Recurrent Neural Networks for Meteorological Time Series Imputation

Anibal Flores[1]
Grupo de Investigación en Ciencia de Datos, Universidad Nacional de Moquegua, Moquegua, Perú

Hugo Tito[2]
E.P. Ingeniería de Sistemas e Informática, Universidad Nacional de Moquegua, Moquegua, Perú

Deymor Centty[3]
E.P. Ingeniería Ambiental
Universidad Nacional de Moquegua
Moquegua, Perú

***Abstract*—The aim of the work presented in this paper is to analyze the effectiveness of recurrent neural networks in imputation processes of meteorological time series, for this six different models based on recurrent neural networks such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) are implemented and it is experimented with hourly meteorological time series such as temperature, wind direction and wind velocity. The implemented models have architectures of 2, 3 and 4 sequential layers and their results are compared with each other, as well as with other imputation techniques for univariate time series mainly based on moving averages. The results show that for temperature time series on average the recurrent neural network achieve better results than the imputation techniques based on moving averages; in the case of wind direction time series, on average only one model based on RNN manages to exceed the models based on moving averages; and finally, for wind velocity time series on average, no RNN-based model manages to exceed the results achieved by moving averages-based models.**

***Keywords*—Recurrent neural network; long short-term memory; gated recurrent unit; univariate time series imputation**

I. Introduction

The imputation of time series is a very important activity within the stage of homogenization of data, it is typical of the processing of meteorological time series. This will allow a subsequent time series to be used in forecasting processes.

There are many reasons why NA values are found: values may not have been measured, values may be measured but lost or values may be measured but erroneously [1]. Missing values can cause problems, since complete data is usually needed for proper processing and analysis.

It's very known that the accuracy of the imputation techniques will allow better results in forecasting or prediction processes [2]. Thus, a good selection of the imputation technique presented a certain problem is very important.

There is not a very large number of imputation techniques for univariate time series, among them can be mentioned those based on moving averages such as: Simple Moving Average (SMA) [3], Linear Weighted Moving Average (LWMA) [3], Exponential Weighted Moving Average (EWMA) [3], Autoregressive Integrated Moving Average (ARIMA) [4] among others.

Nowadays, recurrent neural networks (RNN) such as Long Short-Term Memory (LSTM) [5] and Gated Recurrent Unit GRU) [6] have become the most commonly used in prediction or forecasting models today for the accuracy of the results they offer in different fields such as machine translation, robot control, speech recognition, time series prediction among others. However, despite the benefits described, in the state of the art it is very difficult to find works that use recurrent neural networks for univariate time series imputation, which was one of the main motivations for the realization of the present study.

Thus, this paper presents the results of the implementation of six different models for hourly time series imputation based on recurrent neural networks. The analyzed time series correspond to temperature, wind direction and wind velocity and they were obtained from the Moquegua[1] meteorological station of SENAMHI in southern Peru. The gap-sizes analyzed correspond to short-gaps (1 to 2 NAs), medium-gaps (3 to 10 NAs) and large-gaps (11 to 30 NAs) [7]. Fig. 1 shows a graphical view of the 3-time series for 24 hours.

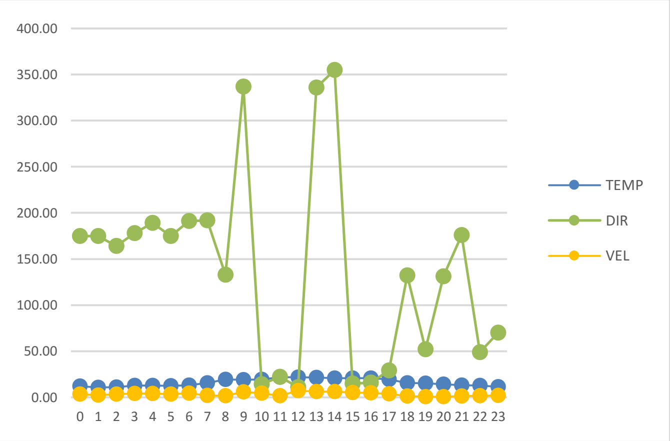


Fig. 1. Hourly Temperature, wind Direction and wind Velocity.

