A Deep Learning Approach to Forecast SARS-CoV-2 on the Peruvian Coast

I. Luis Aguilar[1(✉)], Miguel Ibáñez-Reluz[2], Juan C. Z. Aguilar[3], Elí W. Zavaleta-Aguilar[4], and L. Antonio Aguilar[5]

[1] Department of Mathematics, National University of Piura Castilla s/n, Piura, Peru
laguilari@unp.edu.pe
[2] Medicine Faculty, Cesar Vallejo University, Av. Victor Larco 1770, Trujillo, Peru
mibanezr@ucvvirtual.edu.pe
[3] Department of Mathematics and Statistics, Universidade Federal de São João del-Rei C.P. 110, 36301-160 São João del-Rei, MG, Brazil
jaguilar@ufsj.edu.br
[4] São Paulo State University (Unesp), Campus of Itapeva Rua Geraldo Alckmin 519, 18409-010 Itapeva, SP, Brazil
eli.zavaleta@unesp.br
[5] Artificial Intelligent Research, KapAITech Research Group, Condominio Sol de Chan-Chan, Trujillo, Peru
antonio@kapaitech.com

Abstract. The current spreading of the SARS-CoV-2 pandemic had put all the scientific community in alert. Even in the presence of different vaccines, the active virus still represents a global challenge. Due to its rapid spreading and uncertain nature, having the ability to forecast its dynamics becomes a necessary tool in the development of fast and efficient health policies. This study implements a temporal convolutional neural network (TCN), trained with the open covid-19 data set sourced by the Health Ministry of Peru (MINSA) on the Peruvian coast. In order to obtain a robust model, the data was divided into validation and training sets, without overlapping. Using the validation set the model architecture and hyper-parameters were found with Bayesian optimization. Using the optimal configuration the TCN was trained with a test and forecasting window of 15 days ahead. Predictions on available data were made from March 06, 2020 until April 13, 2021, whereas forecasting from April 14 to April 29, 2021. In order to account for uncertainty, the TCN estimated the 5%, 50% and 95% prediction quantiles. Evaluation was made using the MAE, MAD, MSLE, RMSLE and PICP metrics. Results suggested some variations in the data distribution. Test results shown an improvement of 24.241, 0.704 and 0.422 for the MAD, MSLE and RMSLE metrics respectively. Finally, the prediction interval analysis shown an average of 97.886% and 97.778% obtained by the model in the train and test partitions.

Keywords: Deep learning · SARS-CoV-2 · Temporal convolutional neural networks · Time series data

© Springer Nature Switzerland AG 2021
O. Gervasi et al. (Eds.): ICCSA 2021, LNCS 12951, pp. 304–319, 2021.
https://doi.org/10.1007/978-3-030-86970-0_22

1 Introduction

On December 2019 a novel virus was reported in Wuhan city, Hubei province, China. It was latter identified as SARS-CoV-2; a virus belonging to a family of coronavirus strains responsible for causing severe acute respiratory syndrome [7]. Initial clinical studies revealed the presence of thrombocytopenia, elevated D-dimer, prolonged time, and disseminated intravascular coagulation in SARS-CoV-2 patients [10]. Also, comorbidities were associated as high risk factors in SARS-CoV-2 patients [12]. SARS-CoV-2 shown a fast infection rate which resulted in a global pandemic. In order to better understand the virus spread, researchers over the world began to implement a wide variety of models. These initial models required relatively small data sets. As such, the SIR model offered a fast approach to estimate the SARS-CoV-2 infection tendencies. The SIR [2] model is composed by a system of differential equations involving three epidemiological parameters: susceptible (S), infectious (I) and removed (R). Other variants, like the SEIR [2] and SIRD [3] included additional parameters over the base SIR model such as exposition (E) and deceased (D) cases respectively.

In the following months, new data set repositories [9] allowed the application of deep learning models. Due to the evolving dynamics observed in SARS-CoV-2 cases, a time series approach was used. Therefore, recurrent neural networks (RNN) like long short term memory (LSTM) [15] and its variants were applied. In [8] a LSTM model was used to forecast SARS-CoV-2 cases in Canada. A 80-20% data split strategy was implemented for train and test respectively. The RMSE metric was used to evaluate the model performance. The trained LSTM model was able to estimate a peak in cases within a two week period.

Besides recurrent models, other non-recurrent architectures have shown successfully results in forecast SARS-CoV-2 cases. For example, in [25] a system of two neural networks were used in real time to estimate SARS-CoV-2 trends at country and regional levels. These models used a window of 12 and 8 days for each level respectively. The models were built using a global data set repository [9]. Margin error variations shown consistent accuracy results in several countries, such as Peru and Brazil.

In a comparative study, a Variational auto encoder (VAE) shown better forecasting performance than four recurrent architectures including a: RNN, GRU, LSTM and Bi-LSTM [26]. In a similar fashion, convolutional neural networks (CNN) also reported good performance in forecasting SARS-CoV-2 trends. In [19] a CNN model was applied to forecast the daily, cumulative, recoveries and deceased SARS-CoV-2 cases alongside with hospitalizations (with or without artificial ventilation). The study was executed on France at regional and national levels. Evaluation results for the CNN model reported by the MAE, RMSE, and R2 metrics shown a consistent forecasting performance. Furthermore, CNN variations such as temporal convolutional neural networks (TCN) have been able to forecast using prediction intervals with a multi output setting [19].

