SEAI RD&D 2019

SMARTeBuses

Forecasting Models for Power Prediction

Deliverable number: WP3-D1

This Project has received funding from the SEAI RD&D Programme.
Grand Award Number: 19/RDD/519

Project Acronym: SMARTeBuses
Project Full Title: SMART electric Buses
Grant Number: 19/RDD/519
Project URL: https://smartebuses.github.io/web/

Deliverable type:	Deliverable / Report
Dissemination level:	Public
Delivery Date:	January 31, 2021
Number of pages:	16
Keywords:	SMARTeBuses, Time Series, Artificial Intelligence, Deep Learning, ARIMA, SARIMA, LSTM
Authors:	Padraigh Jarvis, University College Cork Dr. Alejandro Arbelaez - University College Cork

Contents

List of Figures

List of Acronyms

SMARTeBuses SMART electric Buses
SEAI Sustainable Energy Authority of Ireland
AI Artificial Intelligence
ARIMA Autoregressive Integrated Moving Average
AR Autoregression
MA Moving Average
ARMA Autoregressive-Moving Average
SARIMA Seasonal Autoregresssive Integrated Moving Average
I Integrated
RNN Recurrent Neural Network
LSTM Long-Short Term Memory
ANN Artificial Neural Networks

1 Introduction

This document corresponds to the delivery WP3-D1 "*Forecasting Models for Power Prediction*"' of the SMART electric Buses (SMARTeBuses) project, funded by the Sustainable Energy Authority of Ireland (SEAI) RD&D Programme. This project is classified as Non-economic public Good Research under the European Union (EU) State Aid regulations and will exploit, combine and improve cutting-edge Artificial Intelligence (AI) technologies to develop and implement optimization models for the operation of electric buses in Ireland with operational constraints.

In this deliverable, we describe a set of models for time series analysis and forecasting based on historical data. In particular, we focus our attention in statistical and deep learning models to estimate the production and demand of energy in Ireland. We recall that in this project we plan to use AI to maximize the use of renewable energies in the transportation sector, e.g., infrastructure and charging events of the electric fleet.

Figures 1.1 and 1.2 show the time series plot of the demand and clean energy production in Ireland (at 15-minute intervals) for February 2020. In these figures we observe two different patterns, Figure 1.1 shows a clear daily seasonal pattern where the energy demand peaks between 18:00 and 20:00 hours (with approx. 5000 MW) and the lowest demand levels are typically after midnight. On the other hand, Figure 1.2 shows a higher variability due to the wind power is intermittent and it is only available when the wind flows at certain speeds, the system reached a minimum of 87MW on Feb. 12th at 21:00 and for the same time period wind farms in Ireland produced as much as 3337 WM on Feb. 22nd at 18:45.

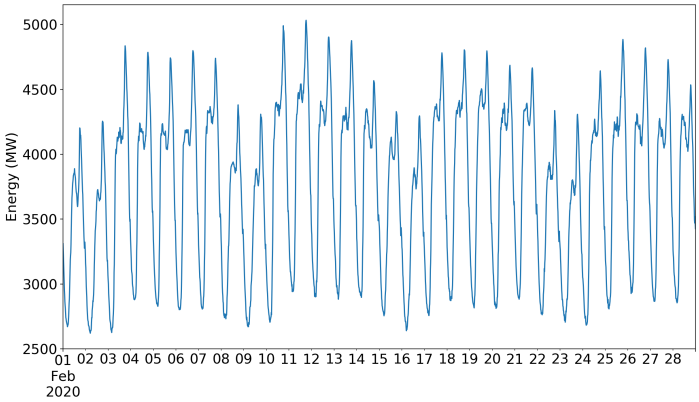

Figure 1.1: Energy Demand in Ireland - Feb 2020

In this deliverable we aim at studying state-of-the-art models that given a sequence of values (or power measurements), Z=$\{Z_1, Z_2, \ldots, Z_n\}$, estimate the next k values in the sequence, i.e., Z_{n+1}, Z_{n+2}, ..., Z_{n+k}. The simplest method to estimate the next element in sequence is to assume that future values are equal to the most recent observation, i.e., $Z_{n+1} = Z_n$. Another simple method consists in predicting the next element as the average of a pre-defined period of time, e.g., an hour or a day. Interestingly, this simple moving average method represents the foundations of the popular Moving Average (MA) method based on statistical properties of the dataset. In this project. we will study the performance

