Applied Soft Computing 125 (2022) 109181

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

COVID-19 ICU demand forecasting: A two-stage Prophet-LSTM approach

Dalton Borges [a,b,c,*], Mariá C.V. Nascimento [b,c]

[a] Instituto de Ciência e Tecnologia, Universidade Federal Fluminense (UFF), Rio das Ostras, RJ, 28.890-000, Brazil
[b] Divisão de Ciências da Computação, Instituto Tecnológico de Aeronáutica (ITA), São José dos Campos, SP, 12.228-900, Brazil
[c] Instituto de Ciência e Tecnologia, Universidade Federal de São Paulo (UNIFESP), São José dos Campos, SP, 12.247-014, Brazil

ARTICLE INFO

Article history:
Received 14 February 2022
Received in revised form 4 June 2022
Accepted 8 June 2022
Available online 17 June 2022

Keywords:
Neural networks
Forecasting
Time series
Machine learning
Decision support systems

ABSTRACT

Recent literature has revealed a growing interest in methods for anticipating the demand for medical items and personnel at hospital, especially during turbulent scenarios such as the COVID-19 pandemic. In times like those, new variables appear and affect the once known demand behavior. This paper investigates the hypothesis that the combined Prophet-LSTM method results in more accurate forecastings for COVID-19 hospital Intensive Care Units (ICUs) demand than both standalone models, Prophet and LSTM (Long Short-Term Memory Neural Network). We also compare the model to well-established demand forecasting benchmarks. The model is tested to a representative Brazilian municipality that serves as a medical reference to other cities within its region. In addition to traditional time series components, such as trend and seasonality, other variables such as the current number of daily COVID-19 cases, vaccination rates, non-pharmaceutical interventions, social isolation index, and regional hospital beds occupation are also used to explain the variations in COVID-19 hospital ICU demand. Results indicate that the proposed method produced Mean Average Errors (MAE) from 13% to 45% lower than well established statistical and machine learning forecasting models, including the standalone models.

1. Introduction

Uncertainty is a constant in the hospital environment. One of the main issues hospitals face is uncertainty over demand for their services [1]. The number of incoming patients varies substantially over time, impacting the number of beds and items needed. Nevertheless, even socially-driven industries, such as hospitals, are still interested in improving their business performance [2]. On the one hand, over-forecasting the demand may waste resources that could be used elsewhere within the hospital. On the other, under-forecasting may pull the hospital away from its purposes, such as to provide excellent and efficient patient care [3] or care for everyone who come to them [1]. Moreover, hospital performance is closely inspected by government and society: generally its primary stakeholders. Thus, the task of balancing service and costs is notably challenging during turbulent scenarios, such as global healthcare crises, when hospitals may face disruptions in the upstream and over-demand in the downstream [4].

The COVID-19 pandemic is the most significant global healthcare crisis of the last century. Despite HID/AIDS being associated with more deaths (32 mi since 1981),[1] the COVID-19 pandemic is a natural disaster with no precedents to our generation, due to its great transmissibility and severe economic impact [4]. It has been causing recessions, damage to people's health and jobs, and overcrowding hospitals worldwide.[2] Those impacts are amplified in developing countries, where social and hospital infrastructure are generally under stress even during regular times [5]. Unfortunately, SARS-CoV-2 variants are still emerging, such as Omicron, reducing the certainties that we may have regarding an imminent end to this pandemic [6]. Additionally, COVID-19 does not seem to be a standalone episode, and the chances of other pandemics striking humanity soon may to be increasing [7,8]. Thus, operating under strict budgetary limits and high volatility in demand may become more frequent to hospitals in general, even in developed countries [3,4].

Ex-ante forecasts are those made using only the information that is available at the time of the forecast. In contrast, ex-post

[*] Corresponding author at: Instituto de Ciência e Tecnologia, Universidade Federal Fluminense (UFF), Rio das Ostras, RJ, 28.890-000, Brazil.
E-mail address: daltonborges@id.uff.br (D. Borges).

[1] HIV/AIDS vs. COVID: https://time.com/5915401/world-aids-day-covid-coronavirus-pandemic/.

[2] OCDE's Economic Outlook: https://www.oecd.org/newsroom/global-economy-faces-a-tightrope-walk-to-recovery.htm.

https://doi.org/10.1016/j.asoc.2022.109181
1568-4946/© 2022 Elsevier B.V. All rights reserved.

forecasts use later information on the predictors, beyond the time at which the forecast is made [9–11]. Generally, researchers employ regression approaches to tackle ex-post forecasting problems. At the same time, ex-ante forecasts may be performed by classical time series forecasting approaches [9]. However, choosing between classical forecasting or regression approaches may lead researchers to perform simplifications that make the forecast lack precision. Selecting a subset of all relevant variables available, lagging the variables beyond its optimum lag window, or discarding temporal patterns that the target variable may have, are examples of such simplifications. To Ioannidis et al. [12] those may be causes for COVID-19 forecasting models to fail.

This paper presents the applicability of an Integrated Multivariate Prophet-LSTM approach to forecasting ICU beds, considering traditional time series patterns and explanatory and correlated variables with different optimal time lags. The addressed hypothesis is that Prophet-LSTM, a hybrid ex-ante/ex-post forecasting approach, results in better ICU bed forecasts during the COVID-19 pandemic than standalone Prophet and LSTM approaches. This modeling was motivated on the premises that Prophet performs well on capturing time series components and linear relationships between the demand and ex-post variables. On the other hand, LSTM efficiently captures additional non-linearities and correlations between the ICU demand and ex-ante variables. Besides the univariate time series components, several additional variables that partially explain the variations in the target variable are considered: the current number of daily COVID-19 cases, vaccination rates, non-pharmaceutical interventions, social isolation index, and regional hospital beds occupation. Also, this paper presents a comparative analysis between the proposed model and benchmark models, typically used in hospital demand and COVID-19 forecasting, namely Autoregressive Integrated Moving Average (ARIMA), Holt-Winters, Random Forest Regressor (RFR), K-Nearest Neighbors Regressor (KNN), Gated Recurrent Units (GRU) and Simple Recurrent Neural Networks (Simple RNN). To the best of our knowledge, no study has yet investigated a similar approach to the case of hospital demand forecasting. The main contributions of this paper can be summarized as follows:

- It provides a thorough analysis of the correlation between new ICU entrances (target variable) and daily COVID-19 new cases, vaccination rates, non-pharmaceutical interventions, regional hospital beds occupation, and social isolation indexes. Other variables that do not display significant correlation to the target variable are also mentioned.
- It uses the variables mentioned earlier to predict daily ICU entrances in all hospitals of a large Brazilian municipality, aggregated to the city level.
- It loosely couples Prophet and LSTM, two trending Machine Learning (ML) approaches, to incorporate ex-post and ex-ant variables.
- It shows that proposed Prophet-LSTM reduces MAE metrics of Prophet and LSTM in 14.2% and 17.3%, respectively, when feeding all three models with the same inputs.
- It shows that the proposed Prophet-LSTM produces Mean Average Error (MAE) metrics 30%, 27%, 45%, and 34% smaller than univariate Holt-Winters, ARIMA, RFR, and KNN models, respectively.
- The proposed Prophet-LSTM produces Mean Average Error (MAE) metrics 13% and 14% smaller than multivariate GRU and Simple RNN models, respectively.

The remainder of this paper is organized as follows: Section 3 introduces the data and the data treatments, and analyses we perform. Next, Section 4 proposes the two-stage model hybridizing Prophet and LSTM. Then, Section 5 presents the results from Prophet-LSTM and the benchmark models, and the discussions concerning the results. Finally, Section 6 shows the conclusions and future work directions.

2. Related work

Demand forecasting plays a central role in hospital applications. Despite forecasting not being a novel topic to statistics, nor to healthcare sciences and hospitals [13,14], it has been constantly improved and adjusted to solve current problems [15].

During the COVID-19 pandemic, forecasting models that once were kept in vaults as strategic secrets became available to society, as part of a worldwide effort to suppress this crisis through knowledge sharing and science. Recently, many researchers have investigated and created forecasting approaches to predict the numbers related to COVID-19, such as: when the pandemic will peak, how long it will last, how many will be infected or die, and how big will be the demand for hospital beds, ventilators, and PPE (Personal Protective Equipment) [4,15–19].

As soon as new pieces of data, estimates and conditions are made available, researchers create new models that accommodate those variables to better forecast the problems at hand. In the context of COVID-19, many variables were investigated to explain how the hospital demand could evolve in the future, such as the hospital carrying capacity [20,21], the daily number of COVID-19 deaths [18,20,21], new COVID-19 confirmed cases [18,19,22], ICU new hospital entrances [19,22], percentage of ICU occupancy [21], lockdowns and non-pharmaceutical interventions [18,23], patients' age [22], and vaccination rates [23]. Thus, depending on when and how strongly the effect of those variables reflect upon the target variable, researchers choose between different forecasting approaches [9].

Hospital demand and occupancy have become even more critical during the COVID-19 pandemic. A few researchers have recently aimed to tailor forecasting models to address hospital needs. Capistran et al. [18] presented a forecasting model to predict hospital occupancy in Mexican metropolitan areas during the COVID-19 pandemic. The model is based on the Susceptible–Exposed–Infected–Recovered–Dead (SEIRD) model, but also considers lockdowns and lockdown-relaxations as two kinds of interventions that affect the transmission rates and effective population size. A more straightforward modeling approach was used by Rivera and Urdinola [24] to forecast hospital demands in Colombia. The authors used an SEIR model to indirectly estimate the hospital demand from the number of infected people, which may fall into three groups: those that can stay at home, people who need regular hospital beds, and patients who need ICU treatment. Similarly, Massonnaud et al. [25] forecasted hospital needs in France, using an SEIR model, but with short-term goals.

Apart from compartmental models, commonly used to model medium-long term infectious diseases, other statistical models have also gained space in forecasting COVID-19 hospital numbers, especially for short-term predictions. Nguyen et al. [26] used a Vector Error Correlation Model (VECM), a multivariate time series model, that is a restricted Vector Autoregression (VAR) model designed to be used with non-stationary data that are known to be cointegrated. The authors present a case for hospital census in the Charlotte metropolitan area of North Carolina. They introduce infection rates as additional variables as a scientific contribution of the method. According to the authors, the need for a stable long-run relationship and the assumption that case severity does not change in time are limitations of the model. A very similar approach was presented one year earlier by Berta et al. [27], but to the case of Italy, Switzerland and Spain. The authors use publicly available data on the number of COVID-19 cases as supporting variables.