Lastly Bayesian optimization techniques have shown a positive impact in learning, when applied to LSTM and CNN models. In [1] a multi-head attention, LSTM and CNN models were developed to predict confirmed SARS-CoV-2 cases.

This study was applied to a short and long horizon using two data sets of several countries such as Peru and Brazil. CNN results in the short horizon were superior than the LSTM model according with the SMAPE, MAPE and RMSE metrics. In the long horizon, the CNN model was superior in Peru and Brazil.

As observed, deep learning models are capable of achieve good forecasting performance across different geographical areas. In this study, a TCN deep learning model is developed to forecast daily SARS-CoV-2 cases. The data used encompass the 3 geographical regions of Peru. Among them, the Peruvian coast was selected due to its high population density areas in contrast with the highlands and amazon rain-forest [16]. The model architecture and hyper-parameters were optimized using a Bayesian approach. To avoid any data leaking. a hold out method was used. Our model was able to predict from March 30 to April 16, 2021 and forecasting 15 days ahead of the available data until April 27, 2021. Finally, a set of evaluation metrics were applied to evaluate the model performance. The following sections will describe the study in more detail.

2 Temporal Convolutional Neural Networks (TCN)

A Temporal convolutional neural network (TCN) is a variation of a CNN model, which can deal with sequence data. A TCN model applies the convolution operation over an input sequence s. Since s contains events in a period of time, convolutions are applied over the time dimension. A TCN is able to estimate future events of s based on past information. To achieve this, a TCN model modifies the standard convolution operation present in a CNN with casual and dilated convolutions. To avoid vanish gradients, skip residual connections are usually applied between TCN layers.

In a casual convolution, the kernel operation maintains the natural order of the input sequence s. This process is illustrated in Fig. 1a, where instead of applying the convolution directly over s, the kernel is moved from left to right. This process forces the model to only relay on past data to make its predictions. This mechanism also prevents any data leaking from the future, thus the name casual. However, a casual convolution is unable to maintain a long horizon history, which is essential in order to capture past patterns. Therefore, another variation called dilated convolutions are applied. Dilated convolution relay on dilatations to increase the receptive field [4]. This effect can be observed in Fig. 1b, where an increasing dilation allows next layers to expand the input information. Formally for an input sequence $s = \{x_0, ..., x_T\}$, where T denotes the sequence length; a casual dilated convolution $\mathcal{F}(s)$, is defined as:

$$\mathcal{F}(s) = (x *_d f)(s) = \sum_{i=1}^{k} f(i) * x_{s-d*i} \tag{1}$$

where the dilation factor d express the amount of variation over the receptive field. Next, a set of kernels f of size k are applied over the elements in s, such that $x_s - d * i$ constrains the process to only considered past information.

The last component of a TCN is a skip residual connection. This mechanism was introduced by [13] in the Residual Network architecture. Skip residual connections create a shortcut in the information flow (gradient). The residual is skipped over a certain number of layers. Then, it is combined in the main information flow. This operation allows to create deeper models and deal with the gradient vanish problem. Figure 1c., shows a standard residual connection scheme.

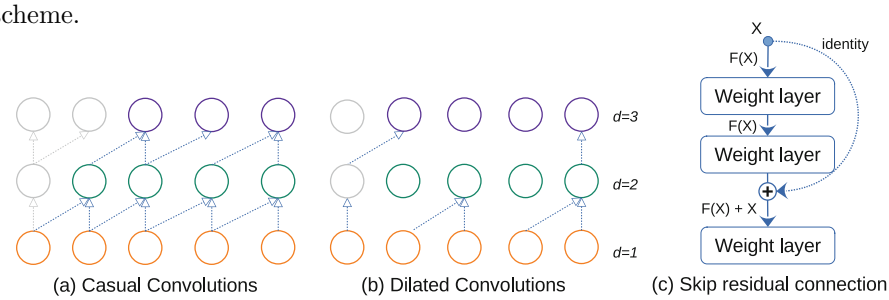

Fig. 1. Mechanisms in a TCN: (a) casual convolutions, (b) dilated convolutions and (c) skip residual connections.

3 Data Set

This study used the open covid-19 data set [18], created by the Ministry of Health in Peru (MINSA). The data set contained the daily reports of SARS-CoV-2 cases across the country. In its raw format, it was composed by 9 features and 155,649 observations, corresponding from March 03, 2020 until April 13, 2021 (**accessed on 16.04.2021**). Some features were not relevant for this study and were removed, therefore, a feature engineering process was applied to create new features which were used by the model. Although the data set had information for 25 regions, only 9 belonging to the coast were selected. These regions included: Lima, Ica, Callao, Piura, Lambayeque, La Libertad, Tumbes, Tacna and Moquegua, with a total of 63,735 samples. The coastal regions were selected due to its high population density [16]. Also, coastal regions shown a much larger number in infections than the highlands and amazon regions. As such determining the infection impact on the coast can help to mitigate the spreading to the remaining regions.