The content of the paper has been organized as follows: In the second section the related work is briefly described as proposed in this study. In the third section, the theoretical concepts and bases that will allow a better understanding of the content of the paper are described. In the fourth section, the models based on recurrent neural networks implemented in this

[1] SENAMHI Lat. 17°10'9" Lon. 70°55'54" Alt. 1450 masl. https://www.senamhi.gob.pe/?&p=estaciones

study are described and detailed. In the fifth section, the results achieved by the six different models in the time series are explained in detail. In the sixth section, the results achieved by the proposed models are compared and discussed with other models and techniques of the state of the art. In the seventh section, the arrived conclusions are explained according to study results. Finally, it describes the future work that can be done to improve the achieved results.

II. Related Work

This section shows a brief review of the works related to this study which are described below:

The first methods of imputation consisted of the use of parameters such as mean, median or mode [8], due to its simplicity there was a risk of inserting bias into the time series.

Another technique used later than the first was to use the last data observed before the missing one. This was called Last Observation Carried Forward (LOCF).

We also have the Hot-Deck [9] technique that consisted of randomly using existing data to replace the Not Available (NA) value.

Another group of techniques widely used are those based on moving averages including Simple Moving Average (SMA) [3], Linear Weighted Moving Average (LWMA) [3], Exponential Weighted Moving Average (EWMA) [3] which basically consisted of using the average of the data around the missing data assigning a weight according to its proximity to the NA value. This set of techniques are implemented in the present study to compare the results achieved by the imputation models based on recurrent neural networks.

An improved technique based on moving averages is what is known as Autoregressive Integrated Moving Averages (ARIMA) [4], which is a statistical technique that works with variations and regressions in a series of time to find patterns that will later serve to make predictions. This work also compares the results of ARIMA with those achieved by the imputation models based on recurrence neural networks.

Another technique used for imputation of time series is known as Local Average of Nearest Neighbors (LANN) [2], this technique is quite simple and consists only of using the prior and next values around an NA value, producing very good results at the level or better than those based on moving averages.

Two new imputation techniques inspired by Case Based Reasoning [10] are CBRi [11] and CBRm [8] which, like LANN, use only the prior and next values of an NA value, completing the missing values from the average of the historical data similar to the prior and next. The difference between the two is that CBRi is designed for short-gaps and CBRm for medium-gaps.

Another new technique is known as Average of Historical Vectors (AHV) [12] that uses only values similar to the prior value of the NA value to calculate the missing data. This technique is complemented by an adjustment algorithm (iNN) [12] and a smoothing algorithm (LANNf) [12].

III. Background

A. Time Series Imputation

The time series imputation refers to the process of calculating and completing the missing data or Not Available (NA) values in a series of time. For this it is very important to determine how the NA values originated, so they can be Missing Completely at Random (MCAR), Missing at Random (MAR) or Not Missing at Random (NMAR) [1]. It is also very important to determine the characteristics of the time series, so it can be very useful some characteristics such as: trend, seasonal or non-seasonal cycles, pulses, etc.

B. Recurrent Neural Networks (RNN)

An RNN is a type of neural network [13] that allows modeling different kind of problems such as time series for prediction.

The architecture of this neural network is very similar to the architecture of a Multilayer Perceptron (MLP) with the difference that an MLP allows connections between hidden units associated with a time delay. These connections allow the RNN to retain and remember information from the past [14], in this way it can find temporary correlations between facts that can be very separated in time. Fig. 2 shows the unfolded structure of an RNN.

Training an RNN is very difficult to implement [13] due to the vanishing and exploding gradients, This led to the implementation of a special type of RNN that is known as LSTM (Long Short-Term Memory) and that solves the above problems.

C. Long Short-Term Memory (LSTM)

As mentioned above, the LSTM networks were created to solve the problem of the vanishing and exploding gradients of the first recurrent neural networks. The LSTM networks work with special hidden units, whose objective is to remember input data for a long time [3], so LSTM networks are better than conventional RNN [5]. LSTM networks have several layers for each time step. Fig. 3 shows the LSTM architecture.

