# Comparison of Models for Predicting the Number of Calls Received in a Call Center through Time Series Analysis

Abraham Gutiérrez *, Jesús Bobadilla, and Santiago Alonso

Computer Systems Department, Polytechnic University of Madrid, Madrid, Spain
Email: abraham.gutierrez@upm.es (A.G.); jesus.bobadilla@upm.es (J.B.); santiago.alonso@upm.es (S.A.)
*Corresponding author

***Abstract*—Time series analysis is a crucial aspect of machine learning that deals with data points ordered by time. Time series data is prevalent in various domains, including finance, economics, healthcare, weather forecasting, and many others. Understanding and modeling time series data is essential for making predictions, identifying trends, and extracting meaningful insights. Effectively modeling time series data is a complex task that requires a combination of statistical methods, machine learning algorithms, and domain-specific knowledge. The choice of a specific model depends on the characteristics of the data and the goals of the analysis or prediction task. Our research provides an innovative method to carry out the analysis of time series data. This method is based on successive sequential steps to perform the temporal analysis. Each step is explained theoretically, and then tested on real data. Furthermore, we apply and compare different models based on both statistical approaches, i.e., Seasonal Autoregressive Integrated Moving Average (SARIMA), Seasonal Autoregressive Integrated Moving Average + Exogenous Variables (SARIMAX), and neural networks, i.e., Long Short-Term Memory (LSTM). For the comparison between the models, the Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) metrics are used, as well as another empirical metric provided by the collaborating company. In each methodology, two verticals are defined, one with exogenous variables and the other without them. The conclusions of the study show that considering the nature of the data analyzed, the model based on neural networks using exogenous variables is the one that provides the best results.**

***Keywords*—artificial intelligence, machine learning, time series, neural networks, AI applied to industry**

## I. Introduction

Time series analysis serves as a vital tool in business, offering a systematic approach to unravel patterns and glean insights from data that unfolds over time. Companies leverage time series analysis in a spectrum of applications, from sales forecasting and supply chain optimization to analyzing financial markets [1] and tracking economic indicators [2], but also to optimize operations, effective resource allocation, customer interactions over time, identification of unusual patterns or anomalies in business data, healthcare systems [3], weather forecasting [4], agricultural commodities [5], retail [6], etc. The overall goal is to improve operational efficiency, make informed decisions and adapt to changing market conditions.

Time series forecasting is the process of analyzing time series data using statistics and modeling to make predictions and inform strategic decision-making [7]. It is not always an exact prediction, and likelihood of forecasts can vary wildly—especially when dealing with the commonly fluctuating variables in time series data and factors outside our control. However, forecasting insight about which outcomes are more likely—or less likely—to occur than other potential outcomes. Naturally, there are limitations when dealing with the unpredictable and the unknown [8].

A repertoire of models and techniques is employed in time series analysis and forecasting, ranging from classical methods to advanced approaches such as machine learning models and exponential smoothing methods. While traditional methods have focused on parametric models informed by domain expertise—such as Autoregressive (AR) [9–11], exponential smoothing [12, 13] or structural time-series models [14]—modern machine learning methods provide a means to learn temporal dynamics in a purely data-driven manner. With the increasing data availability and computing power in recent times, machine learning has become a vital part of the next generation of time-series forecasting models [15–18].

Our research provides an innovative method to carry out the analysis of time series data. This method is based on successive sequential steps to perform the temporal analysis. Each step is explained theoretically, and then tested on real data provided by one of the main telecommunications companies in Spain. The goal is to predict the volume of calls made to its call center. Predicting the number of calls enables us to optimize the workforce at the call center, ensuring we have an adequate number of agents available to provide customers with quality and immediate service.

This paper applies and compares different models based on both statistical approaches, i.e., Seasonal

Manuscript received January 12, 2024; revised March 6, 2024; accepted May 29, 2024; published November 27, 2024.

Autoregressive Integrated Moving Average (SARIMA), Seasonal Autoregressive Integrated Moving Average + Exogenous Variables (SARIMAX) and neural networks, i.e., Long Short-Term Memory (LSTM) [18–21]. For the comparison between the models, the MAE and RMSE metrics are used, as well as another empirical metric provided by the collaborating company. In each methodology, two verticals are defined, one with exogenous variables and the other without them.

The structure of the article is composed of a first section in which the theoretical framework related to the analysis of time series is described. Subsequently, it moves on to the experimentation section, outlining the steps to be taken for the corresponding analysis based on the selected models. Finally, the study's conclusions and bibliographic references are included.

## II. Theoretical Framework

The mathematical formulation of time series involves expressing the underlying structure of the data in mathematical terms. Time series data is a sequence of observations collected or recorded over time, and various mathematical models are used to represent and analyze this data. Here are some fundamental concepts and mathematical formulations commonly used in time series analysis.

A time series is typically denoted as:

$$\{\mathrm{y_t} = \mathrm{T_t} + \mathrm{S_t} + \varepsilon_\mathrm{t}\} \tag{1}$$

where $y_t$ represents the observation at time t; $T_t$ is the trend component (long-term movement or direction in the data); $S_t$ is the seasonal component (repeating patterns or cycles at fixed intervals), and $\varepsilon_t$ are the residuals or errors (random fluctuations or noise in the data).

The Autoregressive Integrated Moving Average (ARIMA) model is a widely used time series model that combines Autoregression (AR), Differencing (I), and Moving Averages (MA). The notation for an ARIMA model is ARIMA (*p*, *d*, *q*), where *p* is the order of autoregression, d is the degree of differencing, and q is the order of the moving average. Denoted by:

$$\Phi(B)(1-B)^d y_t = \Theta(B)\varepsilon_t \tag{2}$$

where $\Phi(B)$ and $\Theta(B)$ are polynomials in the lag operator B, representing autoregressive and moving average components, respectively.