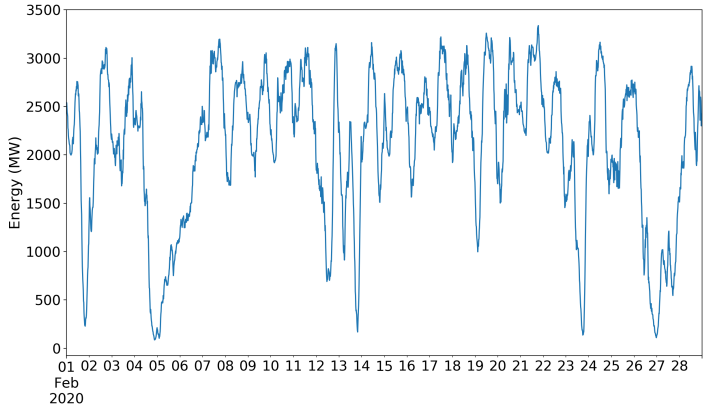

Figure 1.2: Wind Power Production in Ireland - Feb 2020

of the Autoregressive Integrated Moving Average (ARIMA) and Seasonal Autoregresssive Integrated Moving Average (SARIMA) statistical models for energy forecasting.

Alternatively, deep learning models can be employed for time series forecasting. These models are based on Artificial Neural Networks (ANN) with a set of interconnected nodes simulating a human brain. The network includes an input layer representing the time series data for a pre-predefined time window; an output layer with the predictions; and hidden layers. The number and structure of the hidden layers represents the architecture of the network. In this project, we study the performance of the Long-Short Term Memory (LSTM) architecture for the energy forecasting.

This deliverable is structured as follows. Chapter 2 describes main components of the ARIMA and SARIMA models, i.e., Autoregression (AR), MA and the integration of both via the differencing method. Chapter 3 describes the LSTM architecture designed to overcame the limitations of the Recurrent Neural Network (RNN) one. And finally, Chapter 4 provides general conclusions.

2 ARIMA and SARIMA

ARIMA is a well-known statistical method for time series analysis. ARIMA learns patterns from historical data and predicts subsequent values [1]. In this context, ARIMA is capable of producing models for stationary and non-stationary data, e.g., the mean increases (or decreases) with time. Non-stationary data can be transformed into stationary by using the difference between two (or more) subsequent values. Examples of non-stationary data include: finance data [2], biomedical signal processing [3], and audio signal processing [4].

2.1 Random Walk

A random walk time series assumes that the next value in the sequence is a random variation from the previous observed value, i.e., $Z_t = Z_{t-1} + \epsilon_t$, where Z_t denotes the value of the time series at time t, Z_{t-1} denotes the previous data value in the time series, and ϵ_t denotes the error between the two time steps. This model assumes that the random variations are independent and identically distributed (i.i.d).

2.2 Autoregressive Model

The AR component of ARIMA estimates the next value in the time series as a linear combination of the previous values with the shock value (or noise) and a constant value as follows:

$$Z_t = c + \sum_{i=1}^{p} \phi_i Z_{t-i} + \epsilon_t$$

where p denotes the order of the model, Z_{t-i} denotes the past values, ϵ_t denotes a error at time t, ϕ_i are parameters of the model, and c is a constant. parameters.

2.3 Moving Average

The MA component in ARIMA estimates the next value of the time series as a linear combination of the previous values as follows:

$$Z_t = \mu + \sum_{i=1}^{q} \theta_i \epsilon_{t-i}$$

where q denotes the order of the model, μ denotes the mean of the dataset, θ denotes the parameters of the model, and ϵ denotes the error at the different time steps. [5] describes a process to calculate the values of the parameters.