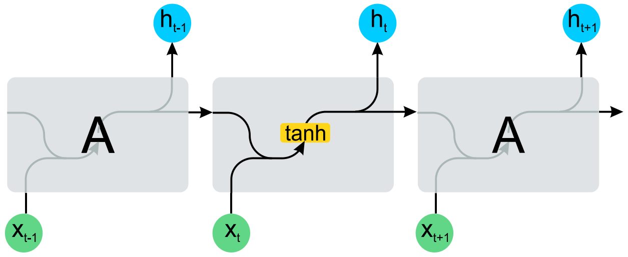


Fig. 2. Architecture of Recurrent Neural Network.

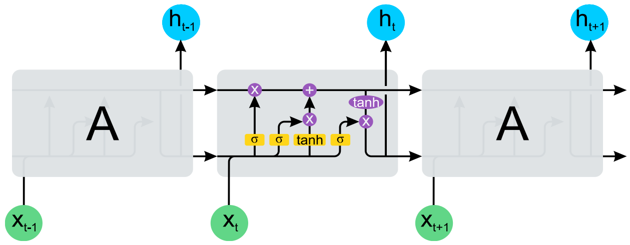


Fig. 3. Architecture of LSTM Network.

D. Gated Recurrent Unit (GRU)

GRUs are an activation mechanism in RNNs and were introduced by K. Cho et al. [6] in 2014. GRUs are a variation of LSTM networks, since both have a very similar architecture. However, unlike LSTM networks, GRUs have fewer parameters, since they lack an output gate. In many studies, LSTM networks have proven to be stronger than GRUs, since they can easily perform unlimited counting, while GRUs do not, so GRUs do not learn certain languages that LSTM can do [15]. Fig. 4 shows a very common architecture of GRU.

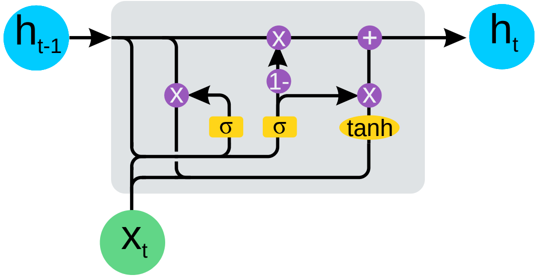


Fig. 4. Architecture of GRU.

According to Fig. 3 the following equations can be obtained and some parameters are described:

$$Z_{t=}\sigma_g(W_z x_t + U_z h_{t-1} + b_z) \qquad (1)$$

$$r_{t=}\sigma_g(W_r x_t + U_r h_{t-1} + b_r) \qquad (2)$$

$$h_{t=}(1 - z_t) \text{ o } h_{t-1} + z_t \text{ o } \sigma_h(W_h x_t + U_h(r_t \text{ o } h_{t-1}) + b_h \qquad (3)$$

Where:

x_t : input vector

h_t : output vector

z_t : updated gate vector

r_t : reset gate vector

W,*U* and *b* : matrix parameters and vector

σ_g : sigmoid function

σ_h : hyperbolic tangent

IV. Models for Experimentation

In the present study, six models based on recurrent neural networks were implemented, which are described below:

As can be seen in Table I of the six models implemented, three correspond to LSTM and three to GRU, the process followed to implement each of them is described below.

A. Time Series Selection

The hourly time series chosen for experimentation corresponds to temperatures, wind direction and wind velocity obtained from the SENAMHI repository. The data used for the training stage corresponds to 6000 hours from 2019-05-20 00:00:00 to 2020-01-24 23:00:00. The same period was used for all three-time series.

Likewise, the data between 2020-01-25 00:00:00 and 2020-01-31 23:00:00 was chosen as testing data (168 hours).

TABLE. I. RNN Models

Model	Name	RNN	Number of layers
1	LSTM LSTM	LSTM	2
2	LSTM LSTM LSTM	LSTM	3
3	LSTM LSTM LSTM LSTM	LSTM	4
4	GRU GRU	GRU	2
5	GRU GRU GRU	GRU	3
6	GRU GRU GRU GRU	GRU	4

B. Inserting NA Values

NA Values were inserted in the three testing time series according to what is shown in Table II.

C. Implementation of Models

Once the time series and training data were selected, the first model was implemented, as shown in Fig. 5.