The Exponential Smoothing State Space (ETS) models represent time series data using error, trend, and seasonality components. The notation for ETS models is $ETS(Error, Trend, Seasonal)$ and is typically denoted as:

$$y_t = level_{t-1} + trend_{t-1} + seasonal_{t-m} + \varepsilon_t \tag{3}$$

The Seasonal Decomposition of Time Series (STL) decomposes a time series into trend, seasonal, and residual components. The decomposition can be represented as:

$$y_t = T_t + S_t + R_t \tag{4}$$

The General Autoregressive Conditional Heteroskedasticity (GARCH) models are used for modeling volatility in time series data, particularly in financial markets. The GARCH (*p*, *q*) model is represented as:

$$\sigma_t^2 = \alpha_0 + \sum_{i=1}^{p} \alpha_i \varepsilon_{t-i}^2 + \sum_{j=1}^{q} \beta_j \sigma_{t-j}^2 \tag{5}$$

Here, $\alpha_0$, $\alpha_i$, $\beta_j$ are model parameters, $\varepsilon_{t-i}$ are the lagged squared residuals, and $\sigma_{t-j}$ are the lagged conditional variances.

These mathematical formulations represent some of the key models used in time series analysis. The choice of a specific model depends on the characteristics of the time series data and the goals of analysis or forecasting. Model parameters are typically estimated using statistical methods, and the model performance is evaluated using various metrics, such as mean squared error or likelihood-based criteria.

### A. The SARIMA Models

In this work, we will use a Seasonal Autoregressive Integrated Moving Average model that is an extension of the ARIMA model that incorporates seasonality. SARIMA is particularly useful for time series data that exhibit patterns that repeat at fixed intervals, such as monthly or quarterly seasonality. The SARIMA model is denoted as SARIMA (*p*, *d*, *q* (*P*, *D*, *Q*) *s*), where *p*, *d*, and *q* are the non-seasonal orders, and *P*, *D*, and *Q* are the seasonal orders, with s representing the length of the seasonal cycle. The components of SARIMA include:

- Non-seasonal Autoregressive component:

$$\phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} \tag{6}$$

- Seasonal Autoregressive component:

$$\Phi_1 y_{t-s} + \Phi_2 y_{t-2s} + \cdots + \Phi_P y_{t-Ps} \tag{7}$$

- Non-seasonal differencing:

$$(1-B)^d (1-B^s)^D y_t \tag{8}$$

Here, B is the backshift operator.

- Non-seasonal Moving Average component:

$$\theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q} \tag{9}$$

- Seasonal Moving Average component:

$$\Theta_1 \varepsilon_{t-s} + \Theta_2 \varepsilon_{t-2s} + \cdots + \Theta_Q \varepsilon_{t-Qs} \tag{10}$$

- Seasonal Component. The seasonal component represents the periodic patterns in the data and is expressed as *St*, where *t* is the time index. The SARIMA model is then formulated as:

$$\begin{aligned}\left(1-\phi_1 B-\phi_2 B^2-\cdots-\phi_p B^p\right)(1-B)^d(1- \\ B^s)^D y_t = \left(1+\theta_1 B+\theta_2 B^2+\cdots+\theta_q B^q\right)\left(1+\Theta_1 B^s+\right. \\ \left.\Theta_2 B^{2s}+\cdots+\Theta_Q B^{Qs}\right)\varepsilon_t\end{aligned} \tag{11}$$

In summary, the SARIMA model is a powerful tool for modeling and forecasting time series data with both non-seasonal and seasonal patterns. Selecting appropriate values for *p*, *d*, *q*, *P*, *D*, and *Q* requires careful analysis of the time series characteristics, and model parameters are typically estimated using statistical methods. SARIMA models are widely used in various fields, including

finance, economics, and environmental science, where seasonality is a significant factor in the data.

### B. The SARIMAX Model

The Seasonal Autoregressive Integrated Moving Average with Exogenous Variables model is an extension of the SARIMA model that incorporates exogenous or external variables. Exogenous variables are additional time series that may influence the target variable but are not predicted by the model. SARIMAX is a powerful tool for time series analysis, as it includes of external factors that may impact the behavior of the time series being modeled.

The general form of the SARIMAX model is SARIMAX (*p*, *d*, *q* (*P*, *D*, *Q*)*s*), where *p*, *d*, and *q* are the non-seasonal orders, *P*, *D*, and *Q* are the seasonal orders, and *s* is the length of the seasonal cycle. The addition of exogenous variables introduces them into the model, and the structure becomes SARIMAX (*p*, *d*, *q* (*P*, *D*, *Q*)*s*)(*Px*, *Dx*, *Qx*), where *Px*, *Dx*, and *Qx* represent the orders of the exogenous variables. The SARIMAX model is formulated as follows:

$$\left(1-\phi_1 B-\phi_2 B^2-\cdots-\phi_p B^p\right)\left(1-\Phi_1 B^s-\Phi_2 B^{2s}-\Phi_p B^{Ps}\right)(1-B)^d(1-B^s)^D y_t = \left(1+\theta_1 B+\theta_2 B^2+\cdots+\theta_q B^q\right)\left(1+\Theta_1 B^s+\Theta_2 B^{2s}+\cdots+\Theta_Q B^{Qs}\right)(1+\beta_1 \mathrm{B}+\beta_2 \mathrm{B}^2+\cdots+\beta_p \mathrm{B}^p)x_t \quad (12)$$

Here, $y_t$ is the time series being modeled, $x_t$ represents the exogenous variable, and $\varepsilon_t$ is the error term. The $\beta$ coefficients represent the impact of the exogenous variable on the time series.

Key points about SARIMAX:

1. **Exogenous Variables**: SARIMAX allows for including of one or more exogenous variables, which can improve the model's predictive performance by capturing additional information that influences the target variable.
2. **Model Selection**: The selection of the orders (*p*,*d*,*q*,*P*,*D*,*Q*,*Px*,*Dx*,*Qx*) involves analyzing the autocorrelation and partial autocorrelation functions, as well as considering the nature of the time series and the potential impact of exogenous variables.
3. **Estimation and Forecasting**: Model parameters are typically estimated using methods like maximum likelihood estimation. Once the model is fitted, it can be used to forecast future time series values.
4. **Diagnostic Checks**: Diagnostic checks, such as residual analysis, are essential to ensure that the model adequately captures the patterns in the time series and that the residuals are white noise.

SARIMAX models are widely applied in various fields, including economics, finance, and environmental science, where external factors can significantly influence the observed time series. When dealing with complex time series data influenced by both internal dynamics and external variables, SARIMAX provides a flexible framework for accurate modeling and forecasting.

### C. Handling Stationarity

Handling non-stationarity is crucial in time series analysis, as many statistical models assume that the underlying data is stationary. A stationary time series has constant statistical properties over time, such as a constant mean and variance. This concept is essential, since most models, such as ARIMA, require the series to be stationary. Here are some standard methods for handling non-stationarity in time series:

**Differencing.** Removes the seasonal effect by subtracting the observation at time t from the observation at time t−s, where s is the length of the season.