3.1 Feature Engineering

A total of five features were selected from the original covid-19 data set [18]. Missing observations were imputed using rolling windows. Next, a set of 9 features were engineered to capture: time dependencies, cases in lower administrative divisions (province and districts) and a seven day record of past SARS-CoV-2 cases per region. As a last pre-processing step, the data was grouped at regional level. This new data set was used by the TCN model for validation and training, containing a total of 10 features, which are described in Table 1.

Table 1. Feature engineering process description.

Features		Description
Selected	Engineering	
Date	Month	Current month, values from 1 to 12
	Day of year	Ongoing day in the year, values from 1 to 365
	Week	Week day, from 0 to Monday until 6 to Sunday
	Weekday	Week of the year, values from 1 to 52
	Quarter	Current quarter
	Month day	Day of the current month
Region	–	Region name used as ID
Province	I_p	Number of infected provinces in a region
District	I_d	Number of infected districts in a region
	W_7	A seven window with previous cases per region
Cases	–	Daily cases variable used as target

3.2 Data Preparation

Since the obtained data set described in Table 1 was in tabular format, a preparation step was necessary in order to be used by the TCN model. First, W_7 was arranged into a fix sequence, where the eight day was used as target y (Fig. 2a). To obtain a fix sequence, all features were padding with zeros to match the length of the longest sequence, which corresponded to Lima. To keep the time correlation, all features were sorted by date.

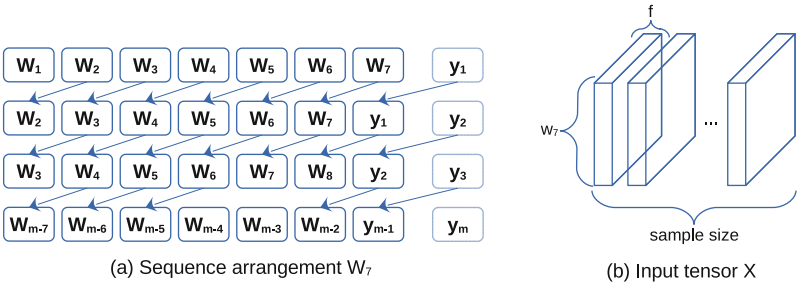

Fig. 2. A sequence window W_7 arrangement (a) was used to represent a past seven-day input to the model, using the eight day as y. Then W_7 (a) was combined with the remaining 9 features, resulting in the X input tensor (b) used by the model.

Next, all features were combined into a 3D input tensor X, with dimensions: $samples \times sequence_7 \times features_{10}$ (Fig. 2b). To normalize the values in X, all features were scaled individually, such that the maximal absolute value of each one was set to 1.0.

3.3 Data Partition

After completing the previous process (subsection 3.2), the data was divided into validation and training sets without overlapping date periods. This avoided any data leaking between the training and validation sets, which could lead to overfitting. Then, each set was split into training and test partitions respectively. In both data sets, the training partition was used for learning, while the test partition for evaluation. Model architecture and hyper-parameters optimization were performed through Bayesian optimization on the validation data set. Table 2 shown the validation data set in more detail.

Table 2. Validation data set division used to build and optimize the TCN model.

Partition	Features	Samples	Data range	Days
Training	10	3366	Mar. 06, 2020–Mar. 14, 2021	374
Test	10	135	Mar. 15, 2021–Mar. 29, 2021	15

The training set was used to create the final TCN model, using the optimal hyper-parameter configuration found in the validation data set. Table 3 shown the training set details.

Table 3. Training data set division used to train the TCN model.

Partition	Features	Samples	Data range	Days
Training	10	3501	Mar. 06, 2020–Mar. 29, 2021	389
Test	10	135	Mar. 30, 2021–Apr. 13, 2021	15

A 15 day ahead window was selected for test and forecasting. We based this value according with the average 14 day incubation period of SARS-CoV-2 [7]. Next, in order to account for possible shifting variations in incubation periods, an extra day was added. This allowed the model to estimate SARS-CoV-2 variations more consistently across regions.

4 TCN Model

A TCN model was developed to estimate the daily SARS-CoV-2 cases in the Peruvian coast. Estimations were categorized into predictions and forecasting. Predictions were made using all the available data set [18], whereas forecasting used a multi-step approach without any available data. A set of 9 regions were selected from the data set [18] including: Lima, Ica, Callao, Piura, Lambayeque, La Libertad, Tumbes, Tacna and Moquegua. A predictive window of 15 days ahead were used for prediction and forecasting. The model was built and optimized using a Bayesian approach. The following sections will describe the model architecture as well as the optimization process in more detail.

4.1 Model Architecture

In order to determine the total number of casual convolution (DC-Conv) layers, an empirical approach was applied. In this setting, a single DC-Conv layer was added to the base model until no further improvements were reported on the validation set (Table 3). This process was repeated until the optimal number of DC-Conv layers were set to 5. Each DC-Conv layer applied the kernel operation described in Eq. 1.