2.4 Non-Stationary Data

A time series is stationary when the mean, variance, and autocovariance remains unchanged or constant over time. Figure 2.1 shows an example of a non-stationary behaviour for the popular Air Passengers dataset available in Kaggle.[1] The dataset describes the number of monthly passengers of a US airlines from 1948 to 1960 and this particular patter is commonly known as *trend* in time series and indicates that the dataset exhibits an increasing (or decreasing) patter changing over time.

The Integrated (I) component of ARIMA uses differencing in order to achieve stationary in the dataset, and therefore, it helps to remove the time dependency by subtracting the previous observation, i.e., $Z'_i = Z_i - Z_{i-1}$ or first degree of differencing. In the general form ARIMA incorporates the degree of differencing d as the number of recursive applications of the first order differences. . For instance, the following formulation represents the differencing method with d=2 or Z''_t.

$$\begin{aligned} Z''_t &= Z'_t - Z'_{t-1} \\ &= (Z_t - Z_{t-1})' - (Z_{t-1} - Z_{t-2}) \\ &= Z_t - 2Z_{t-1} + Z_{t-2} \end{aligned}$$

The first degree typically allows de-trending for datasets with linear trends and the second degree allows quadratic trends. In the general form ARIMA incorporates the degree d as the number of recursive applications of the first order differences.

Seasonality also affect the stationary pattern of the time series and seasonal differencing represents the difference between the current value and the corresponding value in previous period, e.g., for monthly data with 12 periods the difference time will be $Z'_t = Z_i - Z_{t-m}$ with m = 12. The second degree d of seasonal differencing can be expressed as:

$$\begin{aligned} Z''_t &= Z'_t - Z'_{t-m} \\ &= (Z_t - Z_{t-m})' - (Z_{t-m} - Z_{t-1-m}) \\ &= Z_t - 2Z_{t-m} + Z_{t-1-m} \end{aligned}$$

[1] This database is available at https://www.kaggle.com/rakannimer/air-passengers

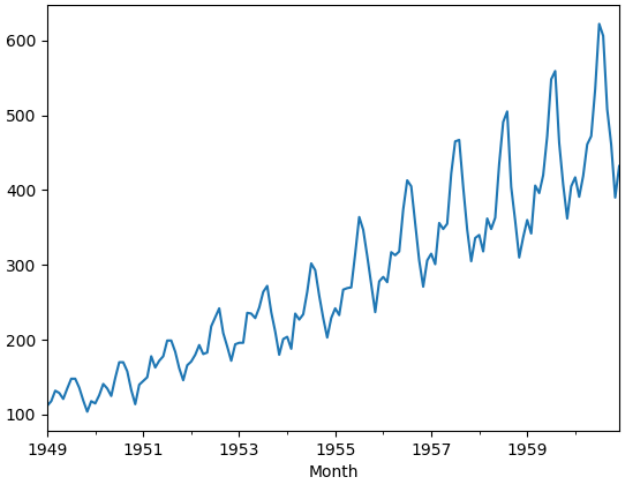

Figure 2.1: Air Passengers Time series - Non-Stationary Data.

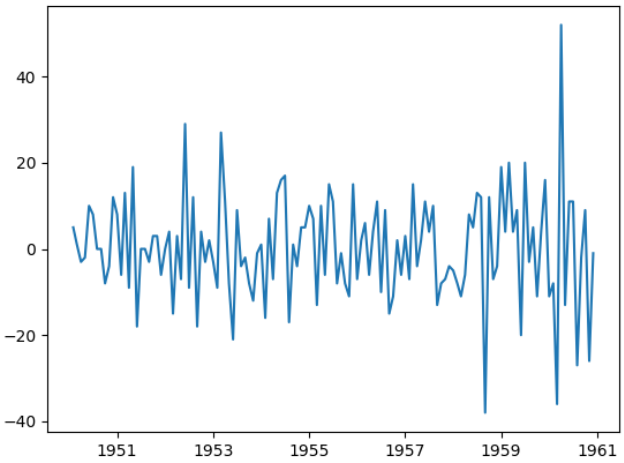

Figure 2.2: Air Passengers Time series After Applying Differencing $d = 12$.