**Trend-Seasonal Decomposition of Time Series (STL)**. Separates the different components of the time series using a process like Seasonal-Trend decomposition using LOESS.

**Detrending**. Removing the trend component from the time series using linear regression or polynomial fitting techniques.

**Transformations**. This method stabilize the variance and make the time series more amenable to modeling. Log-transformations are often used when dealing with data that exhibits exponential growth.

**Augmented Dickey-Fuller Test (ADF)**. Statistical test to check for the presence of a unit root, indicating non-stationarity. If a unit root is found, differencing may be applied.

**The Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test**. Statistical test used to determine the stationarity of a time series. Unlike ADP test, is designed to test the null hypothesis of stationarity around a deterministic trend.

The choice of method depends on the specific characteristics of the time series data and the goals of the analysis or forecasting task. It often involves experimenting with different approaches and evaluating their effectiveness in achieving stationarity.

### D. The LSTM Models

Recurrent Neural Networks (RNNs) are a class of artificial neural networks designed for processing sequential and time-series data. Unlike traditional feedforward neural networks, RNNs have connections that form directed cycles, allowing them to maintain a hidden state that captures information from previous time steps. This makes RNNs well-suited for tasks where the order and context of input data are important, such as natural language processing, speech recognition, and time series analysis [16–18].

RNNs tend to have difficulties capturing long-term dependencies in sequences. Long Short-Term Memory (LSTM) [18–21] is a type of Recurrent Neural Network (RNN) architecture designed to overcome the limitations of traditional RNNs in capturing long-term dependencies in sequential data. LSTMs were introduced by Hochreiter and Schmidhuber in 1997 and have since become widely used for various applications, including time series forecasting, natural language processing, and speech recognition.

The core idea behind LSTM is the introduction of a memory cell, which allows the network to maintain and update information over long sequences. The memory cell acts as a storage unit, selectively adding or removing information. LSTMs have three gates that control the flow of information into and out of the memory cell: one gate

that regulates the input into the memory cell, another gate that decides what information to discard from the memory cell and an output gate that determines the output based on the current input and the memory cell content.

In an LSTM network two states are defined, the cell and the hidden states. The cell state is the internal memory of the LSTM. It runs straight down the entire chain, with only minor linear interactions. The cell state allows information to flow through the network without being altered, which helps in preserving long-term dependencies. The hidden state is the output of the LSTM at a particular time step. It is a filtered version of the cell state and contains information the model has deemed relevant for the task.

LSTMs are trained using BackPropagation Through Time (BPTT), where gradients are computed through the entire sequence. The use of the memory cell and gates facilitates the training of deep networks over long sequences without the vanishing gradient problem often encountered in traditional RNNs.

LSTMs have demonstrated remarkable success in modeling and predicting sequential data, making them a popular choice for a wide range of applications where understanding and capturing long-term dependencies are crucial [16].

## III. Experiments

The first thing to consider is the amount of data at hand. This is a constant across all types of analysis, and time series analysis forecasting is no exception. However, forecasting relies heavily on the amount of data, possibly even more so than other analyses. It builds directly off past and current data. Having a limited amount of data for extrapolation can lead to less accurate forecasting.

The forecast time frame also matters. This is known as a time horizon—a fixed point in time where a process (like the forecast) ends. Forecasting a shorter time horizon with fewer variables is generally easier compared to forecasting a longer time horizon. The longer the time horizon, the more unpredictable the variables will be. Alternatively, having less data can sometimes work with forecasting if time horizons are adjusted.

As always with analysis, the best analysis is only valid if the data is of a usable quality. Dirty, poorly, overly, or inadequate processed and collected data can significantly skew results and create wildly inaccurate forecasts.

Our proposed method is a sequential algorithm that involves a series of steps that start with data collection and preprocessing and, end with the generation of corresponding forecasts. These steps may vary based on the characteristics of the data and the specific objectives of the analysis. The subsections in this section provide a general summary of the key steps in processing time series data. Following these steps can ensure that time series data is adequately prepared and analyzed, resulting in more accurate and meaningful insights or predictions.

### A. Data Collection

Collect and assemble the time series data from relevant sources, ensuring that the data covers the desired period and includes all necessary variables.

In our case, the provided data corresponds to all calls made during two years (2021 to 2022), averaging approximately 250 K calls per month. This information allows grouping at the day, week, month, and year levels. The data is collected from the Contact Center's Interactive Voice Response (IVR) system, which interacts with callers, gathers information, and routes calls to the appropriate recipient. The IVR Data Collector saves information that callers provide (IVR slots). This information includes the caller's ID, agent ID, date/time, the telephone number from which the caller made the call, and a voice recording. The preprocessing on the data is reduced to keeping the call identifier and the date and time in which it occurred. The project aims to predict the calls received on a daily level; that is why, from now on, we will only consider the data per day.

### B. Data Inspection

Explore the principal characteristics of the time series, including the structure, frequency, and any obvious patterns or trends. Visualizations such as line plots, histograms, and autocorrelation plots can be helpful for initial inspection. The preliminary inspection, shown in Fig. 1, is carried out through a line plot with the data in daily aggregations.

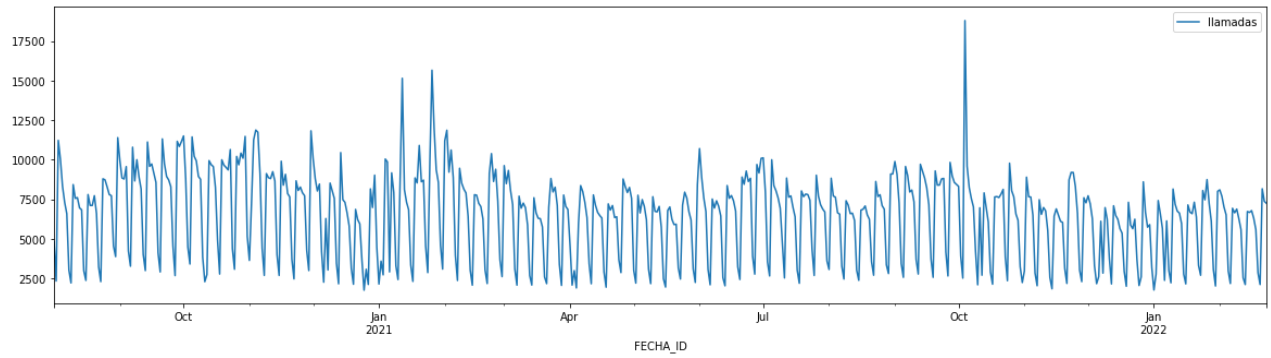


Fig. 1. Data inspection. Data inspection display per day.