DC-Conv layers were followed by a batch normalization layer [17], a ReLU [20] activation and a dropout layer [14] to avoid overfitting. These configuration was maintained with the exception of the output block. In the output block a 3 DC-Conv layers were implemented to learn the 5%, 50% and 95% prediction intervals. The output layers were followed by a ReLU activation to avoid convergence to negative values during training. As the last TCN component, two skip residual connections [13] were created from blocks 1–3 and 2–4. The model was implemented using Pytorch [21]. Figure 3 shown the detailed architecture.

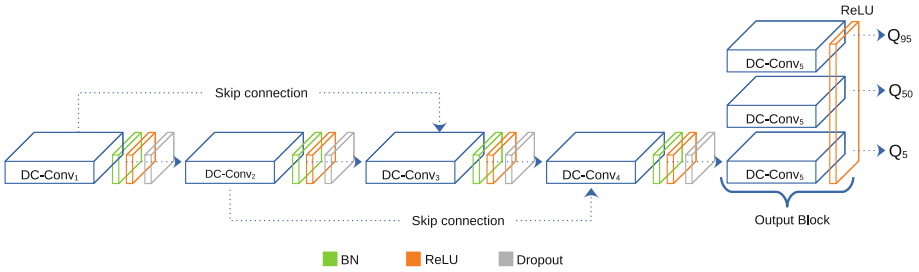

Fig. 3. Temporal Convolutional Neural Network (TCN) architecture. Blue blocks denotes casual dilated convolutions (DC-Conv). Batch normalization (green), relu activation (orange) and dropout (gray) are applied to blocks 1–4. A set of three Dc-Conv layers learns 5%, 50% and 95% prediction intervals. To avoid negative outputs, a relu layer is applied at the end of the output block.

4.2 Bayesian Optimization

In order to obtain the optimum architecture and hyper-parameters, a Bayesian search was performed on the validation data set using the BoTorch [6] library. Minimization of the quantile loss defined in Eq. 3 was used as objective. An initial list of hyper-parameters were build to start the search. Optimization was performed over the validation training data (Table 2) using the Sobol+GPEI strategy present in the Ax [5] library. The hyper-parameters were evaluated against the objective using the test partition from the validation data set (Table 2). Due to the sensitivity in the hyper-parameters, some configurations lead to non-optimal results. In such cases, the last well know configuration was restored. This cycle was repeated until no new improvements in the objective were found.

The search was executed using an AMD Ryzen 7-3700x CPU. Tables 4 and 5 shown the best configuration found for the architecture and hyper-parameters respectively.

Table 4. DC-convs configuration obtained after Bayesian optimization over the validation data set.

Block	Input ch.	Output ch.	Kernel size	Padding	Dilation	Dropout ratio
DC-Conv$_1$	10	32	3×3	8	4	3.00E-01
DC-Conv$_2$	32	32	3×3	16	8	6.58E-01
DC-Conv$_3$	32	32	3×3	16	8	4.66E-01
DC-Conv$_4$	32	32	3×3	20	10	3.62E-01
DC-Conv$_5$	32	1	2×2	8	14	–

Table 5. Optimal hyper-parameter configuration obtained with Bayesian optimization on the validation data set. This configuration was used to built the final model on the training data set.

Hyper-param	Lr	Max Lr	Momentum	Decay	Optim	Epochs	Batch
Value	5.18E-03	6.00E-03	5.27E-01	3.02E-06	RMSProp	7598	60

4.3 Training Configuration

Using the optimum values from Table 5, a new TCN model was trained on the training data set (Table 3). The model output was configured to learn the $Q_5\%$, $Q_{50}\%$ and $Q_{95}\%$ quantiles, which are defined as follows:

$$\mathcal{L}(\xi_i|q) = \begin{cases} q\xi_i, & \xi_i \geq 0 \\ (q-1)\xi_i, & \xi_i < 0 \end{cases} \tag{2}$$

where $q \in [0, 1]$ and ξ_i is defined as the difference between the real values y and predictions $\hat{y}$, such that: $\xi_i = y_i - \hat{y}_i$. Then, the total quantile error is averaged during training over the samples m using the following loss:

$$\mathcal{L}(y, \hat{y}|q) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(\xi_i|q) \tag{3}$$

The loss defined in Eq. 3 was optimized using the resilient backpropagation algorithm (RMSProp) [24]. To avoid any memorization during training, each batch was shuffled. In order to smooth the optimization area, the parameters were initialized using Xavier [11] with a normal distribution. To avoid any sub-optimal optimization, a cycle learning [23] strategy with a step size of 2000 was used during training.

Due to the different magnitude of the target variable y between regions, a normalization step was perform during training using: $log(y+1)$. This operation was reversed with $e(\hat{y}+1)$ to compute the estimates. Finally, in order to apply forecasting beyond the test data, a multi step strategy of 1 day ahead was used.

5 Evaluation Strategy

In order to measure the model performance, an evaluation strategy composed by two levels: global and locally was implemented. In the global level, the training partitions were used to evaluate the model learning capabilities as well as improvement variations between validation and training. On the other hand, the test partitions were used to measure the generalization to new unseen data, which offered a forecasting estimate error. Then at local level, the evaluation was applied per each region.