2.5 ARIMA

The two previously mentioned components (AR(p) and MA(q)) form a Mixed Autoregressive-Moving Average (ARMA) Model when combined together for non-seasonal time series. In this context, the ARMA model can be expressed as:

$$Z_t = c + \epsilon_t + \sum_{i=1}^{p} \phi_i Z_{t-i} + \sum_{i=1}^{q} \theta_i \epsilon_{t-i}$$

These models afford more flexibility when fitting to time series data. It is also commonly found that models which contain autoregressive and/or moving average aspects perform best when both the p and q values are set to two or lower. The Integrated aspect of ARIMA models comes from integrating the stationary ARMA process as thus the full mathematical model for ARIMA can be formulated as:

$$Z'_t = c + \epsilon_t + \sum_{i=1}^{p} \phi_i Z'_{t-i} + \sum_{i=1}^{q} \theta_i \epsilon_{t-i}$$

where Z'_t denotes degree (d) of first order differences.

2.6 Seasonal ARIMA

SARIMA takes the concepts defined in ARIMA and extends them using some seasonality components. In the same vein as ARIMA, SARIMA uses variables for autoregressive order, moving average order, and integration (P, Q, and D respectively). These variables are used in addition to the ones outlined before for ARIMA.

SARIMA introduces another variable s which which controls the interval at which similar data trends are observed at. For instance, in an hour data-set if similar trends are observed every 24 hours, a s value of 24 should be used.

SARIMA(1,0,1) $\times$ (1,0,1) with m=12 can formulated as follows:

$$Z_t = \Phi Z_{t-12} + \epsilon_t + \Theta_{t-12}$$

as it can be observed SARIMA calculates the next value in the series similar to ARIMA with lagged values.

3 Long Short Term Memory

Deep learning is an emerging area in the field of AI with multiple applications in a variety of field such as: natural language processing [6], cyber security [7], fraud detection [8], and speech recognition [9]. In this deliverable, we focus in LSTM, a state-of-the-art deep learning architecture, with outstanding results for energy forecasting [10].

3.1 RNN Architecture

The RNN architecture is a popular deep learning technology to process time dependent data. Figure 3.1 depicts the core components of the architecture with three consecutive memory cells that represent the internal memory of the network capable of remembering previous values of the sequence. As it can be observed the each memory unit takes two input parameters, i.e., current value in the sequence (e.g., X_t) and the output of the previous value memory unit (S_{t-1}) and the last memory outputs the prediction.

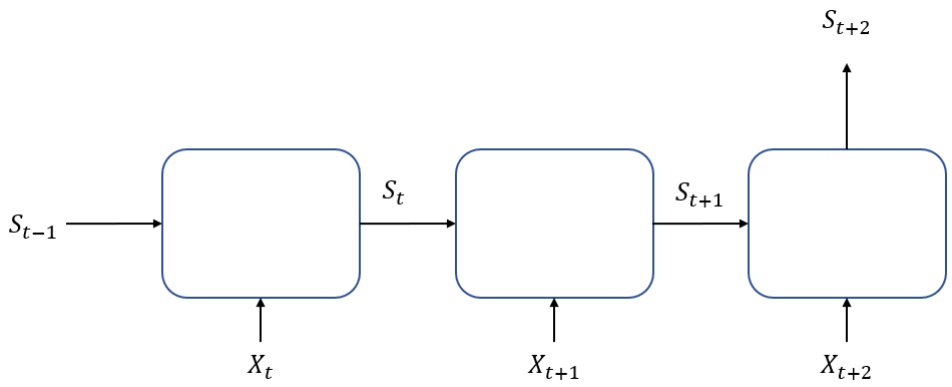

Figure 3.1: RNN architecture.

[11] highlighted that traditional RNNs face a problem with noticing patterns in long term data, in fact only recent data is represented. This means that RNNs are considered to have Short Term memory. As a result traditional RNNs struggle with domains such as music generation and time series pattern recognition, this can lead to back propagation error when building the system. LSTM is a type of RNN introduced by [12], which makes use of constant error flow using constant error carrousels and a gradient-based algorithm.