The next section introduces the COVID-19 data used for the forecasting in the present study.

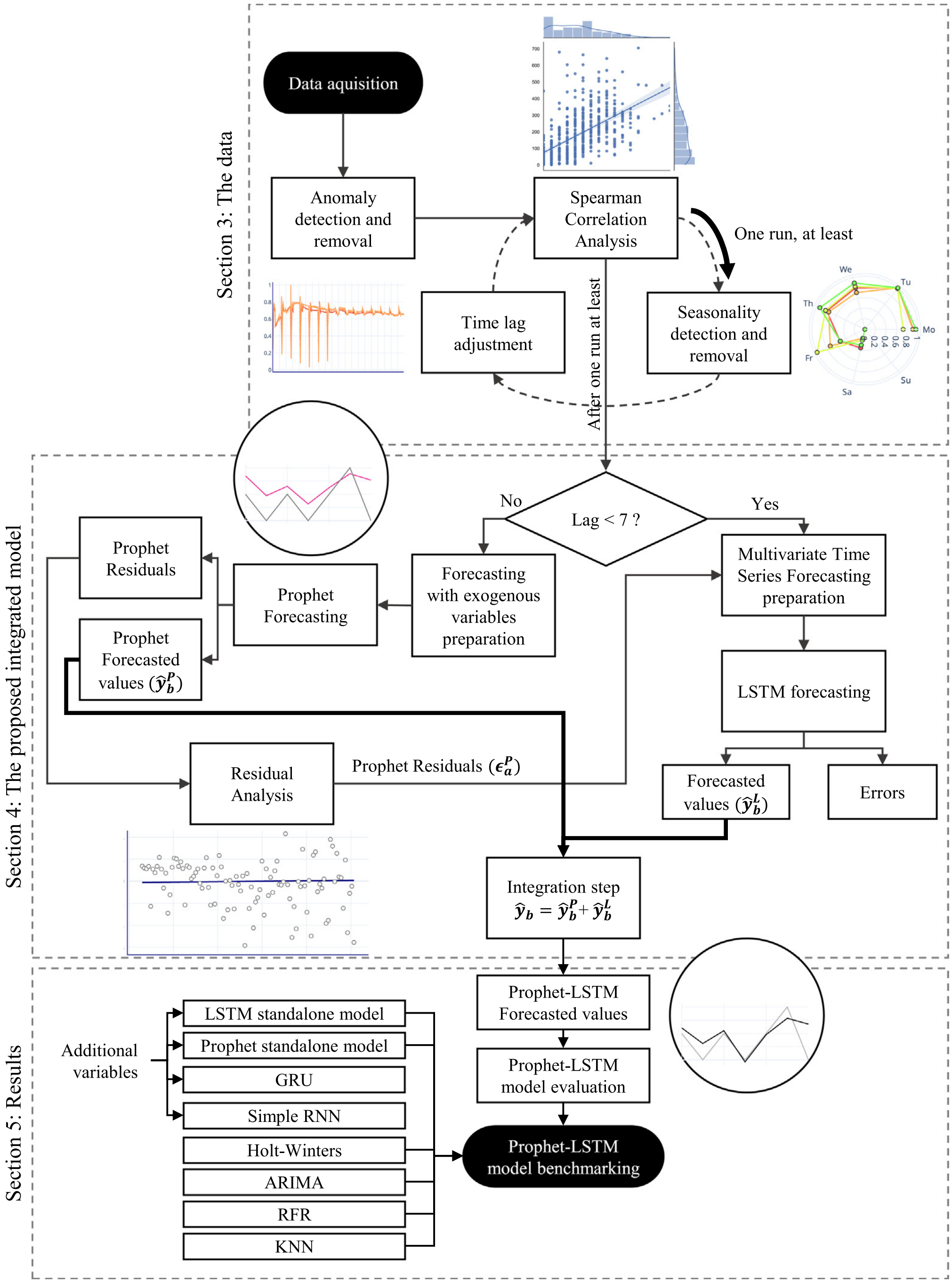
Section 3: The data
Data aquisition
Anomaly detection and removal
Spearman Correlation Analysis
One run, at least
Time lag adjustment
After one run at least
Seasonality detection and removal
We
Tu
Th
Mo
Fr
Su
Sa
Section 4: The proposed integrated model
No
Lag < 7 ?
Yes
Prophet Residuals
Prophet Forecasting
Forecasting with exogenous variables preparation
Multivariate Time Series Forecasting preparation
Prophet Forecasted values ($\hat{y}_b^P$)
LSTM forecasting
Residual Analysis
Prophet Residuals (ϵ_a^P)
Forecasted values ($\hat{y}_b^L$)
Errors
Integration step
$\hat{y}_b = \hat{y}_b^P + \hat{y}_b^L$
Section 5: Results
Additional variables
LSTM standalone model
Prophet standalone model
GRU
Simple RNN
Holt-Winters
ARIMA
RFR
KNN
Prophet-LSTM Forecasted values
Prophet-LSTM model evaluation
Prophet-LSTM model benchmarking

Fig. 1. The methodology of the proposed approach.

3. Case study: The data

This section introduces the investigated data and the pre-processing methodology employed to the data treatment before forecasting. Fig. 1 summarizes the content of this section and the two to follow.

3.1. Data acquisition

Up to July 2021, Brazil was the country with the third-highest number of accumulated cases and the second in the number of COVID-19 accumulated deaths worldwide [28]. Among all Brazilian states, São Paulo had the highest absolute numbers of COVID-19 cases and deaths [29]. Due to the high infection rates, new SARS-COV-2 variants have emerged in the country, such as the more aggressive lineage called P.1. From November 2020, P.1. became dominant in the country [30]. Large cities, such as state capitals, are the most affected ones by the pandemic in Brazil [17]. This is also the case of large inland cities, such as São José dos Campos, in the state of São Paulo.

São José dos Campos is the largest city in the Vale do Paraíba region and a local medical reference. The city has around 731,000 inhabitants and holds the 24th and 19th highest Brazilian HDI (0.807) and annual GDP (US$ 7 billion), respectively [31]. During the COVID-19 pandemic, the city has also served as a scape valve to other parts of the Vale do Paraíba region, which has around 3.3 million inhabitants, by receiving COVID-19 patients from less affluent cities.

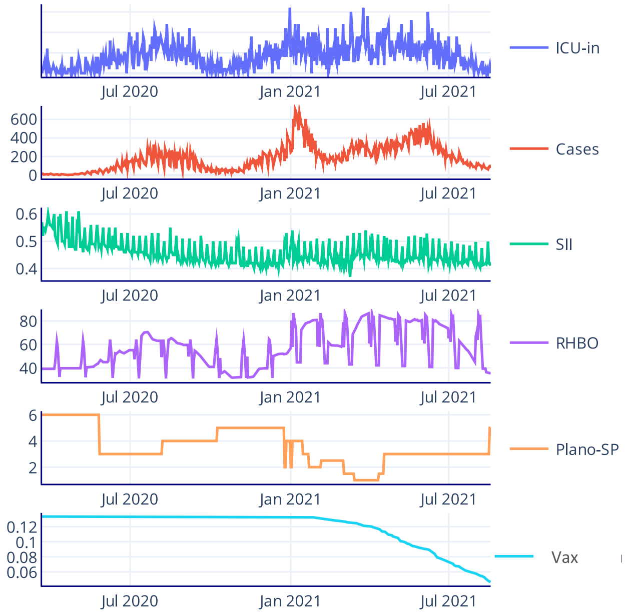

Fig. 2. Target and additional variables used for forecasting.

The primary data source is the municipality of São José dos Campos, which provided to us, under confidentiality agreement, a desegregated dataset to the patient level, reporting all COVID-19 cases in the city and how they have evolved. This data was partially used in a recent study [32]. Thus, we aggregated all the information on a daily level and produced the following first two variables described in each subsection that follows. Besides, several pieces of information were obtained from data freely available online and compiled by the state's COVID-19 contingency center of the state of São Paulo [33]. From this data, we produced the remaining variables of the introduced forecasting model. Fig. 2 graphically shows how each variable, briefly described in the following subsections, evolves in time.

3.1.1. ICU-in

The variable ICU-in accounts for the number of daily entrances in COVID-19 ICUs in all hospitals located in the city. ICU-in is the target variable in this study; all other variables are predictors.

3.1.2. Cases

The variable Cases represents the number of COVID-19 daily cases in the city since the first tracked case on March 05, 2020. Only reported cases are considered, and this variable does not cover sub-notifications.

3.1.3. SII

SII stands for Social Isolation Index and represents the daily percentage of people under social isolation in São José dos Campos. This index is produced by telephonic companies that operate within the state.

3.1.4. RHBO

RHBO stands for Regional Hospital Beds Occupation and refers to the daily percentage of hospital beds in usage in the whole Vale do Paraíba region, including São José dos Campos. For comparison purposes, the population of São José dos Campos represents 22% of the entire population in the Vale do Paraíba region.

3.1.5. Plano-SP

Plano-SP is the state government's color-based reopening plan that grades the regions of São Paulo from red (under severe mobility restrictions) to blue (free of mobility restrictions). This means that cities of a same region have the same mobility restrictions. In particular, Vale do Paraíba is one of the state regions.

Recently, the state government introduced a new phase with more mobility restrictions than the red phase, the "Emergency" phase, as well as two intermediate phases between Red/Emergency and Orange/Red [34]. In this paper, the phases were converted into a linear numeric scale ranging from 0 to 6, with higher values associated with more restricting phases.

3.1.6. Vax

The variable Vax regards an artificial index created in this paper to reflect the percentage of the population that is likely to need an ICU bed. This index is an aggregation of several parameters and variables. It also decreases with the advance of vaccination. It is calculated considering the four types of vaccines available in Brazil, their protection indexes against severe hospitalization after first and second doses. Moreover, since there is a clear relation between people from age groups and the likelihood of them to need an ICU bed due to the COVID-19 infection, we divided the population into m age groups, where $m = 5$.