In an initial verification of the data provided, an apparent degradation is detected between the months of March and July 2020. This degradation is due to the pandemic caused by the SARS-CoV-2 virus that resulted

in the mandatory home confinement of the country, forcing most of the population to work from home. This situation caused the call center to close temporarily, so the data for that period is unrealible.

### C. Handle Missing Values

Check for missing values in the time series and decide on an appropriate strategy for handling them. This may involve imputation or removing observations with missing values. No problems with missing values are detected after analyzing the starting dataset.

### D. Handle Outliers

Identify and handle outliers or anomalies in the data. Outliers can significantly impact the analysis and modeling process, so it is essential to address them appropriately. To treat outliers, it is first necessary to identify them; to do this, any value outside the range defined by the floor and ceiling will be considered an outlier. Fig. 2 shows graphically the outliers that were found thanks to this method.

$$Floor_{i+s} = \left(\frac{1}{s}\sum_{i=1}^{s} x_i\right) - \sigma \quad (13)$$

$$Ceiling_{i+s} = \left(\frac{1}{s}\sum_{i=1}^{s} x_i\right) + \sigma \quad (14)$$

In time series problems, preserving the data continuity is crucial, and outright removal of outliers is discouraged. Maintaining the flow of information is essential in these scenarios. While conventional methods like using mean or median exist, we have opted for a more robust approach—the Hampel filter. This method, also known as the Hampel identifier, excels at identifying and handling outliers, particularly in time series or ordered datasets. The Hampel filter scrutinizes observations that deviate from the median within a local window or neighborhood. This makes it particularly effective when outliers coexist with regular data points.

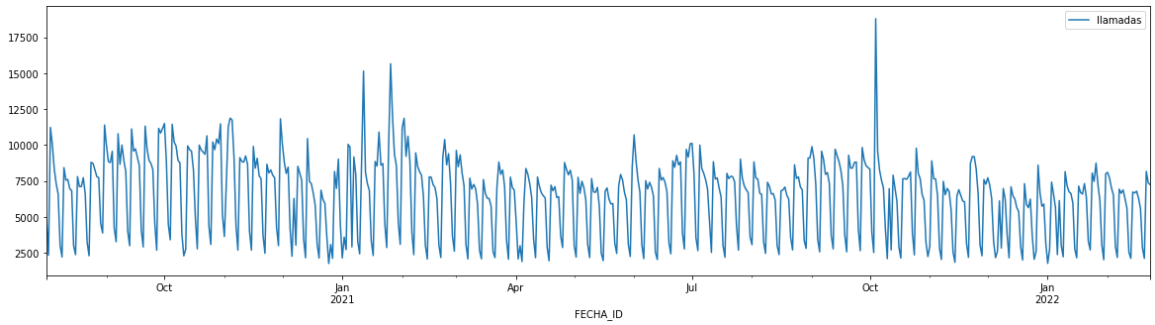


Fig. 2. Outlier detected per day.

### E. Check Frequency, Seasonality and Trend

Confirm that the time series data has a consistent frequency. If there are missing time points, consider imputing or filling in the gaps to create a regular time series. Examine the time series for the presence of seasonality or repeating patterns. This can be done through visual inspection, autocorrelation plots, or statistical tests for seasonality. Investigate the presence of a trend in the time series. Trends can be identified visually or through statistical methods like regression analysis. As shown in Fig. 3, a slight downward trend is observed, which indicates that differentiation of the data is necessary to eliminate this trend.

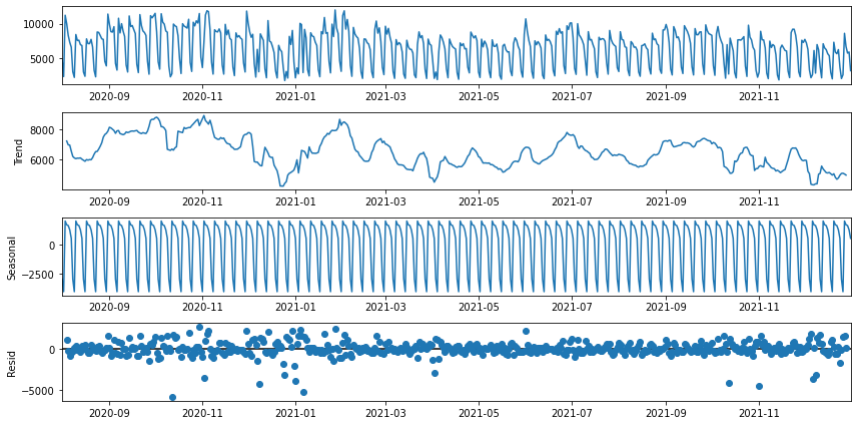


Fig. 3. STL decomposition of raw data.

### F. Stationarity Check

Assess the stationarity of the time series using statistical tests such as the Augmented Dickey-Fuller (ADF) test or the Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test. If the series is non-stationary, consider differencing or other transformations.

TABLE I. ADF AND KPSS TEST ON RAW DATA

| Results | ADF Test | KPSS Test |
|---|---|---|
| Test Statistic | –2.859853 | 0.388766 |
| *p*-value | **0.050220** | **0.016285** |
| #Lags Used | 14 | 14 |
| Number of Observations | 5000000 | 5000000 |
| Critical Value (1%) | –3.443418 | 0.216000 |
| Critical Value (5%) | –2.867303 | 0.146000 |
| Critical Value (10%) | –2.569840 | 0.119000 |

Considering the *p*-value of the ADF test (Table I) in relation to the significance level of 0.05, the null hypothesis cannot be rejected. If the *p*-value is above a critical size, we cannot reject the existence of a unit root. Therefore, the series is non-stationary, and we should consider differentiation.

Considering the *p*-value of the KPSS test (Table I) in relation to the significance level of 0.05, there is evidence for rejecting the null hypothesis in favor of the alternative. Hence, the series is non-stationary according to the KPSS test. It is advisable to apply both tests to ensure the series is truly stationary.