This strategy allowed us to draw a more complete evaluation of the model performance. Next, a set of metrics were selected to measure the inference $\hat{y}$ error against the real targets y for a sample size m. These metrics are: the mean absolute error (MAE), median absolute error (MAD), mean squared logarithmic error (MSLE) and root mean squared logarithmic (RMSLE). The following equations describe them in more detail:

$$MAE = \frac{1}{m}\sum_{i=1}^{m}|y_i - \hat{y}_i| \tag{4}$$

$$MAD = median(|y_i - \hat{y}_i|) \tag{5}$$

$$MSLE = \frac{1}{m}\sum_{i=1}^{m}[log(\hat{y}_i + 1) - log(y_i + 1)]^2 \tag{6}$$

$$RMSLE = \sqrt{\frac{1}{m}\sum_{i=1}^{m}[log(\hat{y}_i + 1) - log(y_i + 1)]^2} \tag{7}$$

Finally, to measure the prediction intervals robustness we used the percentage of real data points y which are between by the lower (L) and upper (U) intervals, such that: $L \leq y \leq U$. This metric is defined as the Prediction Interval Coverage Probability (PICP):

$$PICP = \frac{1}{m}\sum_{i=1}^{m}u_i, \qquad u_i(y) = \begin{cases} 0 & \text{if } L \geq y \geq U \\ 1 & \text{if } L \leq y \leq U \end{cases} \tag{8}$$

6 Results

This section analyzes the prediction and forecasting results using the evaluation strategy defined in Sect. 5. During the analysis the training and validation data sets were used. Predictions were computed on the training data set (Table 3). Forecasting error estimations were approximated from the test partition (Table 3). Finally, the model was evaluated using the metrics defined in Sect. 5. The following sub-sections will describe these analysis in more detail.

6.1 Prediction Analysis

Predictions were computed using the training and test partitions (Table 3). This process was performed in every coastal region. The training period started on March 06, 2020 and concluded on March 29, 2021, having more than one year of data. Meanwhile, the test period considered 15 days ahead of the training one, starting on March 30 and concluding on April 13, 2021. Analysis of the results from Fig. 4 shown that each coastal region have its particular trend, which are well captured by the model predictions $\hat{y}$. Furthermore, prediction intervals help us to drawn possible variations of daily SARS-CoV-2 cases in every region.

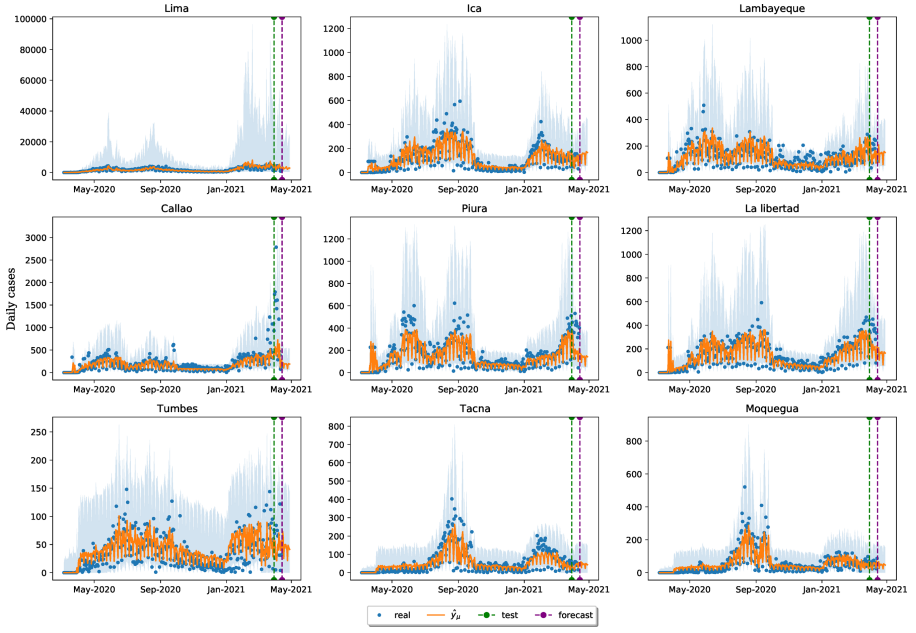

Fig. 4. Daily SARS-CoV-2 predictions from March 06, 2020 to March 29, 2021. The orange line represents the predictions $\hat{y}$, while the light blue are shown the 95% prediction interval. The green line denotes the beginning of the test period (March 30 to April 13, 2021). Meanwhile, the purple line denotes the start of forecasting from April 14 to April 29, 2021. Prediction and forecasting used a 15 day ahead window.