Roughly, the index Vax consists of an estimation on the number of people requiring an ICU bed from each age group, here denoted by S_i, multiplied by the probability of someone needing an ICU bed, denoted by p_i. Eq. (1) presents the proposed index.

$$\text{Vax} = \frac{\sum_{i=1}^{m} p_i S_i}{S_i'} \tag{1}$$

where S_i' is the size of the population in age group i. The probability p_i is calculated considering the previous available data regarding ICU hospitalizations. To estimate S_i, we consider the size of the population in age group i (S_i') minus the estimated size of the population who do not need an ICU bed due to vaccination, here denoted by S_i''. Therefore, $S_i = S_i' - S_i''$. The value of S_i'' takes into account parameters v_j and w_j, that represent the proportion of people (from any age group) that will not develop severe COVID-19 after one and two shots, respectively. Therefore, S_i'' can be calculated as in Eq. (3).

$$S_i'' = \sum_{j=1}^{n} w_j a_{ij} + v_j(b_{ij} - a_{ij}) \tag{2}$$

where a_{ij}, b_{ij}, and c_{ij} represent the population inside the age segment i that took, respectively, two, one, or none of the doses of vaccine j. Therefore, $S_i' = \sum_{j=1}^{n} a_{ij} + b_{ij} + c_{ij}$, where n is the number of available vaccines, and, thus

$$S_i = a_{ij} + b_{ij} + c_{ij} - \sum_{j=1}^{n} w_j a_{ij} + v_j(b_{ij} - a_{ij}) \tag{3}$$

The index considering every age group is, therefore, calculated as indicated in Eq. (4).

$$\text{Vax} = \sum_{i=1}^{m} p_i \sum_{j=1}^{n} \frac{a_{ij} + b_{ij} + c_{ij} - w_j a_{ij} - v_j(b_{ij} - a_{ij})}{a_{ij} + b_{ij} + c_{ij}} \tag{4}$$

The total number of vaccines available is $n = 4$. The values for m and n reflect the state's policies in the beginning of vaccination. Also, all vaccines have different immunization coefficients, which are provided by the companies that produced them and according to the tested population. These coefficients increase with more vaccine doses taken and vary from vaccine makers, as shown in Table 1. Thus, the first dose has a percentage of immunization v_j, whereas the second dose has a percentage of immunization w_j.

Table 1
Vaccine percentage of immunization by doses taken and vaccine maker.

Vaccine	1st dose	2nd dose
Pfizer	89%	92%
Coronavac	42.7%	86%
AstraZeneca	76%	81%
Janssen	85.4%	–

However, even after immunization, different age segments have different probabilities of needing an ICU bed if infected by COVID-19. Thus a coefficient p_i, obtained from the municipal data prior vaccination, representing the probability of hospitalization for age segment i is also employed in the calculations. Vaccination started on January 18, 2021, thus Vax is constant up to this day, when it starts decreasing.

The main idea behind Equation (4) is to better address the target variable of this paper, the vaccination rates. It is constructed in this manner to reflect the impact of vaccination rates over ICU bed demand, which is our target variable in this paper. Additionally, we only use vaccines as source of immunity for two reasons. First, the percentage of the city's population that contracted COVID-19, by the time this work was performed, was relatively low; and second due to the fact that a single person may be reinfected by different SARS-COV-2 variants, which may implicate in hospitalization.

3.2. Data analysis

We start by empowering the relevance of the predictor variables in describing the behavior of the target variable. To do so, we maximize the correlation coefficient (CC) between the number of ICU new entrances (ICU-in) and other variables that may be relevant to the predictive analysis. To maximize the CC, we remove anomalies from the data, lag the variables according to ranges prescribed in the literature, and detached seasonal effects that may make noise to predictions.

3.2.1. Seasonal analysis

The target variable does not display a continuous and consistent weekly seasonal pattern, unlike what was observed on the variable Cases. Fig. 3 exhibits a graphical analysis of the weekly seasonal components of the variables by using radar charts to illustrate how the seasonality of each variable evolves in time. Each set of lines connecting dots in each graph represents the seasonality pattern in a given period of time. The more consistent the seasonality pattern is, the more parallel the set lines are, and less area of the graph is occupied by the lines. On the other hand, the less consistent the seasonality pattern is, the more frequent lines will cross each other and occupy other areas of the graph.

We notice that as well as the target variable, RHBO, Plano-SP, and Vax also do not seem to display a consistent weekly seasonal pattern along the months. However, the opposite is true for Cases and SII variables, which show a consistent weekly seasonal pattern.

We also perform a linear regression to determine whether seasonal components are relevant to describe the variables in an overall manner. First, each variable is seasonally decomposed by taking the 7 day-moving average of the series as a trend component, and the additive and multiplicative seasonalities are computed. We then regress the original variable on its trend and compute the R-squared metric. Finally, we regress the original variable on its trend adjusted by the seasonal component and verify if there is any considerable improvement in the R-squared. The ordinary least squares (OLS) linear regression was used in both cases. The higher the gain, the more relevant the variable's weekly seasonal component. As shown by Table 2, seasonal components are more relevant in Cases and SII variables.

Table 2
R-squared improvement with additive and multiplicative seasonalities over the decomposed trend.

Variable	Additive	Multiplicative
ICU-in	2.31%	2.61%
Cases	5.23%	6.69%
SII	44.35%	44.16%
RHBO	0.03%	0.02%
Plano-SP	0.07%	0.05%
Vax	0	0

We assume that the target variable does not show a consistent and relevant seasonal component when looking at both analyses. Thus we deseasonalize both the variables Cases and SII before lagging them.

3.2.2. Lagging the variables

Despite all the variables considered in our analysis being collected daily, their influence over ICU new entrances (ICU-in) lags in time. Recent studies point out incubation periods (from infection to symptom onset) for SARS-COV-2 ranging from 4 to 6 days [35–37]. Besides, the interval between symptom onset and hospitalization seems to range from 2 to 10 days [38,39]. Especially in the Brazilian case, testings for COVID-19 are not coincident with symptom onset. In fact, tests happen on average 10 days after the first symptoms [40].

Thus, the correlation analysis performed in this paper also aims to find the best lag to each predictor variable, respecting the ranges found in the literature regarding the target variable. The tested lag ranges we test are shown in Table 3.

3.2.3. Correlation analysis

We use the Spearman correlation, a non-parametric test commonly employed in the context of time series analysis. For this, first consider the Spearman rank correlation coefficient (ρ), in Eq. (5):

$$\rho = 1 - \frac{6}{h_s^3 - h_s} \sum_{i=1}^{h_s} D_i^2 \tag{5}$$

where D_i is the pairwise difference between the ranks of samples, and h_s is the number of samples [17]. The ranks of the samples are obtained from the raw daily data. In the case where, for a given predictor variable, h_s ranks are not distinct integers, a more general correlation measure must be used, presented in Eq. (6).

$$\rho = \frac{cov(rg_X, rg_Y)}{\sigma_{rg_X} \sigma_{rg_Y}} \tag{6}$$

where ρ is the Pearson correlation coefficient between the rankings of the predictor and target variables, $cov(rg_X, rg_Y)$ is the covariance of the rank variables, and σ_{rg_X} and σ_{rg_Y} are the standard deviations of the rank variables [41]. In both formulations, a negative value of ρ indicates inversely proportional variables, and a positive value of ρ indicates directly proportional variables.

Table 3 resumes the optimal time lag for each variable, as well as the correlation coefficient found to the optimal time lag. The p-values for all correlations are less than or equal to 0.01, which means that the correlations are statistically significant.

Table 3 suggests that the most correlated predictor is the number of daily COVID-19 cases in the city, and SII is the less correlated. Since the forecasting approach proposed in the next section is non-linear, even variables with small linear correlation coefficients may bring relevant information to the predictions.

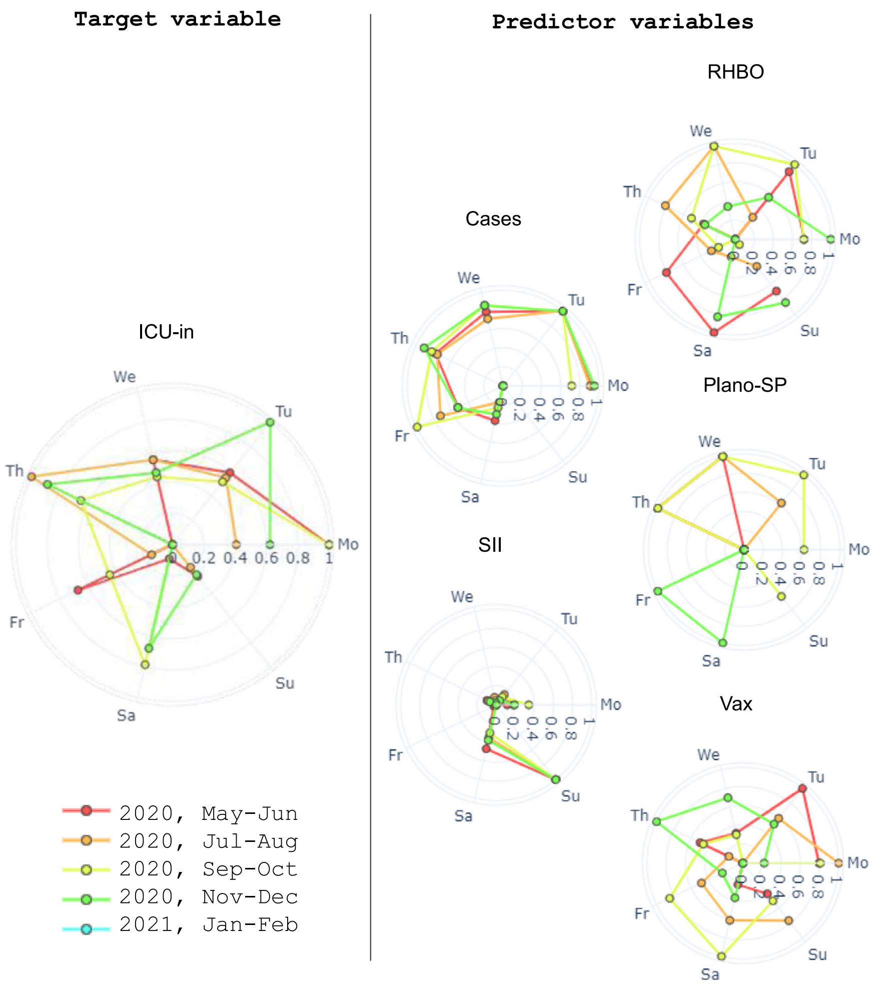

Fig. 3. Seasonal pattern.

