tennessee, Knoxville
Knoxville, TN, United States
kgeorgio@vols.utk.edu

Michela Taufer
The University of Tennessee, Knoxville
Knoxville, TN, United States
taufer@utk.edu

ABSTRACT

The global distribution of COVID-19 vaccines is one of the most challenging tasks the modern health industry has ever faced. Being able to predict the number of vaccinations of a country for the next week can be vital information for adapting its policy response. We present a pipeline that creates a model based on all countries' historical vaccination data and attempts to predict their next ten days of daily vaccinations by utilizing their underlying relationships. We use *Encoder-Decoder Long Short-Term Memory Networks* with walk-forward validation and evaluate the results using mean, per-country, and per-date $RMSE$.

CCS Concepts

•Computing methodologies → Neural networks; •Applied computing → Health care information systems;

Keywords

Multivariate Time Series; Time Series Forecasting; Recurrent Neural Networks; Long-Short Term Memory Networks; COVID-19

1. MOTIVATION

The COVID-19 pandemic is one of the most impactful events of recent human history with profound effects on the health of billions and the economies of almost all of the world's countries. The imperative need for the rapid development and global distribution of COVID-19 vaccines generated one of the most complex tasks modern public health history has faced. The urgency and significance of this task have attracted the interest of a significant portion of the scientific community.

One insight that can be found very useful, is the prediction of a country's future daily vaccinations. By doing so, we could identify potential future reductions of the rate of some countries' vaccinations, which can give their governments the opportunity to prevent unexpected complications.

2. CONTRIBUTIONS

Many studies have been made on the COVID-19 dataset, but most of them are either comparative analyses between two or more countries' historical vaccination records, or attempts to predict the number of future vaccinations by looking at one country at a time. We present a method that utilizes all countries' historical vaccination data in order to identify causal relationships between them and make simultaneous predictions on their next ten days of daily vaccinations per hundred citizens. We first perform a thorough preprocessing to infer the missing values of the dataset, then use an external dataset to recalculate the per hundred people values of the vaccination columns, we train an *Encoder-Decoder Long Short-Term Memory Network* (LSTM), and finally evaluate the predictions using the average, per-country, and per-date $RMSE$.

The rest of the paper is organized as follows: Section 3 describes the dataset, preprocessing, and evaluation techniques used in this study; Section 4 describes and discusses the results; Section 5 summarizes our findings and proposes future improvements.

3. METHODOLOGY

In this section, we describe the two datasets we used and the framework we developed to predict the countries' future daily vaccinations per hundred citizens.

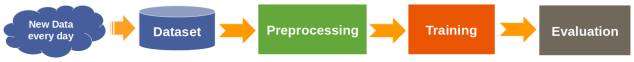

Figure 1: General Pipeline.

3.1 Dataset

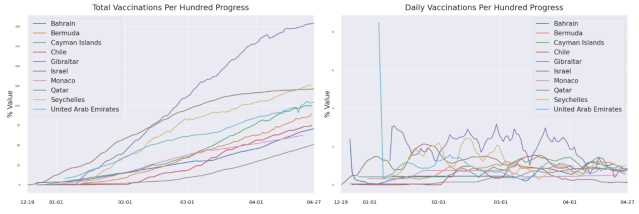

Figure 2: Top 10 Countries in terms of total (left) and daily (right) vaccinations per hundred citizens.

The main data source used, which is the COVID-19 World Vaccination Progress dataset [1], contains information about the daily and total vaccinations of 193 different countries over 135 different dates. The data are being collected almost daily and as of writing this paper, the dataset has 14230 rows

and 15 different features. You can see two example features of the dataset in Figure 2.

Additionally, a secondary dataset was used, namely, DataBank - World Development Indicators [2]. This data source contains numerous useful static features about the world's countries, such as health expenditure per GDP and hospital beds per 1000. In our analysis, only the country population was used to help us recalculate the per hundred citizen values of the columns, we inferred their missing values.

3.2 Preprocessing

We are performing three distinct preprocessing steps on the vaccination dataset. Before starting, we select to keep nine out of the fifteen columns and we select one of them (the *date*) as the index. The main features used in our pipeline are the *daily vaccinations*, the *total vaccinations*, the number of *people vaccinated*, the number of *people fully vaccinated*, and their respective per hundred citizen values as well as the *country ISO code* and the *date*. We made predictions on the *daily vaccinations per hundred citizen* values. The numerical columns used for training are shown in Table 1.

Absolute Columns	Percentage Columns
Daily Vaccinations	Daily Vaccinations per 100
People Vacc.	People Vacc. per 100
People Fully Vacc.	People Fully Vacc. per 100
Total Vaccinations	Total Vaccinations per 100

Table 1: The columns we kept before starting the preprocessing steps.

3.2.1 Fixing the Missing Values

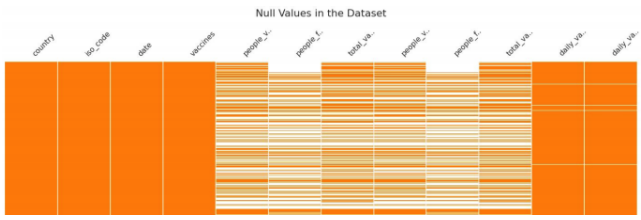

Figure 3: Null values in the dataset's features. The white gaps represent the null values.