This model was trained with the data of the three-time series predicting 168 values for each time series.

Next, the remaining five models were implemented, predicting 168 values for each time series in every model. Fig. 6, Fig. 7, Fig. 8, Fig. 9 and Fig. 10 show the architecture of these models.

D. Evaluating Predictions

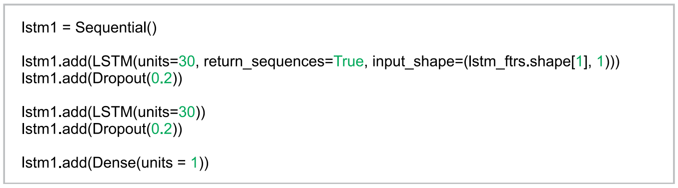
The results of six models are evaluated through Root Mean Squared Error (RMSE) according equation (4):

$$RMSE = \sqrt{\frac{\sum_{i=0}^{n-1}(Pi-Ri)^2}{n}} \qquad (4)$$

The results achieved are described in the next section.

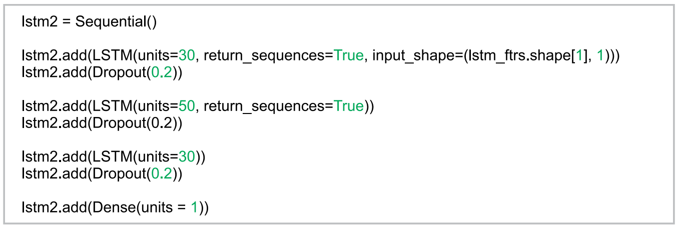
TABLE. II. Number of NA Values

	Temperature	Wind Direction	Wind Velocity
Short-Gaps	76	76	76
Medium-Gaps	124	124	124
Large-Gaps	155	155	155



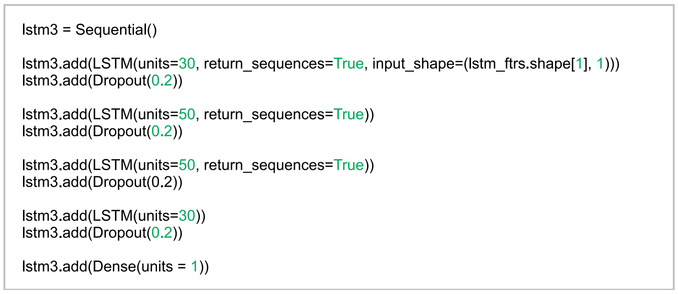
```
lstm1 = Sequential()

lstm1.add(LSTM(units=30, return_sequences=True, input_shape=(lstm_ftrs.shape[1], 1)))
lstm1.add(Dropout(0.2))

lstm1.add(LSTM(units=30))
lstm1.add(Dropout(0.2))

lstm1.add(Dense(units = 1))
```

Fig. 5. Architecture for First LSTM Model in Python.



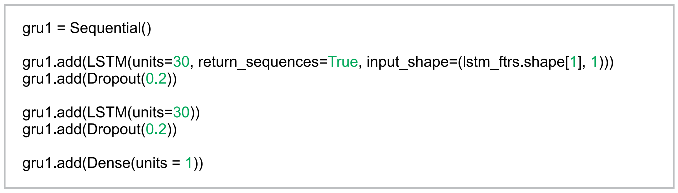
```
lstm2 = Sequential()

lstm2.add(LSTM(units=30, return_sequences=True, input_shape=(lstm_ftrs.shape[1], 1)))
lstm2.add(Dropout(0.2))

lstm2.add(LSTM(units=50, return_sequences=True))
lstm2.add(Dropout(0.2))

lstm2.add(LSTM(units=30))
lstm2.add(Dropout(0.2))

lstm2.add(Dense(units = 1))
```

Fig. 6. Architecture for Second LSTM Model in Python.