Table 3
Optimal time lags and Spearman correlation coefficients (CC) with the target variable.

Variable	Lag range tested	Optimal lag	CC
Cases	0–10 days	0 days	0.69
SII	6–16 days	9 days	−0.07
RHBO	0–10 days	0 days	0.51
Plano-SP	4–16 days	6 days	0.53
Vax	7–21 days	19 days	0.15

Therefore, we deliberately chose to keep all variables, even those with small correlation coefficients, since they may also carry information pertinent to the models.

4. The proposed integrated approach

The approach proposed in this paper integrates the Prophet and LSTM models to tackle the problem of demand forecasting under the influence of independent variables with different optimal time lags.

In the first stage, we run Prophet taking as exogenous variables those with optimum time lag (l), with $l \geq h_b$, where h_b is the forecasting horizon. The introduced model then collects the Prophet fitted values for the training set ($\hat{y}_a^P$), Prophet predicted values ($\hat{y}_b^P$) for the forecasting horizon h_b, and Prophet residuals for the training set (ϵ_a^P), with $\epsilon_a^P = \hat{y}_a^P - y_a$, where y_a refers to the target value for observation a in the training set, and b refers to each observation being predicted, with $a = 1, 2, \ldots, h_a$, $b = 1, 2, \ldots, h_b$, and $h_s = h_a + h_b$.

Then, an LSTM model is applied in the second stage. In this model, the target variables are the set of residuals from Prophet, ϵ_a^P. Besides the residuals, the LSTM model also takes variables those whose $l < h_b$ as input. The LSTM model then returns the predicted values ($\hat{y}_b^L$) for the forecasting horizon h_b. The residuals from Prophet and LSTM are aggregated, producing the Prophet-LSTM predicted values ($\hat{y}_b$), described in Eq. (7):

$$\hat{y}_b = \hat{y}_b^P + \hat{y}_b^L \tag{7}$$

Algorithm 1 presents a pseudocode of the Prophet-LSTM integration.

More details about the Python implementation of the analyses and models are given in Appendix section.

4.1. Fitting and forecasting

Both models, Prophet and LSTM, are explained in the following subsections. In addition, details on how the models were set to the studied data are also provided.

4.2. First stage: Prophet

Prophet is a recent forecasting model introduced by Facebook [42]. The model was originally proposed to forecast daily

Algorithm 1 *Prophet-LSTM*

```
Data: A set of dependent variables $y_a$, a set of independent variables
      $X$, a forecasting horizon $h_b$, and hyperparameters for Prophet and
      LSTM stages
Result: Predictions of Prophet-LSTM, $\hat{y}_b$
Assign to $X^P$ the variables from $X$ with optimum time lag greater than $h_b$
Assign to $X^L$ the variables from $X$ with optimum time lag equal to or lass than $h_b$

Train the Prophet stage of the model, taking $y_a$, $X_a^P$ and set of hyperparameters as inputs, and collect $\hat{y}_a^P$ and the set of Prophet training parameters
Calculate the residuals of the first stage: $\epsilon_a^P = \hat{y}_a{}^P - y_a$
Make predictions with Prophet, taking $y_b$, $X_b^P$ and the set of Prophet training parameters as inputs. Then collect $\hat{y}_b^P$
Train the LSTM stage of the model, taking $\epsilon_a^P$, $X_a^L$ and set of hyperparameters as inputs, and collect the set of LSTM trained parameters
Make predictions with LSTM, taking $h_b$ and a set of LSTM training parameters as inputs. Then collect $\hat{y}_b^L$
Calculate the predictions of Prophet-LSTM: $\hat{y}_b = \hat{y}_b^P + \hat{y}_b^L$
```

demands under the effect of different seasonal patterns, such as weekly and annual seasonalities, plus holiday effects [9]. Despite its novelty, it has been used to tackle demand forecasting problems. For example, Guo et al. [43] used Prophet to forecast the maximum power demand in China. In another Chinese case study, Guo et al. [44] integrated Prophet to Support Vector Regression (SVR) to predict product demands. The proposed integrated model uses SVR to capture nonlinearities in the residuals generated by Prophet. In the case of COVID-19, Prophet has been mainly used to forecast the COVID-19 number of new cases [45,46].

To generate forecasts, Prophet first decomposes the time series in trend (g_t), seasonality (s_t), and holidays (h_t), which are given by Eq. (8) [42,43].

$$y_t = g_t + s_t + h_t + \epsilon_t, \tag{8}$$

where ϵ_t is the residual from the decomposition at time t. In the case of multiplicative seasonalities, g_t and s_t include log transformations.

The trend, g_t, may be described as a piecewise-linear or nonlinear function with saturated logistic growth. In this paper, we use Prophet to model a piecewise-linear function, thus:

$$g_t = (k + a_t^T\delta)t + m + a_t^T\gamma \tag{9}$$

where k is a basic trend growth rate and m is a offset parameter. For each of the S changes in the trend, a changepoint at time s_j is set, with $j = 1, \ldots, S$. Thus, a vector of growth rate adjustments $\delta \in \mathbb{R}^S$ is defined, where δ_j is the change rate at time s_j. The rate at any time t will be the base rate k plus all changes up to t: $k + \sum_{j:t>s_j} \delta_j$. In a cleaner way, a vector $a_t \in \{0, 1\}^S$ can be defined, such that:

$$a_{j,t} = \begin{cases} 1, \text{ if } t \geq s_j \\ 0, \text{ otherwise} \end{cases} \tag{10}$$

Thus the rate at time t is $k + a_t^T\delta$. To each change in k, m must also be updated, in order to connect the segments endpoints. To adjust m, a vector γ is also employed, whose elements $\gamma_j = -s_j\delta_j$.

The changepoints s_j are automatically selected by putting a sparse prior on δ. Thus, to choose the changepoints from several candidates, Prophet uses the prior $\delta_j \sim \text{Laplace}(0, \tau)$, where the parameter τ sets the model's flexibility in altering the rate.

Additional regressors, such as Cases and Vax, are treated as supporting terms in Eq. (9), similarly to tabular supervised settings. Thus, extra regressors need to be known in all future dates for which the target variables will be predicted.

Smoothed weekly seasonality is computed as a Fourier series, given by:

$$s_t = \sum_{e=1}^{E} \left(a_e cos\left(\frac{2\pi e t}{P}\right) + b_e sin\left(\frac{2\pi e t}{P}\right) \right) \tag{11}$$

where $P = 7$ is the regular period we expect in the case of weekly seasonalities associated with the COVID-19 time series presented, and E is a smoothing parameter that regulates how quickly seasonal patterns are fit. Thus, increasing E also increases the risk of overfitting. Fitting seasonality also requires estimating the $2E$ parameters $\beta = [a_1, b_1, \ldots, a_E, b_E]^T$. All Prophet constants are manually set or automatically found via grid search or optimization approaches.

4.2.1. Second stage: LSTM

Neural Networks are the backbones of many Artificial Intelligence (AI) algorithms, which play an important role on daily economic activities [47,48]. Long–Short Term Memory Neural Networks (LSTM) are the most popular deep learning subset of models for time series forecasting [49]. LSTMs are extensions of Recurrent Neural Networks (RNN). The main difference between LSTM and RNN is that an LSTM can store time-dependent information, mapping non-linear relationships between inputs and outputs [50]. Recently, especially after the development of open-source libraries and the improvements in computational power, LSTMs have mainly been used to tackle demand forecasting problems, such as ultra-short-term industrial power demand [51], highly fluctuating product demand [50], and hourly natural gas demand [52]. Similarly to Prophet, during the COVID-19 pandemic, LSTM has been mainly used to forecast the number of COVID-19 new cases [53,54].

LSTM cells and gates form LSTM units, that are merged to compose the LSTM layers of an LSTM neural network. The LSTM units are formed by three main types of gates, that regulate the information flow, and a memory cell. This structure allows an LSTM to decide which information will be forgotten and which will be remembered, promoting the learning process of long-term dependencies [55]. The main gates of an LSTM are the input, output, and forget gates. Associated with these gates, there are the input activation vector i_t, output activation vector o_t and forget activation vector f_t. The first one, along with a second gate c_t^*, determines which information is stored in the memory state vector c_t at time t. The forget gate determines which information must be kept or erased from the memory cell in period $t - 1$. Finally, the output gate decides which information will be used as an output of the memory cell. The architecture of the LSTM model adopted in this paper is explained by Eqs. (12)–(17).

$$f_t = \sigma_{LR}(W_f x_t + U_f h_{t-1} + b_f) \tag{12}$$

$$i_t = \sigma_{LR}(W_i x_t + U_i h_{t-1} + b_i) \tag{13}$$

$$o_t = \sigma_L(W_0 x_t + U_0 h_{t-1} + b_0) \tag{14}$$

$$c_t^* = W_c x_t + U_c h_{t-1} + b_c \tag{15}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \sigma_h c_t^* \tag{16}$$

$$h_t = o_t \odot \sigma_h(c_t) \tag{17}$$

where x_t is an input vector to the LSTM unit, h_t is the output vector of the LSTM unit, σ_{LR} is the LeakyReLU activation function,

