SNS Final Project Report

Covid-19 cumulative cases Inference

Menelaos Kaskiris

SN: 17039570

1. Introduction

Since the beginning of 2020 the world is facing a global pandemic which has taken the world by storm. The virus belongs in the coronavirus family and has been named SARS-Cov-2 or covid-19. The virus is similar to the SARS (severe acute respiratory syndrome) outbreak in China 2002 which was contained relatively effectively to the region with around 8000 infected 800 dead.

Covid-19 on the other hand is transmitted more easily with around 20% of the infected needing hospitalization. This has caused over 120 million confirmed cases worldwide and almost 3 million dead. Even though the death rate of the virus is relatively low the high transmission and hospitalization rate has put a huge strain on health care around the world.

The need for accurate forecasting of cases, hospitalizations, Intensive care unit (ICU) patients etc. is critical for the correct allocation of resources as well as future planning for decision makers.

A variety of forecasting tools exist, including statistical and machine learning models. Statistical models include Vector Autoregressions (VAR), Auto Regressive Integrated Moving Average (ARIMA) and Facebook Prophet, which use statistical methods to predict future values of a time series. Machine learning models focus on neural networks, specifically recurrent neural networks (RNN) as a base for the models for their ability to 'remember' past results in the time series and is specifically designed for such a task.

This paper will focus on machine learning neural networks and specifically an RNN layer with the 'Long Short-Term memory' architecture, which are specialized RNN modules that can learn long-term temporal dependencies. This makes them perfect for modelling time series and forecasting problems. A univariate model was trained to predict the cumulative number of cases for the island of Cyprus using cumulative case data from previous days.

2. Datasets

Two major datasets were used in this experiment. The first dataset came from 'Our world in data[1]' and describes the current covid-19 pandemic It consists of 71911 measurements, 54 features, three location features, and a date column. The dataset contains measurements for 425 different dates.

The second dataset was found from Kaggle[2] and describes the 2002 SARS outbreak. It consists of 2538 measurements, three numerical features, a date column, and a location column. The dataset contains measurements for 96 different dates.

Both datasets were download and stored locally in csv files.

3. Pre processing

The date column was converted to datetime format using the pandas method 'to_datetime()'. All feature columns were dropped apart from the cumulative total cases. For the Covid-19 dataset all countries except the island of Cyprus were dropped since that is the only place we will be

[1] https://ourworldindata.org/coronavirus-testing
[2] https://www.kaggle.com/imdevskp/sars-outbreak-2003-complete-dataset

forecasting. For the SARS dataset all countries except the country of Chine were dropped since it is where most of the cases for the pandemic occurred.

Before feeding the data into the networks for training the MinMaxScaler() method available from the scikit-learn package was used to transform all measurements in the range [0,1]. This was done to prevent very big values from overshadowing smaller ones.

The data was split into training, validation, and testing set. The first 90% of the data was used in the training set, and the rest 10% was used as a testing set. This split allowed around 35 days in the testing set to test the degradation of the predictions of our model.

The TimeseriesGenerator method available from Keras was used to transform the training and validation set into the appropriate format to be used as an input to the Neural Network.

4. Exploratory Data Analysis

Comparing the progression of the viruses in total cumulative cases we can see some similarities as shown in FIGURE 1. We can see the SARS-2002 contained within a few months of it first appearing. On the other hand, covid-19 has been spreading for a year and is just barely slowing down. The spreading of the virus did not reach its peak in Cyprus until almost 5 months after the fist case but the rate at which was growing resembles the SARS outbreak. Training the model on the SARS dataset first may allow us to predict how the pandemic will end.