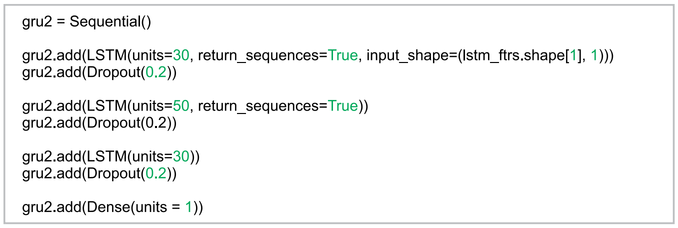
```
lstm3 = Sequential()

lstm3.add(LSTM(units=30, return_sequences=True, input_shape=(lstm_ftrs.shape[1], 1)))
lstm3.add(Dropout(0.2))

lstm3.add(LSTM(units=50, return_sequences=True))
lstm3.add(Dropout(0.2))

lstm3.add(LSTM(units=50, return_sequences=True))
lstm3.add(Dropout(0.2))

lstm3.add(LSTM(units=30))
lstm3.add(Dropout(0.2))

lstm3.add(Dense(units = 1))
```

Fig. 7. Architecture for Third LSTM Model in Python.



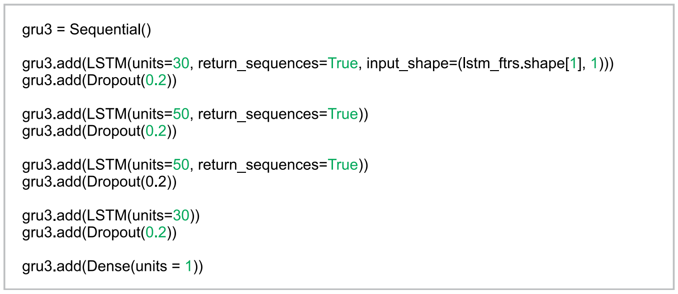
```
gru1 = Sequential()

gru1.add(LSTM(units=30, return_sequences=True, input_shape=(lstm_ftrs.shape[1], 1)))
gru1.add(Dropout(0.2))

gru1.add(LSTM(units=30))
gru1.add(Dropout(0.2))

gru1.add(Dense(units = 1))
```

Fig. 8. Architecture for First GRU Model in Python.



```
gru2 = Sequential()

gru2.add(LSTM(units=30, return_sequences=True, input_shape=(lstm_ftrs.shape[1], 1)))
gru2.add(Dropout(0.2))

gru2.add(LSTM(units=50, return_sequences=True))
gru2.add(Dropout(0.2))

gru2.add(LSTM(units=30))
gru2.add(Dropout(0.2))

gru2.add(Dense(units = 1))
```

Fig. 9. Architecture for Second GRU Model in Python.



```
gru3 = Sequential()

gru3.add(LSTM(units=30, return_sequences=True, input_shape=(lstm_ftrs.shape[1], 1)))
gru3.add(Dropout(0.2))

gru3.add(LSTM(units=50, return_sequences=True))
gru3.add(Dropout(0.2))

gru3.add(LSTM(units=50, return_sequences=True))
gru3.add(Dropout(0.2))

gru3.add(LSTM(units=30))
gru3.add(Dropout(0.2))

gru3.add(Dense(units = 1))
```

Fig. 10. Architecture for Third GRU Model in Python.

V. Results

This section shows the results achieved. Table III shows the corresponding RMSE values for imputation in hourly temperature time series.

According to what is shown in Table III and in Fig. 11 for the imputation process in the temperature time series on an average, the best model is LSTM LSTM LSTM (RMSE 0.5565), this model was also the one that produced the best results for all the gap-sizes.

Likewise, it can be seen that the GRU models were in second, third and fourth place, with the best GRU model being the GRU GRU (RMSE 0.5898) on average. So for this type of time series, the most recommended models would be the LSTM LSTM LSTM and the GRU GRU.

According to what is shown in Table IV and in Fig. 12 for the imputation process of wind direction time series, on average the best model was LSTM LSTM LSTM LSTM and this model was the best for each gap-size.

Likewise, it can be seen that similar to the temperature time series, the 4-layer LSTM model is the only one that managed to outperform the three GRU models and it is shown that the GRU models present more homogeneous results, while the LSTM models present results more heterogeneous that is, there is a greater dispersion among them.

According to Table V and Fig. 13 on average, the best model for imputation of the wind velocity time series was LSTM LSTM LSTM LSTM as well as for each gap-size.