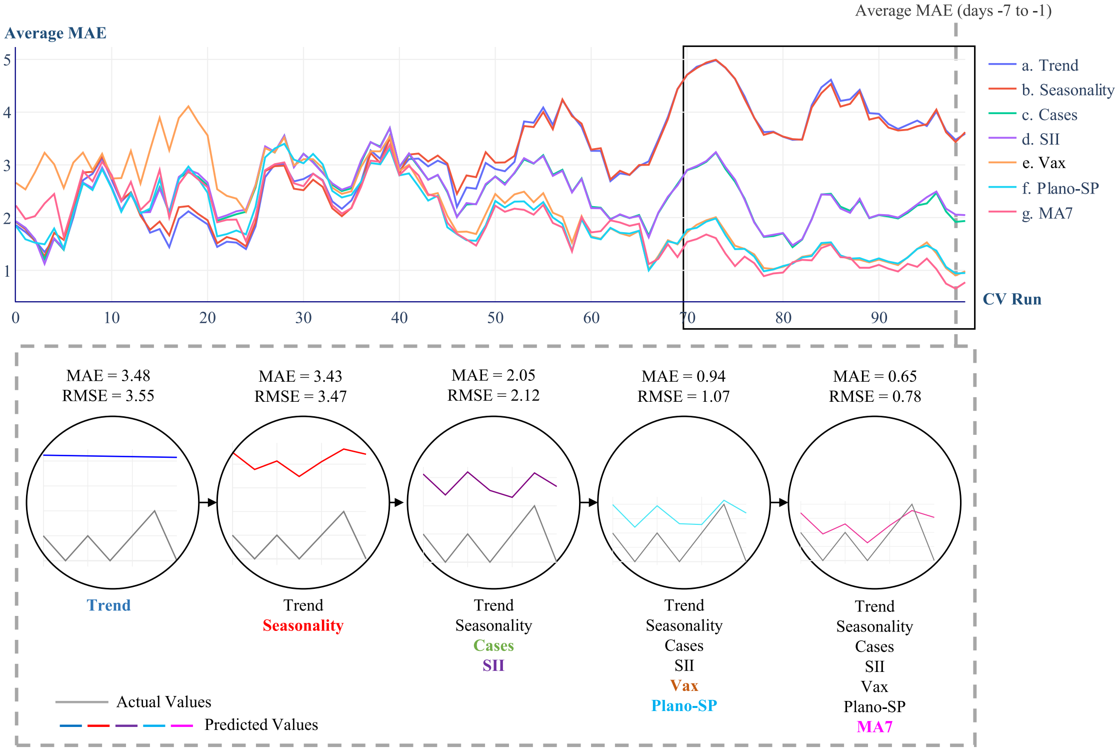

Fig. 4. MAE for feeding forward cross-validation for different Prophet approaches.

σ_L is the linear activation function, σ_h is a hyperbolic tangent activation function, W and U are weight matrices to be learned, whose indexes i, f, o and c refer to the gate or cell the matrices are associated within each step. Finally, $b = [b_f, b_i, b_0, b_c]$ is a bias vector parameter to be learned.

Appendix presents details on the implementation of the method.

5. Results and discussion

This section presents how the experiments were configured and the results obtained. The process starts with the configuration of the first stage, Prophet, and the analysis of its residuals and prediction errors. Then, the second stage, LSTM, is set and the combined Prophet-LSTM metrics are presented. Finally, the performances of Prophet-LSTM and both standalone models are compared with classic univariate models.

5.1. First stage: Prophet output

First, the experiment we carried out using Prophet has as inputs the variables whose optimum correlation with the target variable presented a time lag of 7 days or more, which is the size of the forecasting horizon. Thus, the variables SII and Cases were lagged 7 days ahead, and the variables Vax and Plano-SP were lagged 21 days ahead. All variables were standardized before utilization.

In the upper part of Fig. 4, one may observe how the average MAE evolves in time after each feeding forward cross-validation run, where successive training sets are supersets of those that come before them. Thus, in run 99, the model attempts to predict the last 7 observations available (from 0, the last dataset observation, to −7 days) and with a given MAE. In run 98, the model tries to predict 7 observations of the target variable but between days −8 to −1. This way, in CV run 0, we compute the MAE for the prediction of days −106 to −99, using as the training set all datapoints available before day-106. One may also observe the impact of incrementally adding characteristics, different exogenous variables, and data pre-processing strategies to the Prophet modeling. Thus, the results referred to as “a. Trend” represents a model that only accounts for the trend in the data. The results indicated by “b. Seasonality” display the MAE values for the Prophet model accounting for trend and seasonality. The same is true for all other lines. In particular, the results indicated by “g. MA7” refers to the MAE for the Prophet model that accounts for Trend and Seasonality, and variables Cases, SII, Vax, and Plano-SP. Moreover, this prediction also considers a 7-day Moving Average (MA7) on all exogenous variables series. It is worth mentioning that in the results enumerated from c. to f., the exogenous variables are only standardized and still bring information about their own seasonalities. As observed, taking the moving average of the variables (“g. MA7”), and adding the variables Cases (“c. Cases”) and Vax (“e. Vax”) are the improvements that have greater impacts on Prophet forecasting accuracy. Also, the improvements brought by the components stabilize after CV run 69.

The bottom of Fig. 4 shows the predicted values evolving with the addition of new components and variables to the Prophet modeling. Those relations are extracted by looking inside CV 98, the run with lower MAE. However, the actual values are always the same. Despite the decrease in MAE and RMSE metrics, it is visually possible to notice how the predictions improve after each increment.

As observed in Fig. 5, the residuals look stationary (ADF-statistic is −6.377, with a p-value of 0.0, for a 5.0% significance level), which means they have constant mean, constant variance (homoscedastic), and are free of seasonality. This is graphically represented in Fig. 5 - Residual Plot, which displays a non-repetitive pattern in the observed residuals and mean equal to zero. The residuals also seem to be free of autocorrelation of the first order (DW-statistic of 1.8). Finally, according to the Shapiro–Wilk test for a 5.0% significance level, despite the bell

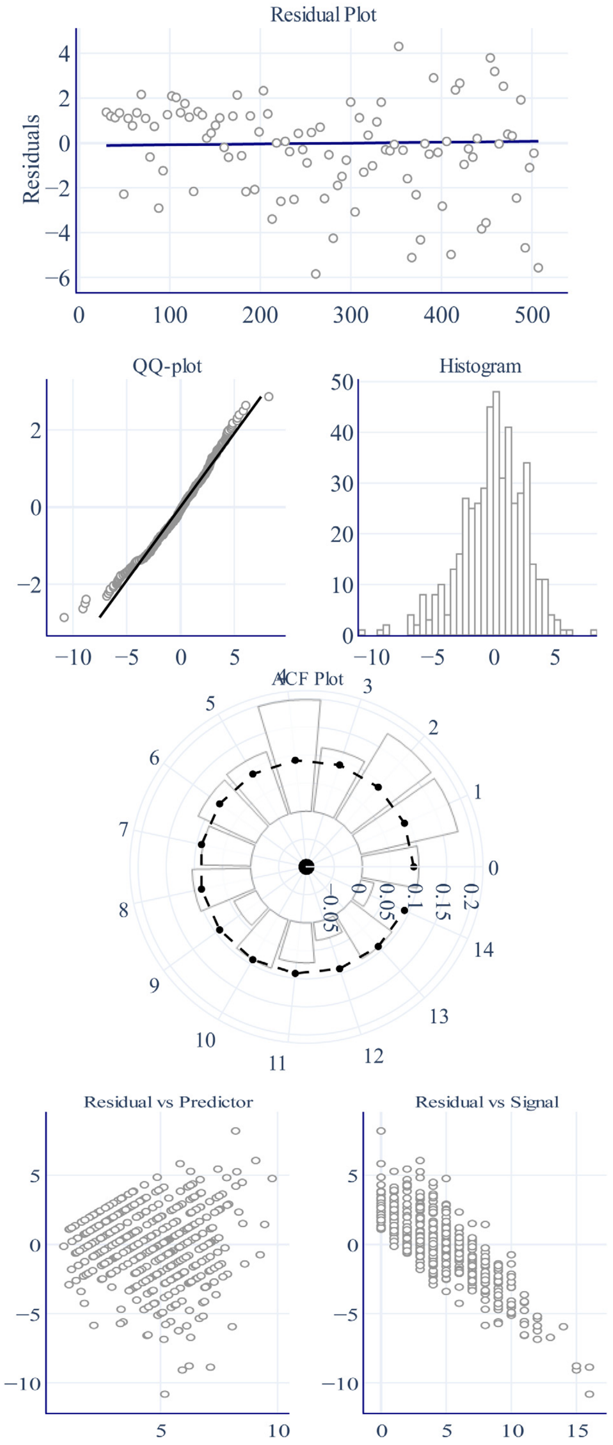

Fig. 5. Residual analysis after first stage with Prophet.

shape in Fig. 5 - Histogram, the residuals are not normally distributed, displaying a W-statistic of 0.975 but a p-value close to zero. Therefore, the residuals may be treated as white noise in a general classification. The residuals also do not display a significant correlation with the predicted values (See Fig. 5 - Residual vs. Predictor).

However, despite the improvements and quality of the residuals, there is still room for improvements in the model. For example, a Pearson's correlation of -0.82 is observed between the residuals and the target variable (See Fig. 5 - Residual vs. Signal). Also, the autocorrelation of orders higher than one was found, as observed in Fig. 5 - ACF Plot. Those realizations may indicate some information left over that should be accounted for, such as those that may be brought by additional independent variables or hidden seasonal components. Thus, to tackle this issue we run the second stage of the proposed method, which relies on a LSTM neural network.

5.2. Second stage: LSTM output

The LSTM multivariate model takes the residuals from the Prophet training set as the target variable. The LSTM model explains the variations in the residuals in terms of the residuals themselves and the variables Cases and RHBO. The variables Cases and RHBO are in their raw form, with no lags applied to them nor variance smoothing with moving averages.

After manually searching, the LSTM architecture was configured as follows: batch size of 5, input layer with 50 neurons, two hidden layers with 50 and 25 neurons each, LeakyReLU as activation function ($\alpha = 0.5$), and a dropout rate of 0.1. We set Mean Squared Error (MSE) as the loss function, Adam as optimizer, and RMSE as the evaluation metric. The models were also set to train for 200 epochs but with early stopping.

Fig. 6 compares the performances of the proposed Prophet-LSTM integration and standalone models Prophet (MA7) and a pure LSTM model running all variables with no lags and following the same architecture as the one used for the integrated approach. Prophet-LSTM outperformed standalone Prophet and LSTM in 87% of the runs. One can also notice that Prophet and LSTM change positions during the runs in a non-consistent pattern. The average MAE for all runs of Prophet-LSTM is 0.99, whereas the MAE for Prophet and LSTM was 1.15 and 1.20, respectively. Thus, the Prophet-LSTM integration to the case of hospital ICU demand forecasting during COVID-19 in a Brazilian municipality reduces standalone Prophet and LSTM MAEs by 14.2% and 17.3%, respectively. This posterior treatment of Prophet residuals with LSTM has a greater impact over the proposed model MAE than the pre-treatment performed in the data, which were performed to lag the variables and remove unwanted seasonal patterns.