3.2 LSTM Architecture

In the same vein as RNN, LSTM also organises the network in succesive memory cells to process sequential data. However, unlike the RNN network that requires two input and one output value, the LSTM architecture takes two input and two output values.

Unlike the RNN architecture that takes two input and one output value, the LSTM architecture takes two input and two output values. The additional output value the memory cells, i.e., C_t, denotes the value of the memory unit and regulates the memory function of the network.

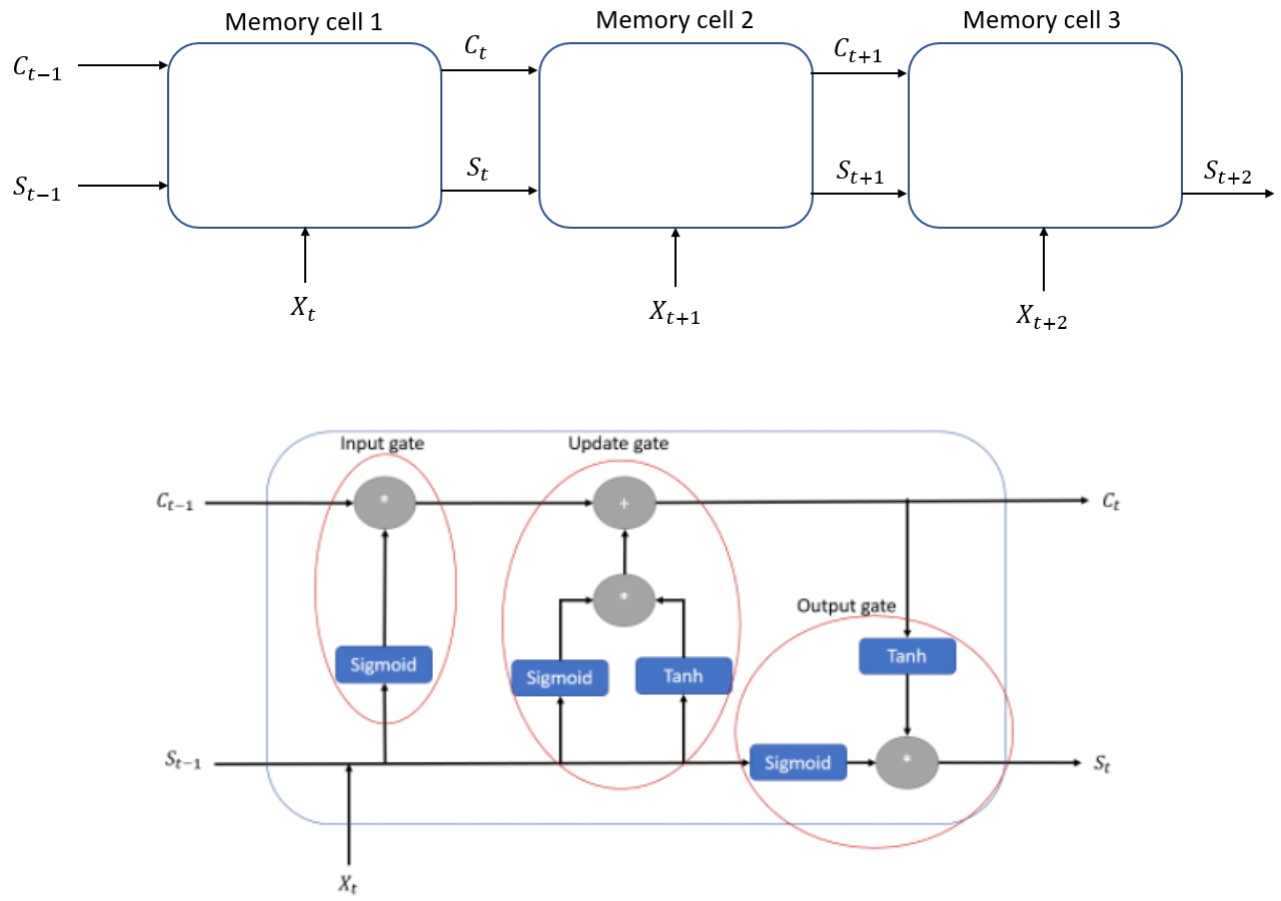

Figure 3.2: LSTM memory cell

Figure 3.2 depicts the structure of the memory cell in the LSTM network. The cell contains three gates, i.e., input, update, and output gates. Each of these gates consists of a Sigmoid activation function and a multiplication operation. Figure 3.3 depicts the behaviour of the *Sigmoid* function, as it can be observed the output