TABLE. III. Temperature Time Series Results

Technique	RMSE			
	Short-Gaps	Medium-Gaps	Large-Gaps	Avg.
LSTM LSTM	0.7080	0.6934	0.6905	0.6973
LSTM LSTM LSTM	**0.5696**	**0.5592**	**0.5407**	**0.5565**
LSTM LSTM LSTM LSTM	0.5830	0.6191	0.6177	0.6066
GRU GRU	0.5848	0.5906	0.5942	0.5898
GRU GRU GRU	0.6095	0.5920	0.5781	0.5932
GRU GRU GRU GRU	0.5894	0.5943	0.5961	0.5933

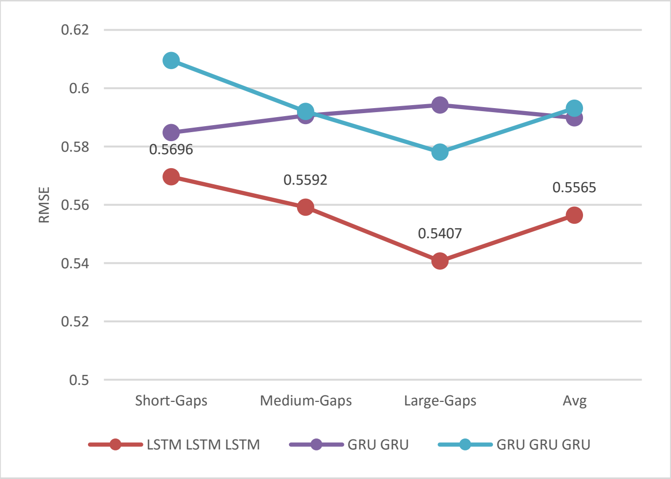


Fig. 11. Top 3 RNN Models for Temperature Time Series Imputation.

TABLE. IV. Wind Direction Time Series Results

Technique	RMSE			
	Short-Gaps	Medium-Gaps	Large-Gaps	Avg
LSTM LSTM	191.1086	190.4177	189.3457	190.2906
LSTM LSTM LSTM	186.5753	186.6744	185.5497	186.2664
LSTM LSTM LSTM LSTM	**150.4429**	**150.7271**	**151.1934**	**150.7878**
GRU GRU	161.7409	162.4947	161.2719	161.8358
GRU GRU GRU	163.6514	164.7290	164.7170	164.3658
GRU GRU GRU GRU	164.0155	164.2335	164.3080	164.1856

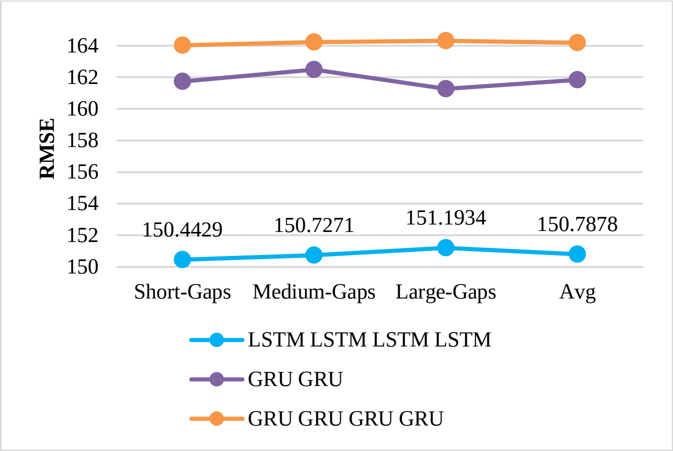


Fig. 12. Top 3 RNN Models for Wind Direction Time Series Imputation.