In the preprocessing phase of the pipeline, we first fix the null values of the *daily vaccinations* (daily), *total vaccinations* (total), *people vaccinated* (vacc), and *people fully vaccinated* (f_vacc) columns. To do so, we define the equations (1 - 4).

$$daily_{current} = total_{current} - total_{previous} \quad (1)$$
$$total_{current} = vacc_{current} + f\_vacc_{current} \quad (2)$$
$$total_{current} = total_{previous} + daily_{current} \quad (3)$$
$$total_{current} = total_{next} - daily_{next} \quad (4)$$

Using these equations repeatedly until the number of missing values does not change and then inferring the rest of the values by averaging the next and the previous values, we are able to infer the 100% of the missing values. We should note that the above operations are being performed separately for each country's records.

To recalculate the per hundred citizen values of the columns we fixed, we need the population of each country. We load an external dataset [2] that has this information and preprocess it. After ensuring that the two datasets have enough countries in common, we generate the per hundred citizen columns, dividing each of the columns we already fixed by the corresponding population.

3.2.2 Cleaning and Normalizing

As we are going to make predictions based on the underlying relationships between the countries, we must not train on recent dates for which there are no data for some countries yet. We identify the number of countries with valid records one month before the latest record, and we drop the most recent dates whose number of countries with valid records is less than 90% of that number.

All the features we are going to use for the training are numerical, so we scale them before feeding them to the network. We saw that scaling them in the $[0, 1]$ range causes the efficiency of the model to drop compared to using larger ranges. This indicates that the values of the percentage features are too low for the float precision of our model to handle. During hyper-parameter tuning, we found that the optimal range was $[0, 1000000]$. The normalization was performed on a per-country basis.

3.2.3 Reshaping the Dataset

Next, we create a new instance of each column we kept for each country present in the dataset. This results in transforming the data so that the number of rows is equal to the number of dates, and the number of columns is equal to the number of countries times the number of columns. This enables us to feed the daily records of all countries simultaneously for each time step and thus allow it to find correlations between them. This operation transformed the dataset's shape of *11882 rows* and *10 columns* into a shape of *126 rows* and *1409 columns*. The 10^{th} column before and the 1409^{th} after reshaping is the date. The result of this transformation is shown in Table 2.

Country 1 Columns	..	Country n Columns
Daily Vacc. C_1	..	Daily Vacc. C_n
Vacc. C_1	..	Vacc. C_n
Fully Vacc. C_1	..	Fully Vacc. C_n
Total Vacc. C_1	..	Total Vacc. C_n
% Daily Vacc. C_1	..	% Daily Vacc. C_n
% Vacc. C_1	..	% Vacc. C_n
% F. Vacc. C_1	..	% F. Vacc. C_n
% Total Vacc. C_1	..	% Total Vacc. C_n

Table 2: The columns after the preprocessing steps. There are 1409 columns in total.

Finally, we reshape again the dataset by grouping the rows/dates in 10-day time windows. We have 126 days of data entries in total, so we create 12 10-day groups and drop the last 6 days of it. We do this because we want to train our network using *Walk-Forward Validation*.

3.3 Training

Using Keras [3] and Tensorflow [4], we created a sequential model with two *Long Short-Term Memory* layers and

two *Dense* Layers. The input shape of the network is *(time window size, # features) = (10, 1408)* because we will train in 10-day time steps and we are not including the date column during training. The first and the second layer have *800* units each, while the two *Dense* layers have *400* and *1408* (the number of features), respectively. The implementation of the *Walk-Forward Validation* is being handled by Keras' *TimeDistributed* layer. The model is shown in Figure 4.

We use the *ADAM* optimizer [5] and the *root mean squared error* (*RMSE*) as our *loss function*. Before training, we split the dataset into a train (80% - nine 10-day groups) and a test (20% - three 10-day groups) set. Ultimately, we use the trained model to generate a prediction dataset with the same size as the test set.

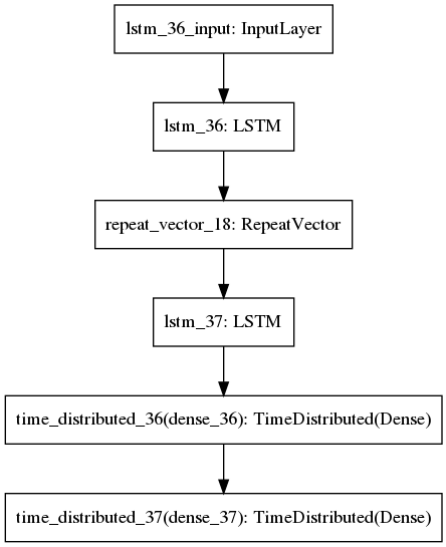

Figure 4: The network we trained our dataset on.