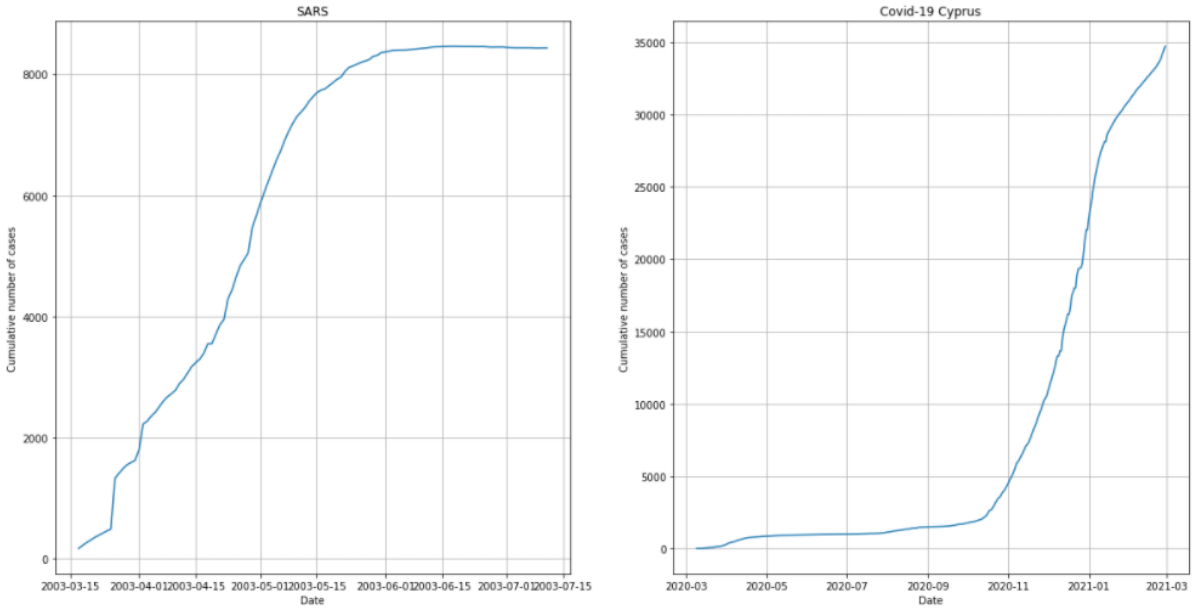

Figure 1: SARS and Covid-19 case comparison

5. Method

To determine the optimal hyperparameters and architecture for our network we will be iteratively changing a single parameter and testing the results for comparison. All tests will be run on the covid-19 dataset. The parameters that will be optimized are the activation function, Day Lag and the number of layers and nodes in the network. Once the optimal model is obtained the model will be re-initialized with random weights will be trained on the SARS dataset. The first layer of the network will then be frozen, and the network will be trained on the whole covid-19 dataset and results will be inferred for the future.

Ideally each network configuration would be fit multiple times to reduce the effect of the stochastic nature of the algorithm and over a big epoch number to let the network fully converge but due to time constrains each configuration was plotted once over 25 epochs. This was enough for the training loss to stabilize within $1 * 10^{-3}$ variation.

5.1 Tests

5.1.1 Activation Function

Two activation function conditions were tested

1. Rectified Linear Unit (relu)
2. No activation functions

5.1.2 Day lag

Day Lag refers to the days prior to the inference day the model will be provided. Three conditions were tested of five, 10 and 15 days.

5.1.3 Layers and Nodes

Five different models will be tested outlined below. The numbers after the name of the layer indicate the number of nodes in the layer. The '->' sign indicates next layer. Two different layers were used in each network. A series of LSTM layers were used with a Dense layer as the output layer of the network:

1. LSTM 256->Dense
2. LSTM 256->128->Dense
3. LSTM 256->128->64->Dense
4. LSTM 256->128->64->32->16->Dense
5. LSTM 256->128->64->32->16->8->Dense

For every LSTM layer apart from the one before the Dense layer the hyperparameter 'return_sequences' was tuned to True to enable each of the layers of the LSTM to learn from the whole sequence of the previous layer.

5.2 Constants

5.2.1 Optimizer

The 'Adam' optimizer was used since it has been shown to perform well for most applications and without requiring specific hyperparameters, it is relatively computationally efficient and as not as prone to be stuck in local minima as Stochastic Gradient Descent (SDG)[3].

5.2.2 Loss Function

The loss function used to train the networks was 'Mean Squared Error'

5.2.3 Epochs

The models were train on 25 epochs which allowed enough time for the training loss to stabilize within 1x10^-3 variation.

5.2.3 Batch size

A batch size of 1 was used throughout the experiment due to the low number of measurements

[3] https://ruder.io/optimizing-gradient-descent/index.html

5.2.4 Inference

Inference for each model was done by initially using the last n values of the training set (where n is the day lag) to predict one day into the future using the predict() method available in Keras. The prediction was appended to the input to the model and the first entry (oldest day) was removed. This process continued iteratively until a prediction was made for every member of the testing set.

5.2.5 Transfer Learning

After obtaining the best model using only covid-19 data the model will be recompiled to randomize the weights and trained on the whole SARS dataset. The first layer of the model will then be frozen and

6. Results

6.1 Activation function

Looking at the train loss graphs in figures 1 and 2 it can be seen the networks are learning.