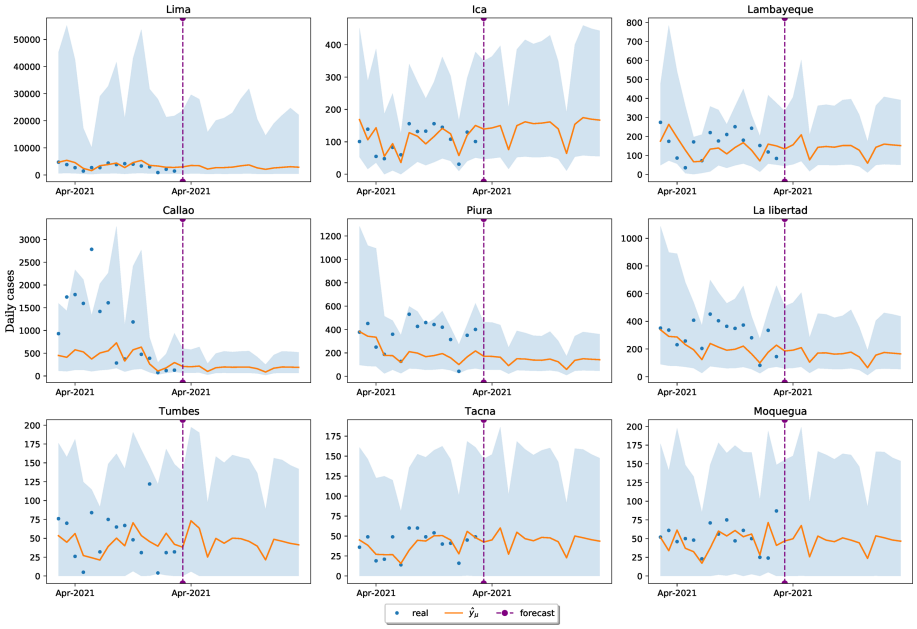

Fig. 5. An enlarged view of the daily SARS-CoV-2 forecasting tendencies in the Peruvian coast. The orange line denotes the prediction $\hat{y}$. Meanwhile the light blue area shown the 95% prediction interval. The start of the forecasting period is denoted with a purple line. Estimations were computed from April 14 to April 29, 2021.

Results on Lima suggest a large daily variations of SARS-CoV-2 cases. This is reflected by the large prediction area describe by the prediction interval. This behavior is less present in the remaining regions. The next sub-section will explore the forecasting behavior in more detail.

6.2 Forecasting Analysis

Forecasting was performed with 15 day ahead of the available test data. Results were computed from April 14 to April 29, 2021 for the 9 coast regions. Forecasting tendencies in Fig. 5 suggest a continuous grown for all regions with different rates. This behavior is accentuated by the peaks observed in the prediction intervals. Also an underestimation of $\hat{y}$ in the test period is observed for: Lambayeque, Callao, Piura and La Libertad observed. However, these variations are well captured by the prediction intervals across the 9 regions.

In order to obtain a quantifiable analysis of the forecasting period (15 days), the maximum and mean tendencies were averaged. Figure 6 shown a possible maximum of 22,937 cases for Lima. This represents the maximum expected variation. The remaining regions shown a more moderate estimate. It is important to note that this variations could change drastically with more recent data.

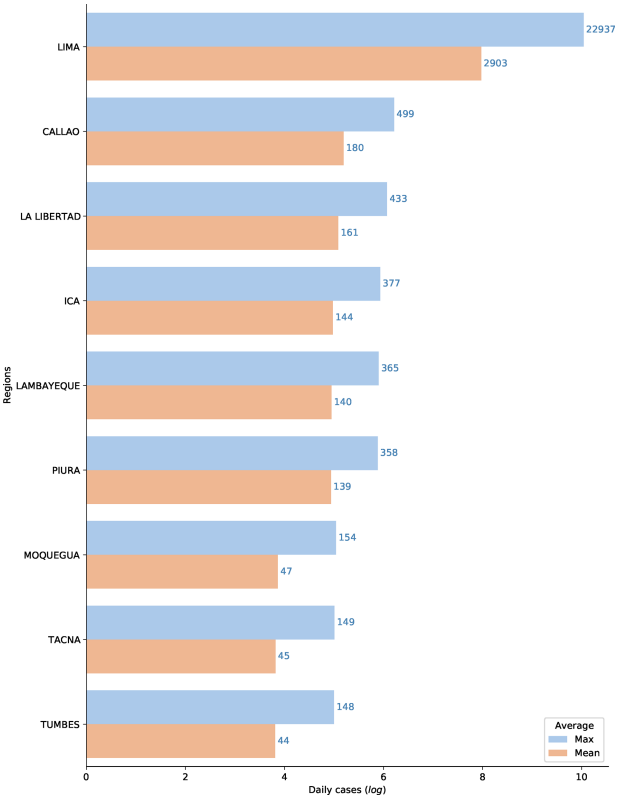

Fig. 6. An average estimated of the mean (orange) and maximum (light blue) $\hat{y}$ cases observed over the 15 day forecast for each coastal region. (Color figure online)

Nevertheless, factors like SARS-CoV-2 variants [22] combined with other components could explain the variations shown in Lima. For example, the Peruvian presidential elections took place on April 11, 2021, just 3 days before the star of the forecasting period on April 14, 2021. In this event, citizens mobilize in Lima, which contains the highest population density of the country. The previous factors could be related with the high forecasting variations observed.

6.3 Evaluation Metrics Analysis

Evaluation results from the validation and training data sets (global) shown in Table 6, indicates a slightly error increase in the training partition. These results suggest that, some data distributions were only present in the training set. On the other hand, the model shown an improvement on the test partition with a consistent decrease for the MAD, MSLE and RMSLE metrics. These results also serves as estimated errors to forecasting. Results from Table 7 shown the estimate error per each region (locally). Error variations are observed across regions,

Table 6. Evaluation metrics applied to the training and validation sets. In bold are shown the improvements from the final TCN model.