3.4 Evaluation

To evaluate our results, we need to reconstruct the test and prediction sets in the original format. Both sets are three-dimensional, where the first dimension is the number of 10-day windows, the second is the number of days in each window (which is always 10) and the last one is the number of features. First, we flatten the first and the second dimension, so from the shape $(3, 10, 1408)$ we end up with a $(30, 1408)$ shape. Then, we reshape again the two sets back to the original format where we had multiple date entries (one for each country) and only 9 columns plus the *date* column. We end up with a shape of $(4598, 10)$.

We only want to predict the *daily vaccinations per hundred citizens* column, so we only keep it and the *date* column. To evaluate the results, we use the *RMSE* and we calculate: the average *RMSE* over the whole dataset, the average *RMSE* per country (170 values), and the average *RMSE* per date (30 values).

4. RESULTS

We trained the deep neural network on the first 90 days of the dataset and made predictions on the last 30 days of it. Even though our model's output includes all the input features, since we only want to predict the *daily vaccinations per hundred citizens*, we make our evaluation on that column. By calculating the *RMSE* between the test set and our predictions for each data entry, we got an average of 0.24617. The average *RMSE* per date and per country are shown in Figures 5 and 6, respectively. Out of the 170 countries, 164 have mean *RMSE* less than 1.0, 153 less than 0.5, and 73 less than 0.1. The minimum value is 0.00023 and the maximum is 5.5133. Out of the 30 dates, 25 have mean *RMSE* less than 0.30, 15 less than 0.25, and 11 less than 0.20. The minimum value is 0.16928 and the maximum is 0.3518. Additionally, in Figure 7, we show the true versus predicted values for 12 different countries.

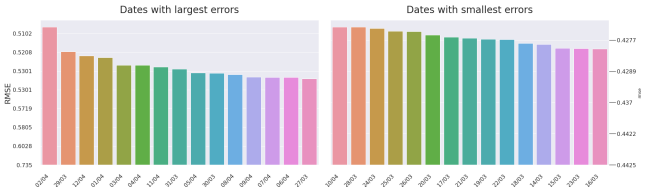

Figure 5: Dates with the largest (left) and smallest (right) errors.

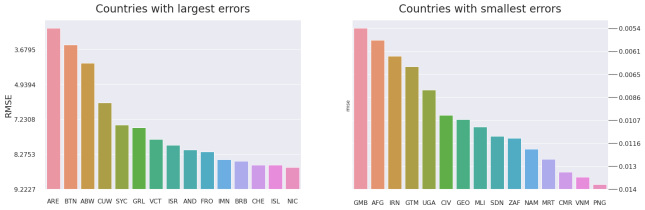

Figure 6: Countries with the largest (left) and smallest (right) errors.

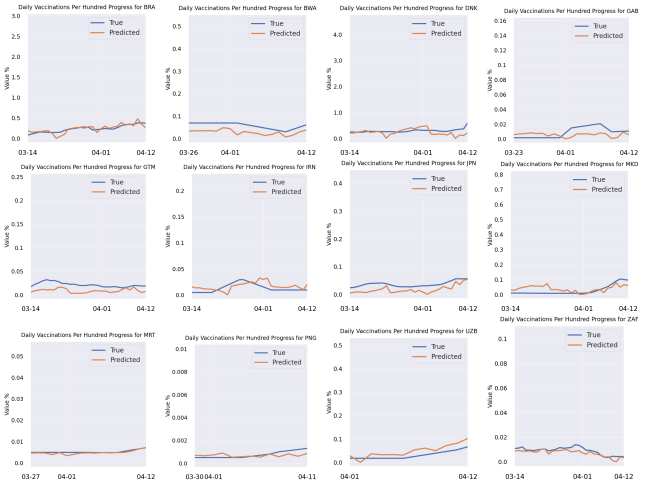

Figure 7: The true versus the predicted daily vaccinations per hundred for 12 different countries.

5. CONCLUSIONS AND FUTURE WORK

Our method showcases the type of preprocessing needed for making the *COVID-19 vaccination process* dataset suit-

able to be used in a time-series analysis pipeline and a proof-of-concept multivariate network that can be trained on this dataset with 10-day training windows. We also showed that it is possible to predict the *daily vaccinations per hundred citizens* column with satisfactory results using *Multivariate Long Short-term Memory Networks* showing that such networks can be trained on small datasets like the one we used.

Future orientations for this work include trying custom *loss functions* for training, incorporating static features of the countries such as the *health expenditure per GDP*, and training using one *LSTM* network per country, combining their weights using the Functional API of Tensorflow at the end.

6. REFERENCES

[1] Covid-19 world vaccination progress. https://www.kaggle.com/gpreda/covid-world-vaccination-progress. Accessed: 2021-04-30.

[2] Databank - world development indicators. https://databank.worldbank.org/source/world-development-indicators. Accessed: 2021-04-30.

[3] A deep learning api written in python, running on top of the machine learning platform tensorflow. https://github.com/keras-team/keras.

[4] An open source machine learning framework. https://github.com/tensorflow/tensorflow.

[5] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015.