### G. *Differencing or Transformation*

If the time series is non-stationary, apply differencing or other transformations to stabilize the mean and variance. This step may involve removing trends, seasonality, or both. A logarithmic transformation is performed as a first option, but the ADF test results are still inconclusive. Therefore, an ordinal differentiation is carried out. The results of the ADF test corroborate this conclusion. The *p*-value (5.225508e–14) is now much less than the significance level of 0.05, which makes us reject the null hypothesis, and therefore, confirm that the series is now trend stationary.

With the autocorrelation function, it can be seen how there is a certain pattern that repeats every 7 lags, which indicates that seasonal differentiation is necessary, so a seasonal differentiation is carried out on the data. The Dickey-Fuller test is performed again, where it is verified that the null hypothesis continues to be rejected, so the series continues to be stationary due to trend (*p*-value is 3.194650e–10). Now that it has been differentiated ordinally and seasonally, it is possible to interpret the autocorrelation functions in Fig. 4 to extract the parameters of the SARIMA and SARIMAX models.

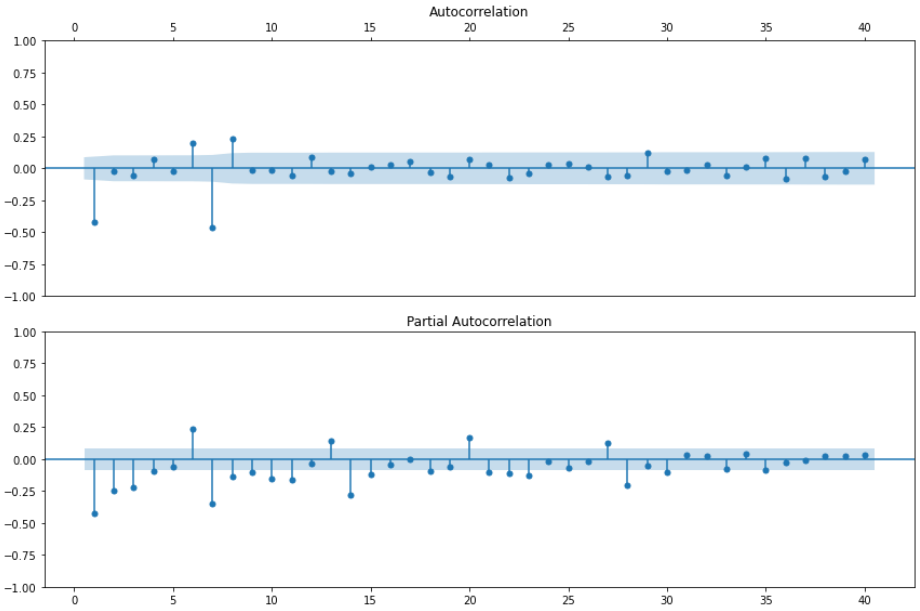


Fig. 4. Simple and partial autocorrelation functions of seasonally differentiated data.

### H. *Feature Engineering*

Create additional features or variables that might be useful for analysis or modeling. For example, extract day-of-week, month, or other temporal features. The feature engineering process is applied to the SARIMAX and LSTM models to use exogenous variables. The model without exogenous variables only has the number of daily calls as a feature. The exogenous variables that expand the model include the day of the week, week, month, year, holiday, special day (though not officially a holiday, a majority of people likely consider it as such), working days of the month, working days of the month without holidays, sending an invoice (the invoice has been sent to the client), and sending SMS (SMS related to any non-payment has been sent).

### I. *Normalization or Scaling*

Normalize or scale the time series data to a consistent range. This step can be important for certain modeling techniques, especially those sensitive to the scale of the input features. Depending on the nature of the data, we choose to perform MinMaxScaler type normalization for numerical data and OneHotEncoder type encoding for categorical features. MinMaxScaler scales and translates each feature individually to be in the given range on the training, in our case, between 0 and 1. MinMaxScaler does not reduce the effect of outliers. However, it linearly scales

them down into a fixed range, where the largest occurring data point corresponds to the maximum value and the smallest one corresponds to the minimum value. OneHotEncoder encodes categorical features as a one-hot numeric array. By default, the encoder derives the categories based on the unique values in each feature.

### J. Modeling

Choose an appropriate modeling approach based on the characteristics of the time series data.

#### 1) SARIMA model

The interpretation of autocorrelation graphs leads to the determination of SARIMA ($p$, $d$, $q$ ($P$, $D$, $Q$)$s$) values:

- $p$ = 0. The partial autocorrelation function shows a structure based on 3 significant lags.
- $d$ = 1. Data has been ordinally differentiated only once.
- $q$ = 1. Single significant delay observed in the simple autocorrelation function.
- $P$ = 0. At the seasonal part of the autocorrelation functions every 7 delays a certain structure is observed.
- $D$ = 1. It has only been seasonally differentiated once.
- $Q$ = 0. Simple autocorrelation function reveals no significant delay in the seasonal part.
- $s$ = 7. Determined by the repetition of the pattern every 7 delays during seasonal differencing.

Table II presents the summary of the most important variables. Special attention should be paid to the *p*-value of ma.L1; if it is greater than 0.05, it would indicate that the model is prone to overfitting. In this case, it is lower, so we will continue with the chosen model.

TABLE II. SARIMA SUMMARY

| | ma.L1 | sigma2 |
|---|---|---|
| coef | –0.7054 | 0.0747 |
| std err | 0.020 | 0.002 |
| *z* | –34.704 | 32.481 |
| *P*>\|*z*\| | **0.000** | 0.000 |
| [0.025 | –0.745 | 0.070 |
| 0.975] | –0.666 | 0.079 |

#### 2) SARIMAX model

In the case of the SARIMAX, the autocorrelation functions have been interpreted differently. The ordinal part remains the same; however, the stationary part would remain:

- $P$ = 3. The partial autocorrelation function has 3 significant lags.
- $D$ = 1. It has only been seasonally differentiated once.
- $Q$ = 1. One of the lags of the autocorrelation function has been considered significant.

Table III presents the summary of the most important variables. It is important to note that some variables have a *p*-value greater than 0.05. This is a good indicator to determine which variables are necessary. In this case, variables x6, x8, x9, and x10 have *p*-values greater than 0.05, so the model must be rebuilt without these variables.