Data set	Partition							
	Train				Test			
	MAE	MAD	MSLE	RMLSE	MAE	MAD	MSLE	RMSLE
Validation	85.061	21.189	0.358	0.599	241.295	70.324	1.093	1.045
Training	87.423	25.536	0.514	0.717	245.211	46.082	0.389	0.624
Difference	↑2.362	↑4.347	↑0.156	↑0.119	↑3.260	↓**24.241**	↓**0.704**	↓**0.422**

Table 7. Metric scores per each coastal regions on the test partition (Table 3). The test period spanned from March 30 to April 13, 2021, with a window of 15 days.

Region	MAE	MAD	MSLE	RMSLE
LIMA	1102.096	973.231	0.249	0.499
CALLAO	674.262	480.318	0.931	0.965
ICA	30.554	26.634	0.154	0.393
PIURA	154.217	182.766	0.441	0.664
LAMBAYEQUE	76.616	89.297	0.400	0.633
LA LIBERTAD	112.487	116.535	0.240	0.490
TUMBES	29.418	25.229	0.786	0.887
TACNA	9.816	9.052	0.098	0.314
MOQUEGUA	17.435	13.856	0.202	0.449

Table 8. Prediction interval coverage probability (PICP) results for each coastal regions on the training and test partitions (Table 3). In bold are shown the average PICP estimates for both partitions.

Region	PICP-Train (%)	PICP-Test (%)
LIMA	100.000	100.000
CALLAO	96.915	86.667
ICA	98.715	100.000
PIURA	98.201	93.333
LAMBAYEQUE	96.658	100.000
LA LIBERTAD	97.943	100.000
TUMBES	95.373	100.000
TACNA	98.715	100.000
MOQUEGUA	98.458	100.000
AVERAGE	97.886	**97.778**

suggesting that some tendencies are more complex to learn than others. MAE and MAD metrics shown large variations, whereas MSLE and RMSLE shown 0.931 and 0.965 as their larger errors. PICP results in Table 8 shown an average of 97.886% and 97.778% for training and test respectively. Individually, almost all regions shown values above 90% for training and test with the exception of Callao (86.667% in test). These results shown the model robustness when addressing variations in the SARS-CoV-2 cases among regions.

7 Conclusions

In this research a TCN model was built to predict and forecasting the daily SARS-CoV-2 cases in the Peruvian coast. Bayesian optimization was used to determine the model architecture and best hyper-parameters. Due to the complexity of the problem, the model generated prediction intervals, allowing to capture variations in the daily SARS-CoV-2 cases per region. Maximum average forecast values suggested a very high estimate for Lima (22,937 cases) within 15 days. The presence of a new SARS-CoV-2 variant as well as the past presidential elections may contributed to the observed results in Lima. Evaluation metric analysis on the training partitions point out a possible difference in distribution between validation and training data sets. However, despite the difference, the model reported improvements over the test partition in three metrics: MAD, MSLE and RMSLE with a decrease of 24.241, 0.704 and 0.422 respectively. On the other hand, regional error analysis on the test partition suggested that some SARS-CoV-2 distributions like Lambayeque, Callao, Piura and La Libertad are harder to learn than others. Nonetheless, PICP results shown values greater than 90.0% (train) and 97.778 % (test) for almost all regions, with the exception of Callao (86.667%-test). On average the model prediction intervals captured about 97.886% (training) and 97.778% (test). The presented model can be used as a tool to estimate tendencies in daily SARS-CoV-2 cases, aiding in the decision making of health policies in each region. This study could also by applied to other countries. As such, it is recommended to add more variables to improve performance.

References

1. Abbasimehr, H., Paki, R.: Prediction of COVID-19 confirmed cases combining deep learning methods and Bayesian optimization. Chaos Solitons Fractals **142**, 110511 (2021). https://doi.org/10.1016/j.chaos.2020.110511, https://linkinghub.elsevier.com/retrieve/pii/S0960077920309036
2. Abou-Ismail, A.: Compartmental models of the COVID-19 pandemic for physicians and physician-scientists. SN Compre. Clin. Med. **2**(7), 852–858 (2020). https://doi.org/10.1007/s42399-020-00330-z, https://link.springer.com/10.1007/s42399-020-00330-z
3. Anastassopoulou, C., Russo, L., Tsakris, A., Siettos, C.: Data-based analysis, modelling and forecasting of the COVID-19 outbreak. PLOS ONE **15**(3), e0230405 (2020). https://doi.org/10.1371/journal.pone.0230405, https://dx.plos.org/10.1371/journal.pone.0230405