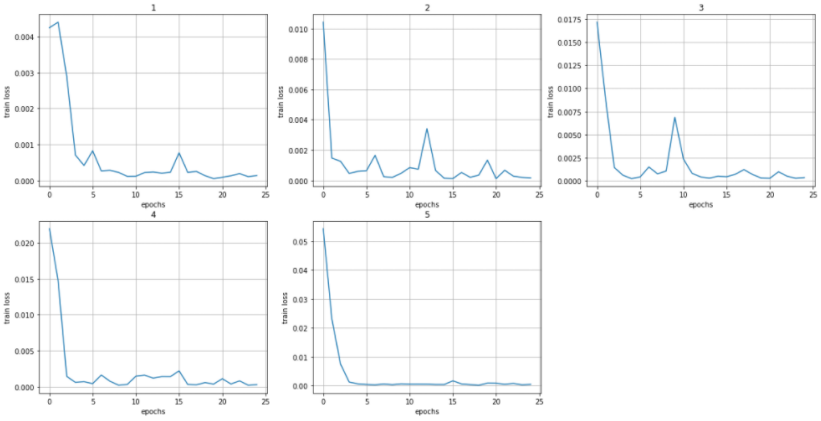

Figure 2: RelU activation

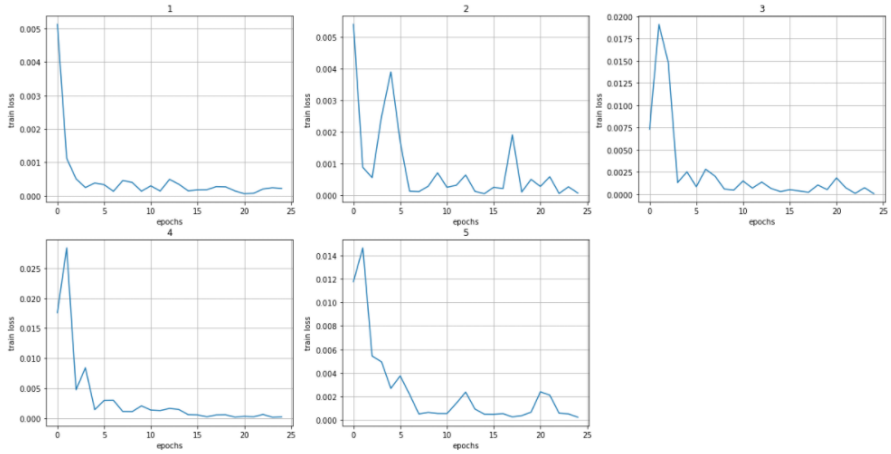

Figure 3: No activation function

To assess the quality of the networks predictions were made for every single network and accuracy result we calculated using equation 1. Figure 5 shows the predictions of the networks for all the configurations while figure 6 shows the accuracy of the prediction. The blue line labelled 'Truth' indicates the true value of the cases while the numbers 1-5 refer to the network architectures indicated in 5.1.3.