value ranges between 0 and 1 and therefore it scales down the value of the cell state, e.g., if the output of the first gate is 0.5 then all the values in the cell state are halved.

The input gates is responsible for determining what information is discarded from the previous iteration. This is done by passing the value of the previous sequence and the current sequence though a *Sigmoid* action function, this is the value then used for the element wise multiplication. The mathematical model for this is:

$$f_t = Sigmoid(W_t * [S_{t-1}, X_t] + b_t \tag{3.1}$$

$$C`_{t-1} = f_t * C_{t-1} \tag{3.2}$$

where f_t represents the output of the *Sigmoid* activation function; $C`_{t_1}$: is the output of the input gate; W_f: is the weights for *Sigmoid* neuron f; and b_f: is the bias for *Sigmoid* neuron f.

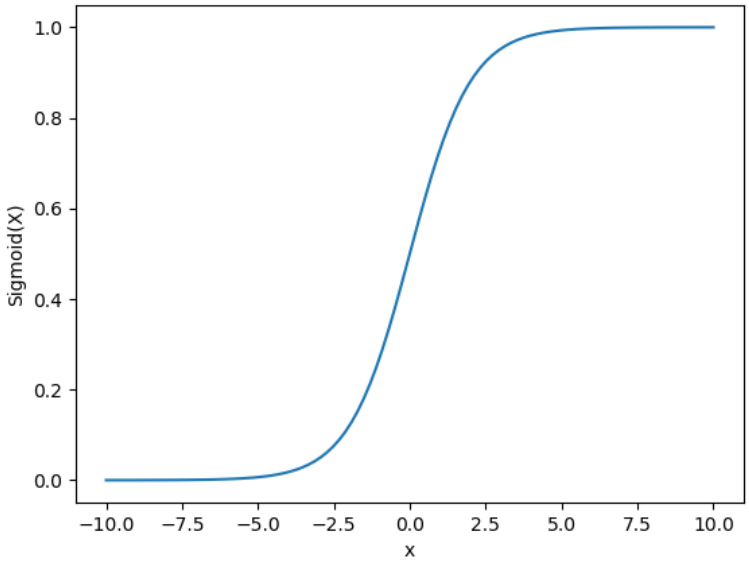

Figure 3.3: *Sigmoid* Function

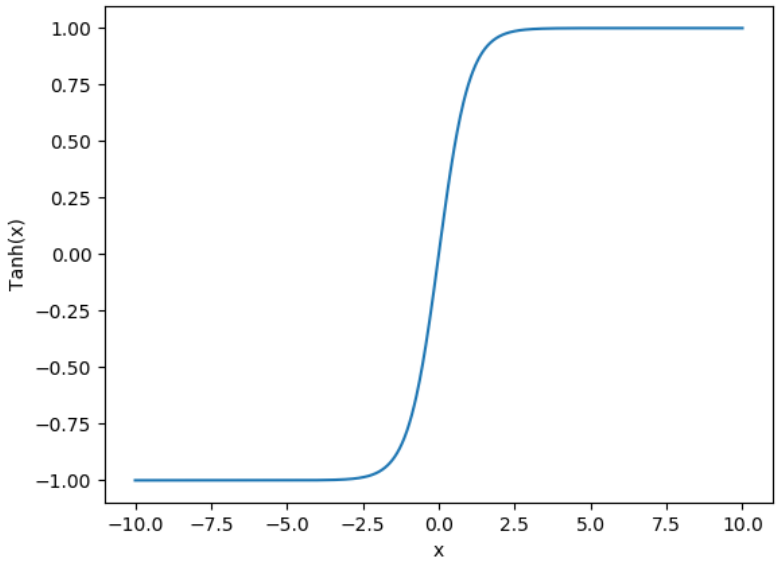

Figure 3.4: *Tanh* Function.