4. Bai, S., Kolter, J.Z., Koltun, V.: An empirical evaluation of generic convolutional and recurrent networks for sequence modeling (2018). http://arxiv.org/abs/1803.01271
5. Bakshy, E., et al.: AE: a domain-agnostic platform for adaptive experimentation (2018)
6. Balandat, M., et al.: BoTorch: a framework for efficient monte-carlo bayesian optimization (2019). http://arxiv.org/abs/1910.06403
7. Bchetnia, M., Girard, C., Duchaine, C., Laprise, C.: The outbreak of the novel severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2): a review of the current global status. J. Infect. Public Health **13**(11), 1601–1610 (2020). https://doi.org/10.1016/j.jiph.2020.07.011, https://linkinghub.elsevier.com/retrieve/pii/S1876034120305918
8. Chimmula, V.K.R., Zhang, L.: Time series forecasting of COVID-19 transmission in Canada using LSTM networks. Chaos Solitons Fractals **135**, 109864 (2020). https://doi.org/10.1016/j.chaos.2020.109864, https://linkinghub.elsevier.com/retrieve/pii/S0960077920302642
9. Dong, E., Du, H., Gardner, L.: An interactive web-based dashboard to track COVID-19 in real time. Lancet Infect. Dis. **20**(5), 533–534 (2020). https://doi.org/10.1016/S1473-3099(20)30120-1, https://linkinghub.elsevier.com/retrieve/pii/S1473309920301201
10. Giannis, D., Ziogas, I.A., Gianni, P.: Coagulation disorders in coronavirus infected patients: COVID-19, SARS-CoV-1, MERS-CoV and lessons from the past. J. Clin. Virol. **127**, 104362 (2020). https://doi.org/10.1016/j.jcv.2020.104362, https://linkinghub.elsevier.com/retrieve/pii/S1386653220301049
11. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Teh, Y.W., Titterington, M. (eds.) Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. Proceedings of Machine Learning Research, vol. 9, pp. 249–256. PMLR, Chia Laguna Resort (2010). http://proceedings.mlr.press/v9/glorot10a.html
12. Harrison, S.L., Fazio-Eynullayeva, E., Lane, D.A., Underhill, P., Lip, G.Y.H.: Comorbidities associated with mortality in 31,461 adults with COVID-19 in the United States: a federated electronic medical record analysis. PLOS Med. **17**(9), e1003321 (2020). https://doi.org/10.1371/journal.pmed.1003321, https://dx.plos.org/10.1371/journal.pmed.1003321
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778. IEEE (2016). https://doi.org/10.1109/CVPR.2016.90, http://ieeexplore.ieee.org/document/7780459/
14. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R.: Improving neural networks by preventing co-adaptation of feature detectors (2012). http://arxiv.org/abs/1207.0580
15. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997). https://doi.org/10.1162/neco.1997.9.8.1735, https://direct.mit.edu/neco/article/9/8/1735-1780/6109
16. INEI: Perú - Censos Nacionales 2017: XII de Población, VII de Vivienda y III de Comunidades Indígenas (2017). http://webinei.inei.gob.pe/anda_inei/index.php/catalog/674

17. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: Bach, F., Blei, D. (eds.) Proceedings of the 32nd International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 37, pp. 448–456. PMLR, Lille (2015). http://proceedings.mlr.press/v37/ioffe15.html
18. MINSA: Casos positivos por COVID-19 - [Ministerio de Salud - MINSA] (2020). https://www.datosabiertos.gob.pe/dataset/casos-positivos-por-covid-19-ministerio-de-salud-minsa
19. Mohimont, L., Chemchem, A., Alin, F., Krajecki, M., Steffenel, L.A.: Convolutional neural networks and temporal CNNs for COVID-19 forecasting in France. Appl. Intell. (2021). https://doi.org/10.1007/s10489-021-02359-6, https://link.springer.com/10.1007/s10489-021-02359-6
20. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th International Conference on International Conference on Machine Learning, ICML'10, Omnipress, Madison, WI, USA, pp. 807–814 (2010)
21. Paszke, A., et al.: PyTorch: an imperative style, high-performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., d' Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 32, pp. 8024–8035. Curran Associates, Inc. (2019). http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf
22. Reuters: Peru hits new COVID-19 case record as Brazilian variant spreads (2021). https://www.reuters.com/article/us-health-coronavirus-peru-idUSKBN2BG3CL
23. Smith, L.N.: Cyclical learning rates for training neural networks. In: 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 464–472. IEEE (2017). https://doi.org/10.1109/WACV.2017.58, http://ieeexplore.ieee.org/document/7926641/
24. Tieleman, T., Hinton, G.: Lecture 6.5-rmsprop: divide the gradient by a running average of its recent magnitude. COURSERA Neural Netw. Mach. Learn. **4**(2), 26–31 (2012)
25. Wieczorek, M., Siłka, J., Połap, D., Woźniak, M., Damaševičius, R.: Real-time neural network based predictor for cov19 virus spread. PLOS ONE **15**(12), e0243189 (2020). https://doi.org/10.1371/journal.pone.0243189, https://dx.plos.org/10.1371/journal.pone.0243189
26. Zeroual, A., Harrou, F., Dairi, A., Sun, Y.: Deep learning methods for forecasting COVID-19 time-Series data: a comparative study. Chaos Solitons Fractals **140**, 110121 (2020). https://doi.org/10.1016/j.chaos.2020.110121, https://linkinghub.elsevier.com/retrieve/pii/S096007792030518X