Regarding the computational time taken to perform the models, each CV run of standalone Prophet takes 1.27 s in average (with standard deviation (s) of 0.16 s and Coefficient of Variation (CV) of 0.12). As expected, the models running neural nets are considerably more time consuming than Prophet. Standalone LSTM performs each CV run in 11.82 s (with $s = 5.49$ s and $CV = 0.46$), while the Prophet-LSTM model performs each CV run in 15.99 s (with $s = 6.53$ s and $CV = 0.40$). Since Prophet-LSTM sequentially performs both standalone Prophet and LSTM approaches, its total running time is a rough aggregation of both models individual times. On the other hand, Prophet-LSTM CV roughly weights the CVs of both Prophet and LSTM individual models, thus a CV falling in between the CVs of both individual models was expected. Despite the longer time taken by Prophet-LSTM to train and predict the data, especially in comparison to standalone Prophet (almost ten times more time consuming), the total Prophet-LSTM computational running time still seems to have almost no impact over the hospitals demand forecasting routines. It still runs in a scale of seconds.

It is also noticeable how the performance of Prophet-LSTM improves over time, as soon as more training samples are available. However, despite this apparent continuous extension of the model's knowledge, Prophet-LSTM do not seem appropriate

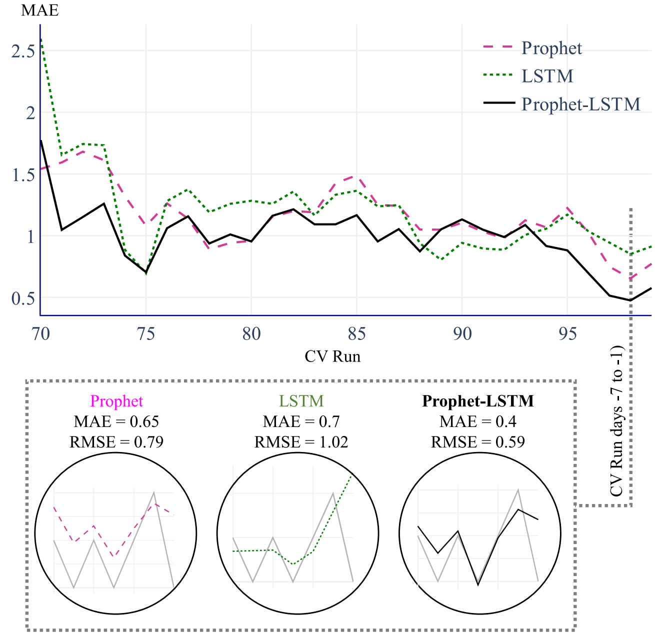

Fig. 6. Target and additional variables used for forecasting.

Table 4
Comparison of forecasting models.

Model	Average MAE	MAE CV
Prophet-LSTM	0.99	0.25
Prophet	1.16	0.22
LSTM	1.20	0.31
GRU	1.13	0.40
Simple RNN	1.16	0.43
Holt-Winters	1.43	0.34
ARIMA	1.37	0.31
KNN	1.51	0.63
RFR	1.82	0.52

to incremental learning tasks, mainly because it is considerably dependent on hyperparameter tuning and feature engineering. Additionally, Prophet-LSTM may also face difficulties in handling intermittent demand or in forecasting time series whose behavior is not partially explained by a set of independent variables.

5.3. Benchmarking

In addition to Prophet and LSTM standalone models, we also compare the results from Prophet-LSTM to other models commonly employed in demand forecasting and to predict COVID-19 numbers. The four univariate forecasting models are Holt-Winters, ARIMA, KNN, and RFR, and the two multivariate models are the GRU and Simple RNN. A good explanation of those models and their main mathematical formulations are presented by Souza et al. [56], Kadri and Abdennbi [57], and Sezer et al. [49]. Table 4 summarizes the average MAE for all models and their respective MAE coefficient of variance, considering the results of all 30 cross-validation runs.

In general, Prophet-LSTM outperforms all models, presenting smaller MAE, RMSE, and sMAPE metrics, also displaying MAE with a smaller coefficient of variation (CV). These results suggest that the model is both more accurate and more stable.

Fig. 7 graphically summarizes how the MAE of the univariate models and Prophet-LSTM varies on time.

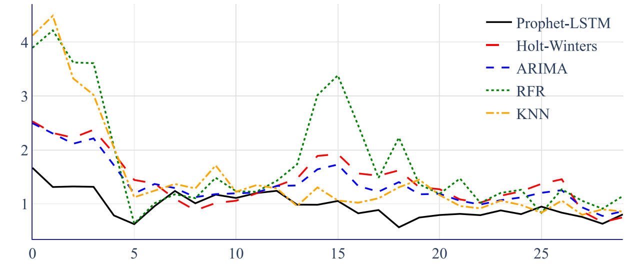

Fig. 7. MAE over time for benchmark models.

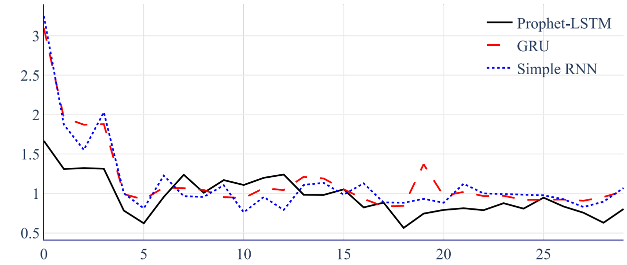

Fig. 8. MAE over time for benchmark models.

Fig. 8 graphically summarizes how the MAE of the multivariate models and Prophet-LSTM varies with time. Notice that despite the better performance of the multivariate models, especially in contrast to the performance of univariate models, the two multivariate benchmarks are considerably less stable than Prophet-LSTM, displaying a much higher MAE CV, as shown by Table 4.

Finally, we also run Mann–Whitney U Tests to all cross-validation runs, comparing Prophet-LSTM to the other models. The null hypothesis (H_0) assumes no significant difference between the two series. The alternative hypothesis (H_1) investigates if the series are different. The H_1 is accepted in all cases for p-values < 0.05. In detail, the p-value of Prophet is 0.008; LSTM is 0.012; GRU is 0.018; Simple RNN is 0.045; Holt-Winters, ARIMA, and RFR are 0.000; KNN is 0.002.

6. Conclusions

In this paper, we showed that a two-stage integrated approach of Prophet and LSTM models results in significantly (max p-value $= 0.012$) better COVID-19 ICU demand forecastings than both standalone models, Prophet and LSTM, and also four univariate benchmarks (ARIMA, Holt-Winters, RFR, and KNN) and two multivariate benchmarks (GRU and Simple RNN). During the Prophet stage, the variables Cases and Vax showed the most significant impact on the model performance. After the second stage, the MAE CV was also improved in relation to standalone LSTM. Despite the smaller forecasting errors, the Prophet-LSTM model is also more time-consuming than both standalone models. Thus, running this model for several time series may demand simplifications or the usage of hierarchical structures. In contrast to Prophet, Prophet-LSTM also significantly increases the time needed to set the optimal parameters before the final forecast. However, in the case of COVID-19 ICU forecasting, the model appears to be an interesting option that could potentially be replicated to other municipalities with similar data available.