TABLE III. SARIMAX SUMMARY

| | coef | std err | *z* | *P*>\|*z*\| | [0.025 | 0.975] |
|---|---|---|---|---|---|---|
| x1 | –3653.1650 | 104.076 | –35.101 | 0.000 | –3857.151 | –3449.179 |
| x2 | –2871.3935 | 234.900 | –12.224 | 0.000 | –3331.790 | –2410.997 |
| x3 | 1404.5050 | 404.241 | 3.474 | 0.001 | 612.208 | 2196.802 |
| x4 | 7.4570 | 2.107 | 3.539 | 0.000 | 3.327 | 11.587 |
| x5 | –436.4175 | 190.470 | –2.291 | 0.022 | –809.732 | –63.103 |
| x6 | 57.2398 | 121.052 | 0.473 | **0.636** | –180.018 | 294.498 |
| x7 | –289.4350 | 93.146 | –3.107 | 0.002 | –471.998 | –106.872 |
| x8 | 2.5123 | 481.682 | 0.005 | **0.996** | –941.567 | 946.591 |
| x9 | –378.9693 | 208.737 | –1.816 | **0.069** | –788.085 | 30.147 |
| x10 | 3.4417 | 239.960 | 0.014 | **0.989** | –466.870 | 473.754 |
| ma.L1 | –0.5455 | 0.032 | –17.266 | 0.000 | –0.607 | –0.484 |
| ar.S.L7 | 0.0921 | 0.045 | 2.058 | 0.040 | 0.004 | 0.180 |
| ar.S.L14 | 0.0524 | 0.045 | 1.157 | 0.247 | –0.036 | 0.141 |
| ar.S.L21 | 0.0675 | 0.037 | 1.808 | 0.071 | –0.006 | 0.141 |
| ma.S.L7 | –0.9998 | 0.045 | –22.259 | 0.000 | –1.088 | –0.912 |
| sigma2 | 8.729e+05 | 0.004 | 1.94e+08 | 0.000 | 8.73e+05 | 8.73e+05 |

#### 3) LSTM model

The LSTM model is constructed using the Keras library, employing a relatively straightforward structure based on the "Sequential" model. The architecture consists of an input layer with a shape of (31, 1) to capture information from the 31 days preceding the prediction. Subsequently, there is an LSTM layer with 62 neurons, followed by another LSTM layer with 32 neurons. The output is facilitated by a dense layer with 31 neurons, representing the prediction for the subsequent 31 days.

For the LSTM layers, the ReLU function serves as the recurrent activation, while Sigmoid is applied for normal activation. The dense layer adopts a ReLU activation. The model is compiled using the Adam optimizer, and the Mean Squared Error (MSE) is declared as the loss function and metric. This configuration ensures that the model focuses on minimizing the mean squared error during training, providing a robust framework for time series forecasting. Table IV shows the defined architecture:

TABLE IV. MODEL ARCHITECTURE

| Layer (type) | Output Shape | Param # |
|---|---|---|
| input (InputLayer) | [(None, 31, 1)] | 0 |
| lstm_1 (LSTM) | (None, 31, 64) | 16896 |
| lstm_2 (LSTM) | (None, 32) | 12416 |
| dense_out (Dense) | (None, 32) | 1023 |

A validation split of 25% has been defined for training the model. A small batch size of 3 is chosen to prevent a substantial decline in model quality while preserving its generalization capabilities. Training spans 120 epochs, and as observed, overfitting begins beyond this point. Due to the time series nature of the data, the shuffle parameter is set to false, ensuring chronological training order.

*4) LSTM model with exogenous variables*

Prediction with additional variables necessitated the creation of two distinct models. The first model comprises an LSTM layer with 64 neurons featuring sigmoid activation and recurrent ReLU activation, a Dense layer with 31 neurons employing ReLU activation, and a Flatten layer. This model takes "escalations" as input. The second model incorporates a Dense layer with 64 neurons and another Dense layer with 31 neurons, both utilizing "ReLU" activation taking exogenous variables as input. Subsequently, these models are fused using a Multiply layer, followed by a Flatten layer, and finally a Dense layer with 31 neurons with ReLU activation serving as the output layer. Table V shows the defined architecture.

TABLE V. MODEL ARCHITECTURE

| Layer (type) | Output Shape | Param # | Connected to |
|---|---|---|---|
| lstm_input (InputLayer) | [(None, 31, 1)] | 0 | |
| lstm (LSTM) | (None, 64) | 16896 | lstm_input[0][0] |
| dense_1_input (InputLayer) | [(None, 31, 8)] | 0 | |
| dense (Dense) | (None, 31) | 2015 | lstm[0][0] |
| dense_1 (Dense) | (None, 31, 64) | 576 | dense_1_input [0][0]] |
| flatten (Flatten) | (None, 31) | 0 | dense[0][0] |
| dense_2 (Dense) | (None, 31, 31) | 2015 | dense_1[0][0]] |
| multiply (Multiply) | (None, 31, 31) | 0 | flatten[0][0] dense_2[0][0] |
| flatten_1 (Flatten) | (None, 961) | 0 | multiply[0][0] |
| dense_3 (Dense) | (None, 31) | 29822 | flatten_1[0][0] |

Training parameters for this model mirror those of the model without exogenous variables. A validation set encompassing 25% of the training data is established, employing a batch size of 15, and the shuffle is set to false to ensure orderly data processing. The model is trained for 70 epochs, fine-tuning its ability to integrate both escalations and exogenous variables for more nuanced predictions.

*5) Evaluation*

Evaluate the model performance using appropriate metrics, such as Mean Squared Error (MSE), Mean Absolute Error (MAE), or others, depending on the nature of the problem. The company uses a simple calculation as a Metric (CM) to evaluate its models: the percentage obtained by dividing the MAE by the sum of the actual values. Table VI summarizes the comparison between the different models.

TABLE VI. COMPARISON OF RESULTS

| model | MAE | RMSE | CM |
|---|---|---|---|
| SARIMA | 537.88 | 622,18 | 15.37 |
| SARIMAX | 237.63 | 282,78 | 8.25 |
| LSTM | 221.83 | 255,18 | 7.05 |
| LSTM Exo. | 160.22 | 179,31 | 4.05 |

*6) Adjust and iterate*

Iterative refinement is often necessary to improve the model performance. In view of the results obtained, the company considered that no further adjustments were necessary.