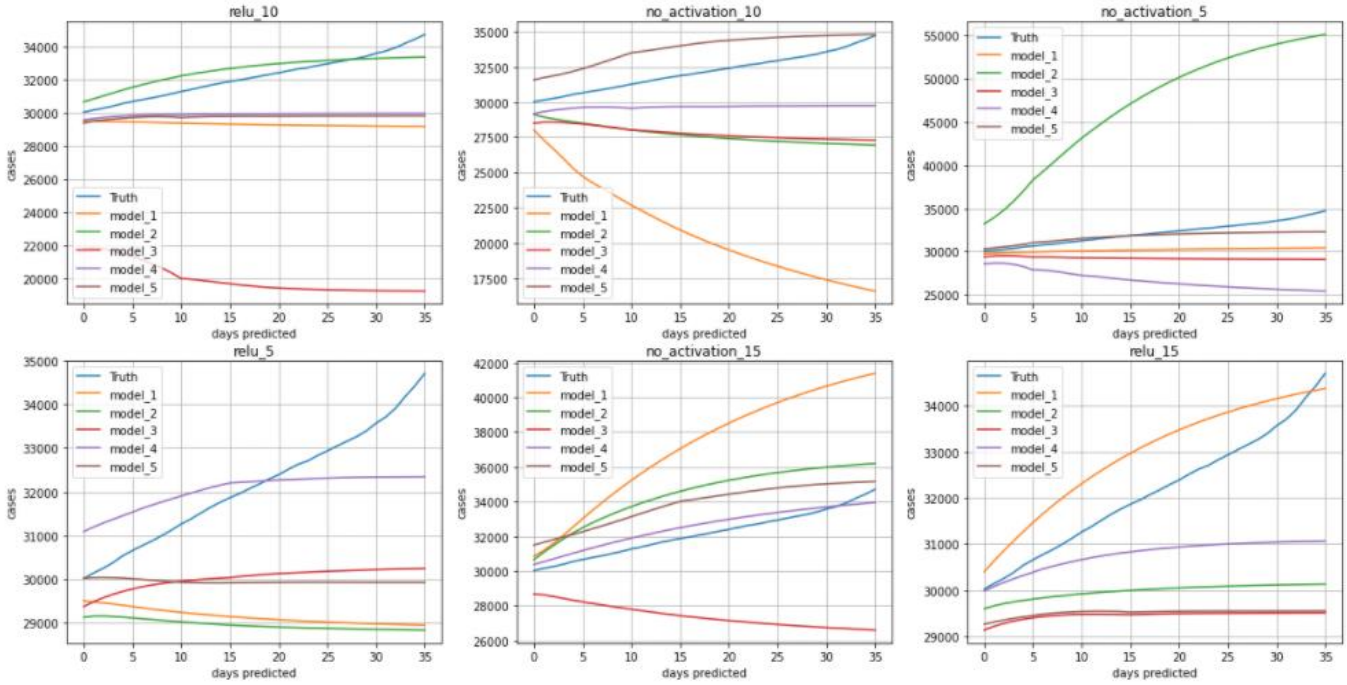

Figure 5: Predictions

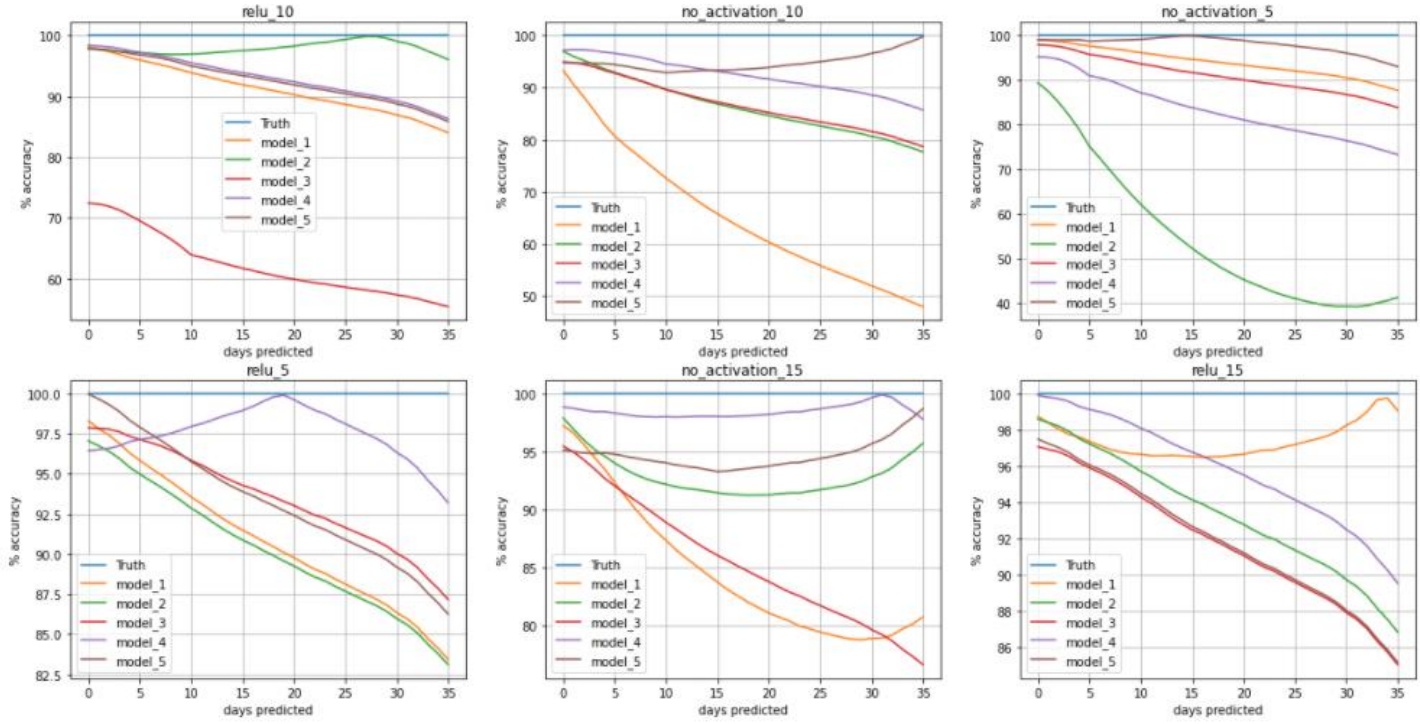

Figure 4:accuracy scores

The best model was chosen based on the biggest average accuracy over all the models, the standard deviation of the accuracy and the minimum accuracy over the 35 days predicted. Three metrics were selected to get the most well-rounded model and to account for the stochastic nature of the algorithms since we didn't fit multiple models. Table 1 shows the best models along with their parameters. The model with no activation function, 15-day lag and five-layer depth (model four from section 5.1.3) was chosen to be used for transfer learning.