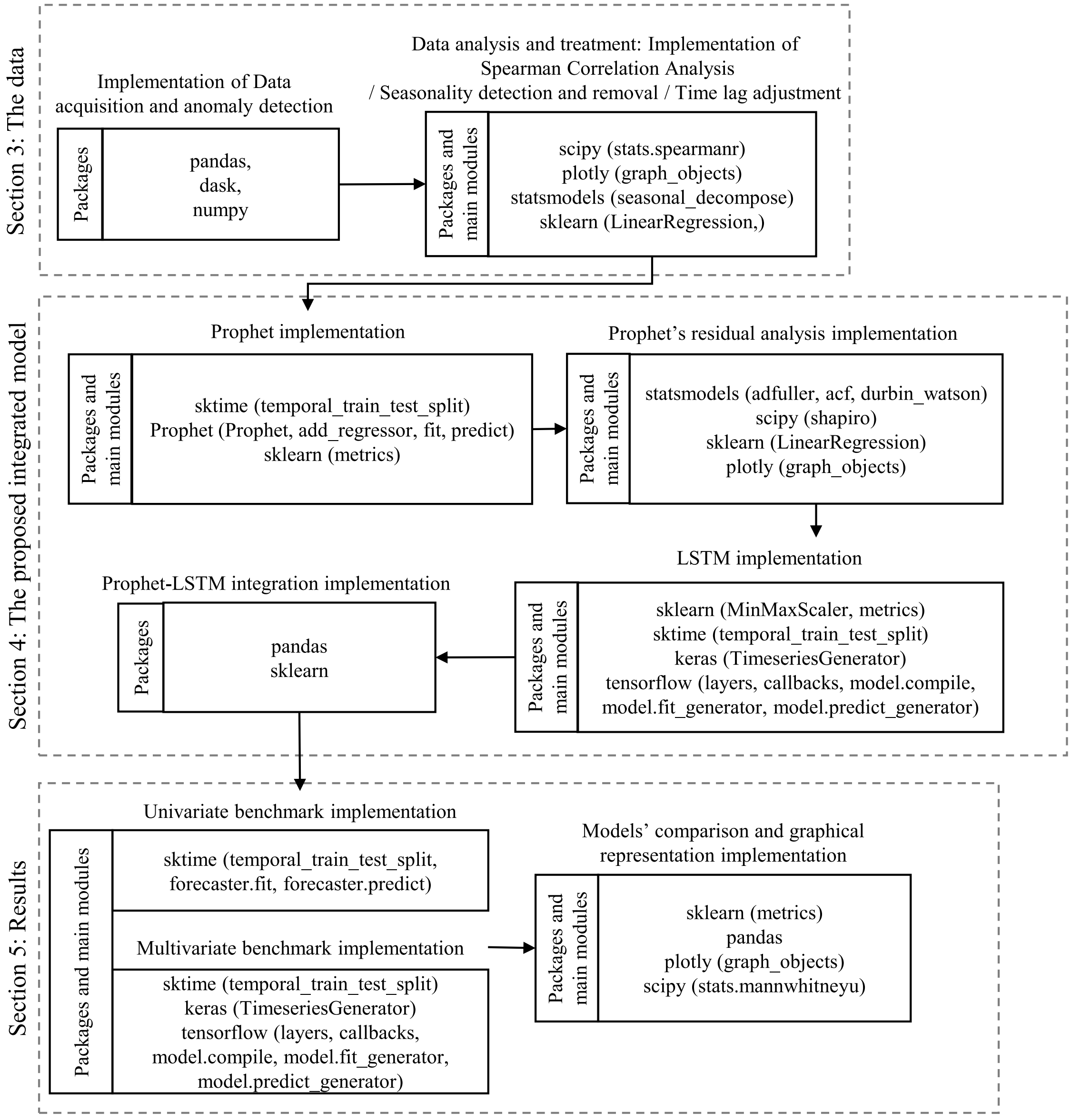
Section 3: The data
Implementation of Data acquisition and anomaly detection
Packages
pandas, dask, numpy
Data analysis and treatment: Implementation of Spearman Correlation Analysis / Seasonality detection and removal / Time lag adjustment
Packages and main modules
scipy (stats.spearmanr)
plotly (graph_objects)
statsmodels (seasonal_decompose)
sklearn (LinearRegression,)
Section 4: The proposed integrated model
Prophet implementation
Packages and main modules
sktime (temporal_train_test_split)
Prophet (Prophet, add_regressor, fit, predict)
sklearn (metrics)
Prophet's residual analysis implementation
Packages and main modules
statsmodels (adfuller, acf, durbin_watson)
scipy (shapiro)
sklearn (LinearRegression)
plotly (graph_objects)
LSTM implementation
Packages and main modules
sklearn (MinMaxScaler, metrics)
sktime (temporal_train_test_split)
keras (TimeseriesGenerator)
tensorflow (layers, callbacks, model.compile, model.fit_generator, model.predict_generator)
Prophet-LSTM integration implementation
Packages
pandas
sklearn
Section 5: Results
Univariate benchmark implementation
Packages and main modules
sktime (temporal_train_test_split, forecaster.fit, forecaster.predict)
Multivariate benchmark implementation
sktime (temporal_train_test_split)
keras (TimeseriesGenerator)
tensorflow (layers, callbacks, model.compile, model.fit_generator, model.predict_generator)
Models' comparison and graphical representation implementation
Packages and main modules
sklearn (metrics)
pandas
plotly (graph_objects)
scipy (stats.mannwhitneyu)

Fig. A.9. Main libraries and modules used to implement the analyses and forecasting models in Python.

CRediT authorship contribution statement

Dalton Borges: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Writing – original draft, Writing – review & editing, Project administration. **Mariá C.V. Nascimento:** Validation, Resources, Writing – review & editing, Supervision, Funding acquisition.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Dalton Garcia Borges de Souza reports financial support was provided by Coordination for the Improvement of Higher Education Personnel (CAPES).

Acknowledgments

The authors would like to thank the Brazilian Coordination for the Improvement of Higher Education Personnel (CAPES), Brazil for fully supporting this work through the Call CAPES-Print 2020 and Grant: 88887.507037/2020-00.

Appendix. Implementation

All data acquisition, analysis, and treatment, as well as the models, are implemented in Python. Fig. A.9 shows the main libraries, modules and the flow in which the analysis and experiments were performed. The framework of Fig. A.9 follows a similar distribution of the processes presented in Fig. 1.

In the first implementation block, regarding data acquisition and anomaly detection, the framework acquires the data (approximately 40 GB) and preprocesses it to patient-level variables in a tabular format. The library Dask is widely used in this step, taking raw data from hospitals and the government as input. Then, data preprocessing occurs by removing simple anomalies, such as patients with dissonant ages and incorrect dates. Finally, the variables for the models are created and consolidated in a daily granularity. This step is performed primarily through the use of the libraries pandas and NumPy, taking patient-level variables as input.

The next stage of the framework consists of data analysis and treatment. It searches for seasonal components in the data and removes them when needed. The libraries used in these steps are

StatsModels, mainly the module seasonal_decompose, and Plotly, for data visualization. The primary input data for the functions of both libraries are the dependent and independent variables. Finally, the modules Linear Regression, of Sklearn library, and Spearmanr, of Scipy library are used to check the optimal correlation between the dependent and independent variables, which are also their primary data input. For the Spearman test, α, the threshold value used to judge whether a test statistic is statistically significant, was set to 0.05.

For the Prophet implementation block, the Sktime library is employed to split the data into training and testing sets, taking the set of target and exogenous variables as input. Then, the Prophet function initiates the model. Its inputs are the growth model (linear in our case), the width of the confidence interval (95% in the case), and Boolean variables indicating the set of seasonalities used. After that, function add.regressor add the exogenous variables to the model and the function fit trains the model, taking as inputs the instantiated model and the training set of the target variable. Finally, the function predict is used to predict future values with the fitted model, the test set, and the size of the forecasting horizon. The framework collects the RMSE, MAE, and SMAPE using the Scikit-learn library.

After Prophet, the framework performs the following tests for the residual analysis:

- Augmented Dickey–Fuller Test, Autocorrelation Test, and Durbin Watson Statistic (functions adfuller, acfn and durbin_watson of Statsmodels library), taking the series of residuals, an α of 0.05 and, in the case of Autorrelation Test, a given number of lags to be mapped (15, in the case) as input.
- The Shapiro–Wilk test (function shapiro of Scipy library), which takes as input the series of residuals and α of 0.05. The residual graphs are then drawn with the libraries Plotly (graph_objects module) and Scikit-learn (LinearRegression module), taking are primary inputs the target variable series of values and the series of residuals.

The LSTM implementation block relies primarily on Keras and TensorFlow libraries. However, for the data preparation, the framework employs the library Sktime to split the data into training and testing sets, taking the set of target and explanatory variables as input, and the library Sklearn (MinMaxScaler function) to normalize the explanatory variables between 0 and 1. The module TimeseriesGenerator of Keras is also employed to prepare the data, taking the window length (5 in the case), sampling_rate (1 in the case), batch_size (5 in the case), and a set of explanatory and target variables as input. The model is then created, and a set of LSTM, LeakyRelu, Dropout, and Dense layers are added through keras.layers modules of Library Keras. The model is then compiled (function compile of Tensorflow, taking the mean squared error as loss function, Adam as optimizer, and RMSE as metric) and fit to the data (function fit_generator of Tensowflow, taking as inputs 100 epochs with early_stopping and the data prepared previously). Finally, predictions are made with module predict_generator.

After running the LSTM model, the Prophet-LSTM model is simply built with Pandas and Numpy operations. Finally, the full integration is analyzed with metrics drawn from Scikit-learn libraries, through the module metrics.

The benchmark implementation block re-implements the Prophet and LSTM models, in the same way they were implemented before; however, taking the whole set of variables is available as input. The Simple RNN and GRU models follow similar implementations as the LSTM model, but with GRU and RNN gates, instead of LSTM gates. Finally, the Univariate models are implemented with Sktime library. The data is first split into training and testing sets (function temporal_train_test_split), the training set is used to train the modules (forecaster.fit), and the trained model and testing sets are used to make the predictions (forecaster.fit).

Once the predicted values are collected from Prophet-LSTM, Prophet and LSTM standalone models, and all benchmark models, they are compared using the module metrics of Scikit-Learn and Mann–Whitney U Test (function stats.mannwhitneyu of Scipy library). Finally, we draw the comparison graphs with Library Plotly (graph_objects module).

References

[1] M. Gaynor, G.F. Anderson, Uncertain demand, the structure of hospital costs, and the cost of empty hospital beds, J. Health Econ. 14 (3) (1995) 291–317.

[2] P. Raju, S.C. Lonial, Y.P. Gupta, Market orientation and performance in the hospital industry, Mark. Health Serv. 15 (4) (1995) 34.

[3] G. Abraham, G.B. Byrnes, C.A. Bain, Short-term forecasting of emergency inpatient flow, IEEE Trans. Inf. Technol. Biomed. 13 (3) (2009) 380–388.

[4] K. Nikolopoulos, S. Punia, A. Schäfers, C. Tsinopoulos, C. Vasilakis, Forecasting and planning during a pandemic: COVID-19 growth rates, supply chain disruptions, and governmental decisions, European J. Oper. Res. 290 (1) (2021) 99–115.

[5] D. Egger, E. Miguel, S.S. Warren, A. Shenoy, E. Collins, D. Karlan, D. Parkerson, A.M. Mobarak, G. Fink, C. Udry, et al., Falling living standards during the COVID-19 crisis: Quantitative evidence from nine developing countries, Sci. Adv. 7 (6) (2021) eabe0997.

[6] S.S.A. Karim, Q.A. Karim, Omicron SARS-CoV-2 variant: a new chapter in the COVID-19 pandemic, Lancet 398 (10317) (2021) 2126–2128.

[7] W. Dodds, Disease now and potential future pandemics, in: The World's Worst Problems, Springer, 2019, pp. 31–44.

[8] C. Castillo-Chavez, R. Curtiss, P. Daszak, S.A. Levin, O. Patterson-Lomba, C. Perrings, G. Poste, S. Towers, Beyond Ebola: Lessons to mitigate future pandemics, Lancet Global Health 3 (7) (2015) e354–e355.

[9] R.J. Hyndman, G. Athanasopoulos, Forecasting: Principles and Practice, OTexts, 2018.

[10] N.H. Chan, Time Series: Applications to Finance with R and S-Plus, Vol. 837, John Wiley & Sons, 2011.

[11] R.J. Hyndman, S. Fan, Density forecasting for long-term peak electricity demand, IEEE Trans. Power Syst. 25 (2) (2009) 1142–1153.

[12] J.P. Ioannidis, S. Cripps, M.A. Tanner, Forecasting for COVID-19 has failed, Int. J. Forecast. (2020).