## IV. RESULT AND DISCUSSION

As seen in Table VI, the best results in terms of the metrics used are obtained with the LSTM Exo model. This is evident in the level of model fit shown in Fig. 5. For example, Tables VII and VIII present the errors in predictions for both the statistical models and those based on neural networks for the first 5 days of January 2022.

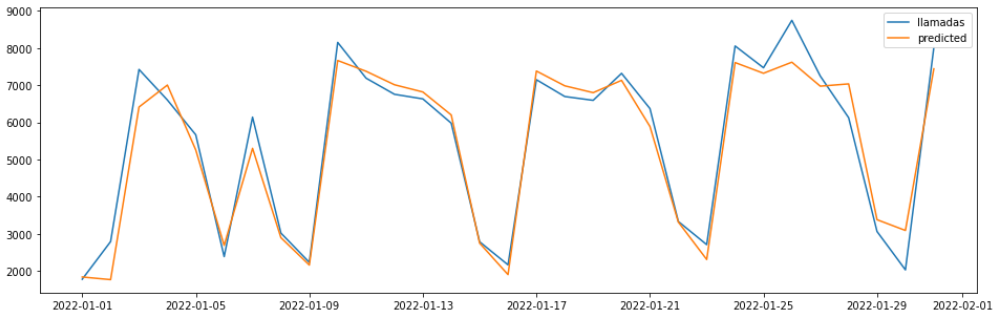


Fig. 5. LSTM model predictions with exogen variables.

TABLE VII. EXAMPLE OF SARIMA AND SARIMAX PREDICTIONS

| date | calls | SARIMA | | SARIMAX | |
|---|---|---|---|---|---|
| | | prediction | error | prediction | error |
| 2022-01-01 | 1777 | 1.954,70 | 177,70 | 1.883,62 | 106,62 |
| 2022-01-02 | 2791 | 3.070,10 | 279,10 | 3.014,28 | 223,28 |
| 2022-01-03 | 7426 | 7.648,78 | 222,78 | 7.648,78 | 222,78 |
| 2022-01-04 | 6603 | 7.395,36 | 792,36 | 6.933,15 | 330,15 |
| 2022-01-05 | 5666 | 6.175,94 | 509,94 | 6.062,62 | 396,62 |

TABLE VIII. EXAMPLE OF LSTM AND LSTM EXO PREDICTIONS

| date | calls | LSTM | | LSTM Exo. | |
|---|---|---|---|---|---|
| | | prediction | error | prediction | error |
| 2022-01-01 | 1777 | 1.880,06 | 103,06 | 1.874,73 | 97,73 |
| 2022-01-02 | 2791 | 2.958,46 | 167,46 | 2.944,50 | 153,50 |
| 2022-01-03 | 7426 | 7.641,35 | 215,35 | 7.567,09 | 141,09 |
| 2022-01-04 | 6603 | 6.900,13 | 297,13 | 6.834,10 | 231,10 |
| 2022-01-05 | 5666 | 5.983,29 | 317,29 | 5.869,97 | 203,97 |

Finally, Fig. 6 confirms that the errors of the test set and the validation set for the LSTM Exo model converge, indicating the absence of overfitting in the model.

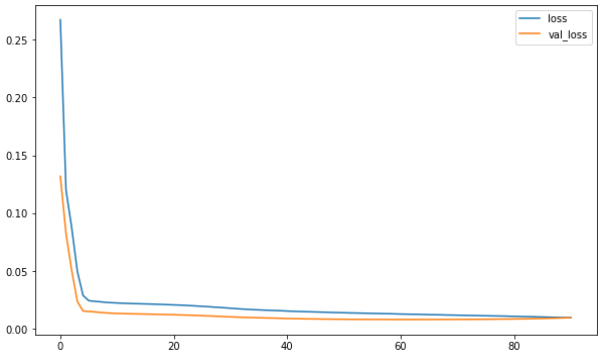


Fig. 6. Error diagnosis of the LSTM model with exogenous variables.

It is important to note that the results obtained depend on the nature of the data and the objectives set for the problem. Regardless, it is necessary to highlight the differences between the different methods used. The distinctions among the results obtained from ARIMA, ARIMAX, and LSTM models in time series analysis are noteworthy:

- Pattern Assumption: ARIMA models assume linear patterns suitable for data with well-defined trends and seasonality. LSTM models capture complex nonlinear patterns, making them effective for data with intricate dependencies and long-term memory.
- Handling Seasonality: While ARIMA models are designed to handle seasonality, they may struggle with highly irregular patterns. LSTM models adapt to and capture both regular and irregular seasonality, making them versatile for diverse temporal patterns.
- Temporal Dependency: ARIMA models assume a fixed temporal dependency structure, limiting their ability to capture long-term dependencies. LSTM models are specifically designed for sequences and can capture long-term dependencies in time series data with extended temporal relationships.
- Interpretability: ARIMA models' results are often more interpretable, given explicit mathematical formulations, making it easier to understand the impact of each component. On the other hand, LSTM models are often considered black boxes, capturing intricate patterns but posing challenges in understanding the underlying logic.
- Computational Intensity: ARIMA models are typically computationally less intensive, making them more suitable for scenarios with limited computational resources. In contrast, LSTM models are more computationally demanding, requiring robust hardware, and may not be suitable for real-time applications with strict computational constraints.

These distinctions highlight the trade-offs between interpretability, computational efficiency, and the ability to capture complex patterns. The choice between ARIMA, ARIMAX, and LSTM models depends on the specific characteristics of the time series data, the goals of the analysis, and the available computational resources.

## V. CONCLUSION

In this paper, several models have been developed capable of predicting the number of calls received in a call center with an approximate precision ranging between 84% and 95%, using a dataset that spans approximately two years of calls.

Considering the objective of modeling time series, two widely used statistical models, SARIMA and SARIMAX, were chosen. Their results have been compared with a neural network-based model (LSTM). For each of these models, two options have been defined: one using exogenous variables and one without.

After observing the performance of the models, it has been concluded that, despite the high precision offered by purely statistical models, the deep learning model (LSTM) exhibits significantly better performance with less and simpler data preprocessing. On the other hand, the results indicate that using exogenous variables in the models is an important component, as these variables provide additional information to the model, allowing it to achieve better results.

Future work would involve comparing the network models developed in the paper with other alternative models such as stacked LSTM, CNN LSTM, and convolutional LSTM among others.