Table 1: Best models

Activation	Day lag	Model number	Average acc	Standard deviation	Minimum accuracy
Relu	10	2	97	1	96
RelU	10	5	94	1.7	93
RelU	5	5	98	1.7	93
None	5	4	97	1.5	93
None	15	4	98	0.5	97.8
None	15	1	97	1	96.5

The accuracy scores of the model per day are shown in table 2. A complete list of the accuracy for all 35 days can be found in the iPython notebook.

Table 2: daily accuracy

Day	Accuracy
1	98.8
2	98.7
3	98.5
4	98.4
5	98.3
6	98.3
7	98

6.4 Transfer Learning

The model was made to predict 80 days into the future. As it can be seen from figure 6 the model utilizing transfer learning seems to keener in flattening the curve, even though it displays a sharper gradient at the start of its prediction. On the other hand, the model not utilizing transfer learning seems to be a lot gentler keeps increasing up to 40,000 cases.

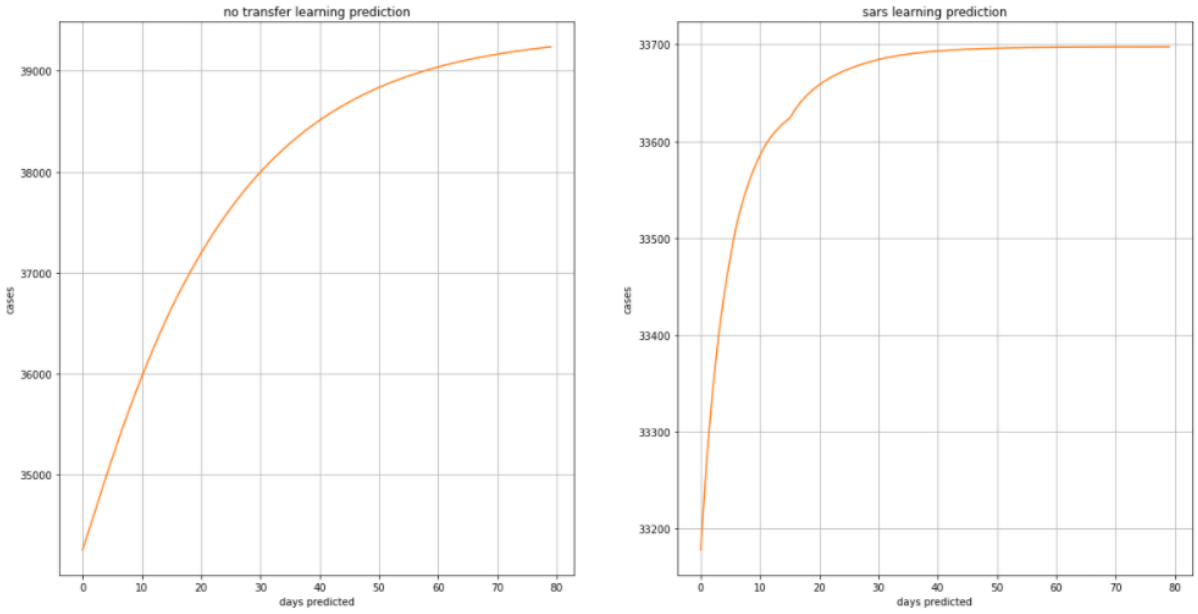

Figure 6: model predictions without and with transfer learning

7. Conclusion

Many models performed exceptionally well in forecasting the cumulative number of cases, without a huge number of data points to train on. Ultimately the best one was the second deepest model, and 4 out of the top 6 models were either 5-layer or 6-layer deep networks. The other two parameters looked at were equally represented in the to networks so it may be that they do not affect the inferring capabilities of the network.

8. References

[1] Dataset: Hasell, J., Mathieu, E., Beltekian, D. et al. A cross-country database of COVID-19 testing. Sci Data 7, 345 (2020)