[13] X. Zhu, G. Zhang, B. Sun, A comprehensive literature review of the demand forecasting methods of emergency resources from the perspective of artificial intelligence, Nat. Hazards 97 (1) (2019) 65–82.

[14] M. Gul, E. Celik, An exhaustive review and analysis on applications of statistical forecasting in hospital emergency departments, Health Syst. 9 (4) (2020) 263–284.

[15] I. Rahimi, F. Chen, A.H. Gandomi, A review on COVID-19 forecasting models, Neural Comput. Appl. (2021) 1–11.

[16] D.G.B. de Souza, F.T.A. Júnior, N.Y. Soma, Forecasting COVID-19 cases at the Amazon region: a comparison of classical and machine learning models, BioRxiv (2020).

[17] T.T. da Silva, R. Francisquini, M.C. Nascimento, Meteorological and human mobility data on predicting COVID-19 cases by a novel hybrid decomposition method with anomaly detection analysis: a case study in the capitals of Brazil, Expert Syst. Appl. (2021) 115190.

[18] M.A. Capistran, A. Capella, J.A. Christen, Forecasting hospital demand in metropolitan areas during the current COVID-19 pandemic and estimates of lockdown-induced 2nd waves, PLoS One 16 (1) (2021) e0245669.

[19] E. Koç, M. Türkoğlu, Forecasting of medical equipment demand and outbreak spreading based on deep long short-term memory network: the COVID-19 pandemic in Turkey, Signal Image Video Process. (2021) 1–9.

[20] I. COVID, C.J. Murray, et al., Forecasting the impact of the first wave of the COVID-19 pandemic on hospital demand and deaths for the USA and European economic area countries, MedRxiv (2020).

[21] I. COVID, C.J. Murray, et al., Forecasting COVID-19 impact on hospital bed-days, ICU-days, ventilator-days and deaths by US state in the next 4 months, MedRxiv (2020).

[22] M. Goic, M.S. Bozanic-Leal, M. Badal, L.J. Basso, COVID-19: Short-term forecast of ICU beds in times of crisis, PLoS One 16 (1) (2021) e0245272.

[23] R.K. Borchering, C. Viboud, E. Howerton, C.P. Smith, S. Truelove, M.C. Runge, N.G. Reich, L. Contamin, J. Levander, J. Salerno, et al., Modeling of future COVID-19 cases, hospitalizations, and deaths, by vaccination rates and nonpharmaceutical intervention scenarios—United States, April–September 2021, Morb. Mortal. Wkly. Rep. 70 (19) (2021) 719.

[24] C. Rivera-Rodriguez, B.P. Urdinola, Predicting hospital demand during the COVID-19 outbreak in Bogota, Colombia, Front. Public Health 8 (2020) 710.

[25] C. Massonnaud, J. Roux, P. Crépey, COVID-19: Forecasting short term hospital needs in France, Medrxiv (2020).

[26] H.M. Nguyen, P.J. Turk, A.D. McWilliams, Forecasting COVID-19 hospital census: A multivariate time-series model based on local infection incidence, JMIR Public Health Surveill. 7 (8) (2021) e28195.

[27] P. Berta, P. Paruolo, S. Verzillo, P.G. Lovaglio, A bivariate prediction approach for adapting the health care system response to the spread of COVID-19, Plos One 15 (10) (2020) e0240150.

[28] WHO, Coronavirus disease 2019 (COVID-19) pandemic, 2021, URL: https://covid19.who.int/. Accessed on 30.07.2021.

[29] Ministry of Health of Brazil, Painel coronavírus, 2021, URL: https://covid.saude.gov.br/. Acessed on 30.07.2021.

[30] N.R. Faria, T.A. Mellan, C. Whittaker, I.M. Claro, D.d.S. Candido, S. Mishra, M.A. Crispim, F.C. Sales, I. Hawryluk, J.T. McCrone, et al., Genomics and epidemiology of the P. 1 SARS-CoV-2 lineage in Manaus, Brazil, Science 372 (6544) (2021) 815–821.

[31] IBGE, IBGE cidades 2021, 2021, URL: https://cidades.ibge.gov.br/. Accessed on 30.07.2021.

[32] R. Francisquini, A.C. Lorena, M.C.V. Nascimento, Community-based anomaly detection using spectral graph filtering, Appl. Soft Comput. (2022) http://dx.doi.org/10.1016/j.asoc.2022.108489.

[33] CCE-SP, Centro de conntingência do Estado de São Paulo, 2021, URL: https://www.saopaulo.sp.gov.br/coronavirus/. Accessed on 30.07.2021.

[34] CCE-SP, Plano São Paulo, 2021, URL: https://www.saopaulo.sp.gov.br/planosp/. Accessed on 30.07.2021.

[35] S.A. Lauer, K.H. Grantz, Q. Bi, F.K. Jones, Q. Zheng, H.R. Meredith, A.S. Azman, N.G. Reich, J. Lessler, The incubation period of coronavirus disease 2019 (COVID-19) from publicly reported confirmed cases: estimation and application, Ann. Int. Med. 172 (9) (2020) 577–582.

[36] L.A. Dos Santos, P.G. de Góis Filho, A.M.F. Silva, J.V.G. Santos, D.S. Santos, M.M. Aquino, R.M. de Jesus, M.L.D. Almeida, J.S. da Silva, D.M. Altmann, et al., Recurrent COVID-19 including evidence of reinfection and enhanced severity in thirty Brazilian healthcare workers, J. Infect. 82 (3) (2021) 399–406.

[37] W. He, G.Y. Yi, Y. Zhu, Estimation of the basic reproduction number, average incubation time, asymptomatic infection rate, and case fatality rate for COVID-19: Meta-analysis and sensitivity analysis, J. Med. Virol. 92 (11) (2020) 2543–2550.

[38] C. Faes, S. Abrams, D. Van Beckhoven, G. Meyfroidt, E. Vlieghe, N. Hens, et al., Time between symptom onset, hospitalisation and recovery or death: Statistical analysis of belgian covid-19 patients, Int. J. Environ. Res. Public Health 17 (20) (2020) 7560.

[39] R.N. Thompson, F.A. Lovell-Read, U. Obolski, Time from symptom onset to hospitalisation of coronavirus disease 2019 (COVID-19) cases: implications for the proportion of transmissions from infectors with few symptoms, J. Clin. Med. 9 (5) (2020) 1297.

[40] F.E.T. Lima, N.L.S.d. Albuquerque, S.d.S.G. Florencio, M.G.M. Fontenele, A.P.O. Queiroz, G.A. Lima, L.M.d. Figueiredo, S.M.C. Amorim, L.P. Barbosa, Time interval between onset of symptoms and COVID-19 testing in Brazilian state capitals, August 2020, Epidemiol. Serv. Saúde 30 (2020) e2020788.

[41] J.L. Myers, A. Well, R.F. Lorch, Research Design and Statistical Analysis, Routledge, 2010.

[42] S.J. Taylor, B. Letham, Forecasting at scale, Amer. Statist. 72 (1) (2018) 37–45.

[43] C. Guo, Q. Ge, H. Jiang, G. Yao, Q. Hua, Maximum power demand prediction using fbprophet with adaptive Kalman filtering, IEEE Access 8 (2020) 19236–19247.

[44] L. Guo, W. Fang, Q. Zhao, X. Wang, The hybrid PROPHET-SVR approach for forecasting product time series demand with seasonality, Comput. Ind. Eng. 161 (2021) 107598.

[45] G. Battineni, N. Chintalapudi, F. Amenta, Forecasting of COVID-19 epidemic size in four high hitting nations (USA, Brazil, India and Russia) by Fb-Prophet machine learning model, Appl. Comput. Inform. (2020).

[46] M. Indhuja, P. Sindhuja, Prediction of covid-19 cases in India using prophet, Int. J. Stat. Appl. Math 5 (4) (2020).

[47] H. Lu, Y. Li, M. Chen, H. Kim, S. Serikawa, Brain intelligence: go beyond artificial intelligence, Mob. Netw. Appl. 23 (2) (2018) 368–375.

[48] Z. Chen, H. Lu, S. Tian, J. Qiu, T. Kamiya, S. Serikawa, L. Xu, Construction of a hierarchical feature enhancement network and its application in fault recognition, IEEE Trans. Ind. Inf. 17 (7) (2020) 4827–4836.

[49] O.B. Sezer, M.U. Gudelek, A.M. Ozbayoglu, Financial time series forecasting with deep learning: A systematic literature review: 2005–2019, Appl. Soft Comput. 90 (2020) 106181.

[50] H. Abbasimehr, M. Shabani, M. Yousefi, An optimized model using LSTM network for demand forecasting, Comput. Ind. Eng. 143 (2020) 106435.

[51] M. Tan, S. Yuan, S. Li, Y. Su, H. Li, F. He, Ultra-short-term industrial power demand forecasting using LSTM based hybrid ensemble learning, IEEE Trans. Power Syst. 35 (4) (2019) 2937–2948.

[52] H. Su, E. Zio, J. Zhang, M. Xu, X. Li, Z. Zhang, A hybrid hourly natural gas demand forecasting method based on the integration of wavelet transform and enhanced Deep-RNN model, Energy 178 (2019) 585–597.

[53] V.K.R. Chimmula, L. Zhang, Time series forecasting of COVID-19 transmission in Canada using LSTM networks, Chaos Solitons Fractals 135 (2020) 109864.

[54] İ. Kırbaş, A. Sözen, A.D. Tuncer, F.Ş. Kazancıoğlu, Comparative analysis and forecasting of COVID-19 cases in various European countries with ARIMA, NARNN and LSTM approaches, Chaos Solitons Fractals 138 (2020) 110015.

[55] I.E. Livieris, E. Pintelas, P. Pintelas, A CNN–LSTM model for gold price time-series forecasting, Neural Comput. Appl. 32 (23) (2020) 17351–17360.

[56] D.G.B. de Souza, E.A. dos Santos, F.T. Alves Júnior, M.C.V. Nascimento, On comparing cross-validated forecasting models with a novel fuzzy-TOPSIS metric: A COVID-19 case study, Sustainability 13 (24) (2021) 13599.

[57] F. Kadri, K. Abdennbi, RNN-based deep-learning approach to forecasting hospital system demands: application to an emergency department, Int. J. Data Sci. 5 (1) (2020) 1–25.