## CONFLICT OF INTEREST

The authors declare no conflict of interest.

## AUTHOR CONTRIBUTIONS

Abraham Gutiérrez and Jesús Bobadilla conducted the research, analyzed data, and wrote portions of the introduction, methodology, results, and conclusion. Santiago Alonso contributed to the introduction and literature review. All authors participated in reviewing and approving the final version.

## FUNDING

This work was partially supported by the Ministerio de Ciencia e Innovación of Spain under the project PID2019-

106493RB-I00 (DL-CEMG) and the Comunidad de Madrid under the Convenio Plurianual with the Universidad Politécnica de Madrid in the actuation line of the Programa de Excelencia para el Profesorado Universitario.

## ACKNOWLEDGMENT

Thank the group Xfera Móviles, S.A.U. for the support provided to carry out the study both at the level of data and technological infrastructure. The project has been developed under the agreement of the Chair in Artificial Intelligence for the Analysis of the Telecommunications Market.

## REFERENCES

[1] O. B. Sezer, M. U. Gudelek, and A. M. Ozbayoglu, "Financial time series forecasting with deep learning: A systematic literature review: 2005–2019," *Applied Soft Computing*, vol. 90, 106181, May 2020.

[2] O. Claveria, E. Monte, and S. Torra, "Economic forecasting with evolved confidence indicators," *Economic Modelling*, vol. 93, pp. 576–585, Dec. 2020.

[3] E. J. Topol, "High-performance medicine: The convergence of human and artificial intelligence," *Nature Medicine*, vol. 25, pp. 44–56, Jan. 2019.

[4] J. M. Dad, M. Muslim, I. Rashid, I. Rashid, and Z. A. Reshi, "Time series analysis of climate variability and trends in Kashmir Himalaya," *Ecological Indicators*, vol. 126, 107690, Jul. 2021.

[5] M. Mudelsee, "Trend analysis of climate time series: A review of methods," *Earth-Science Reviews*, vol. 190, pp. 310–322, Mar. 2019.

[6] R. Murugesan, E. Mishra and A. H. Krishnan, "Forecasting agricultural commodities prices using deep learning-based models: Basic LSTM, bi-LSTM, stacked LSTM, CNN LSTM, and convolutional LSTM," *International Journal of Sustainable Agricultural Management and Informatics*, vol. 8, no. 3, pp. 242–277, Sep. 2022.

[7] Y. Xie, C. Li, M. Li, F. Liu, and M. Taukenova, "An overview of deterministic and probabilistic forecasting methods of wind energy," *Iscience*, vol. 26, no. 1, 105804, Jan. 2023.

[8] V. F. Silva, M. E. Silva, P. Ribeiro, and F. Silva, "Time series analysis via network science: Concepts and algorithms," *WIREs Data Mining and Knowledge Discovery*, vol. 11, no. 3, Mar. 2021.

[9] C. A. Thilker, H. Madsen, and J. B. Jørgensen, "Advanced forecasting and disturbance modelling for model predictive control of smart energy systems," *Applied Energy*, vol. 292, 116889, Jun. 2021.

[10] J. D. Hamilton, *Time Series Analysis*, Princeton University Press, 2020.

[11] S. R. Beeram and S. Kuchibhotla, "Time series analysis on univariate and multivariate variables: A comprehensive survey," *Communication Software and Networks. Lecture Notes in Networks and Systems*, vol. 134, pp. 119–126, Oct. 2021.

[12] H. Hidayatulah and S. Parasian, "Comparison of forecasting accuracy rate of exponential smoothing method on admission of new students," *Journal of Critical Review*, vol. 7, no. 2, pp. 268–274, Jul. 2020.

[13] Q. T. Tran, L. Hao, and Q. K. Trinh, "A comprehensive research on exponential smoothing methods in modeling and forecasting cellular traffic," *Concurrency and Computation: Practice and Experience*, vol. 32, no. 23, e5602, Nov. 2020.

[14] V. Cerqueira, L. Torgo, and I. Mozetič, "Evaluating time series forecasting models: An empirical study on performance estimation methods," *Machine Learning*, vol. 109, pp. 1997–2028, Oct. 2020.

[15] K. Christensen, M. Siggaard, and B. Veliyev, "A machine learning approach to volatility forecasting," *Journal of Financial Econometrics*, vol. 21, no. 5, pp. 1680–1727, Jun. 2022.

[16] R. P. Masini, M. C. Medeiros, and E. F. Mendes, "Machine learning advances for time series forecasting," *Journal of Economic Surveys*, vol. 37, no. 1, pp. 76–111, Jul. 2021.

[17] B. Lim, and S. Zohren, "Time-series forecasting with deep learning: a survey," *A Philosophical Transactions of the Royal Society*, Feb. 2021.

[18] A. K. Dubey, A. Kumar, V. García-Díaz, A. K. Sharma, and K. Kanhaiya, "Study and analysis of SARIMA and LSTM in forecasting time series data," *Sustainable Energy Technologies and Assessments*, vol. 47, 101474, Oct. 2021.

[19] C. Nontapa, C. Kesamoon, N. Kaewhawong, and P. Intrapaiboon, "A new time series forecasting using decomposition method with SARIMAX model," in *Proc. 27th International Conference on Neural Information Processing, ICONIP 2020*, Bangkok, Thailand, Nov. 2020, pp. 743–751.

[20] D. H. Hopfe, K. Lee, and C. Yu, "Short-term forecasting airport passenger flow during periods of volatility: Comparative investigation of time series vs. neural network models," *Journal of Air Transport Management*, vol. 115, 102525, Mar. 2024.

[21] A. Jain, T. Sukhdeve, H. Gadia, S. P. Sahu, and S. Verma, "Covid19 prediction using time series analysis," in *Proc. International Conference on Artificial Intelligence and Smart Systems (ICAIS)*, Mar. 2021, pp. 1599–1606.

[22] W. Yu, I. Y. Kim, and C. Mechefske, "Analysis of different RNN autoencoder variants for time series classification and machine prognostics," *Mechanical Systems and Signal Processing*, vol. 149, 107322, Feb. 2021.

[23] M. Khan, H. Wang, A. Riaz, A. Elfatyany, and S. Karim, "Bidirectional LSTM-RNN-based hybrid deep learning frameworks for univariate time series classification," *The Journal of Supercomputing*, vol. 77, pp. 7021–7045, Jan. 2021.