The update gate decides whether to remember or modify its previous value. This gate uses hyperbolic tangent (*Tanh*) and *Sigmoid* activation function. Figure 3.4 displays the behaviour of the *Tanh* function, the value of this function ranges between -1 and 1 and allows increases and decreases of values in the cell. The *Sigmoid* function modifies the output of the *Tanh* function with the use of the element wise multiplication. The resulting value in the cell state is then passed out of the current layer of LSTM as the current cell state value C_t and onto the next memory cell. This can be expressed as:

$$g_t = Sigmoid(W_g * [S_{t-1}, X_t] + b_g) \tag{3.3}$$
$$u_t = Tanh(W_u * [S_{t-1}, X_t] + b_u) \tag{3.4}$$
$$u_t = (u_t * g_t) \tag{3.5}$$
$$C_t = C_{t-1} + u_t \tag{3.6}$$

The output gate decides the prediction value for the next time step. This is done by performing an element wise multiplication with the output of the update gate which has been passed through an Tanh activation function, and a Sigmoid activation function using S_{t-1} and X_t as inputs. This can be formally written as:

$$h_t = Tanh(W_h * C_t + b_h) \tag{3.7}$$
$$k_t = Sigmoid(W_k * [S_{t-1}, X_t] + b_k) \tag{3.8}$$
$$S_t = (h_t * k_t) \tag{3.9}$$

4 Conclusions

In this deliverable we have describe a set of state-of-the-art methods for time series analysis and prediction. In particular, we focussed our attention in two statistical models, i.e., ARIMA and SARIMA and the LSTM deep learning architecture. We plan to evaluate these models in WP3-D2 *"Validation of the wind power prediction models"*. Our current preliminary evaluation indicates that our results are consistent with predictions available in the Smart Grid Dashboard system available at http://smartgriddashboard.eirgrid.com/.

We plan to use this models mainly in WP5 *Robust Scheduling of Charging Events* to identify suitable charging times for the electric bus fleet while maximising the use of renewable energies.

Bibliography

[1] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[2] C. Granger and C. Starica, "Nonstationarities in stock returns," *The Review of Economics and Statistics*, vol. 87, pp. 503–522, 02 2005.

[3] H. C. Ombao, J. A. Raz, R. von Sachs, and B. A. Malow, "Automatic statistical analysis of bivariate nonstationary time series," *Journal of the American Statistical Association*, vol. 96, no. 454, pp. 543–560, 2001.

[4] S. Davies and D. Bland, "Interestingness detection in sports audio broadcasts," in *2010 Ninth International Conference on Machine Learning and Applications*, 2010, pp. 643–648.

[5] M. Hippke, T. J. David, G. D. Mulders, and R. Heller, "Wōtan: Comprehensive time-series detrending in python," *The Astronomical Journal*, vol. 158, no. 4, p. 143, sep 2019.

[6] L. Deng and Y. Liu, *Deep Learning in Natural Language Processing*. Springer Publishing Company, Incorporated, 2018.

[7] Y. Xin, L. Kong, Z. Liu, Y. Chen, Y. Li, H. Zhu, M. Gao, H. Hou, and C. Wang, "Machine learning and deep learning methods for cybersecurity," *IEEE Access*, vol. 6, pp. 35 365–35 381, 2018.

[8] A. Roy, J. Sun, R. Mahoney, L. Alonzi, S. Adams, and P. Beling, "Deep learning detecting fraud in credit card transactions," in *2018 Systems and Information Engineering Design Symposium (SIEDS)*, 2018, pp. 129–134.

[9] D. Yu and L. Deng, *Automatic Speech Recognition: A Deep Learning Approach*. Springer Publishing Company, Incorporated, 2014.

[10] B. Lim and S. Zohren, "Time series forecasting with deep learning: A survey," 2020.

[11] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.

[12] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

SMARTeBuses

SMART electric Buses

January 29, 2021