TABLE. V. WIND VELOCITY TIME SERIES RESULTS

Technique	RMSE			
	Short-Gaps	Medium-Gaps	Large-Gaps	Avg
LSTM LSTM	4.2175	4.2011	4.2451	4.2212
LSTM LSTM LSTM	2.7628	2.8512	2.8832	2.8324
LSTM LSTM LSTM LSTM	**2.3773**	**2.5066**	**2.5097**	**2.4645**
GRU GRU	3.5715	3.6008	3.6502	3.6075
GRU GRU GRU	3.6009	3.6276	3.6752	3.6345
GRU GRU GRU GRU	3.5501	3.5335	3.5599	3.5478

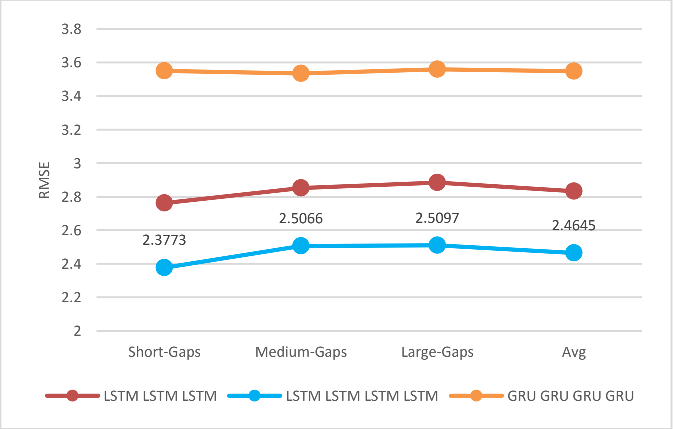


Fig. 13. Top 3 RNN Models for Wind Velocity Time Series Imputation

Unlike previous time series, for this type of time series two LSTM models present the best results: LSTM LSTM LSTM LSTM (RMSE 2.4645) and LSTM LSTM LSTM (RMSE 2.8324), while GRU models occupy the third, fourth and fifth place. Likewise, it is important to highlight that, like the previous time series, GRU models present more homogeneous results while LSTM models present more heterogeneous results.

VI. DISCUSSION

Next, the results achieved for the implemented models are compared with other univariate time series imputation techniques.

According to what is shown in Table VI, on average, the best technique for univariate time series imputation of temperatures is the recurrent neural network of 3 layers LSTM LSTM LSTM. However, performing an individual analysis for each gap-size, it is noted that this model is the best for medium-gaps (RMSE 0.5592) and large-gaps (RMSE 0.5407), but for short-gaps this is surpassed by the ARIMA-Kalman model (RMSE 0.4931).

According to Table VII, it is appreciated that on average the best technique for univariate time series imputation of wind directions is the recurrent neural network of 4 layers LSTM LSTM LSTM LSTM surpassing the different techniques based on moving averages in each gap-size.

According to Table VIII, it can be seen that on average the best imputation technique for wind velocity time series is the recurrent neural network of 4 layers LSTM LSTM LSTM LSTM. However, it is appreciated that this only exceeded the other techniques in large-gaps (RMSE 2.5097) while in Short-Gaps the best technique is Linear Weighted Moving Average (RMSE 1.1995) and for medium-gaps the best technique is Local Average of Nearest Neighbors (RMSE 1.6532).

TABLE. VI. TEMPERATURE IMPUTATION WITH ANOTHER TECHNIQUES

Technique	RMSE			
	Short-Gaps	Medium-Gaps	Large-Gaps	Avg
LSTM LSTM LSTM	0.5696	**0.5592**	**0.5407**	**0.5565**
GRU GRU	0.5848	0.5906	0.5942	0.5898
LANN	0.6343	2.0919	4.5210	2.4157
SMA	1.1208	1.8176	4.1515	2.3633
LWMA	0.8859	1.6868	3.9481	2.1736
EWMA	0.7262	1.6778	3.9348	2.1129
ARIMA-KALMAN	**0.4931**	1.13418	10.0067	3.8779

TABLE. VII. WIND DIRECTION IMPUTATION WITH ANOTHER TECHNIQUES

Technique	RMSE			
	Short-Gaps	Medium-Gaps	Large-Gaps	Avg
LSTM LSTM LSTM LSTM	**150.4429**	**150.7271**	**151.1934**	**150.7878**
GRU GRU	161.7409	162.4947	161.2719	161.8358
LANN	161.7666	169.3388	209.8176	180.3076
SMA	150.8792	153.2905	180.5009	161.5568
LWMA	150.4605	154.1056	184.2365	162.9342
EWMA	153.0640	161.0724	196.3272	170.1545
ARIMA-KALMAN	192.7941	233.8457	236.6198	221.0865

TABLE. VIII. WIND VELOCITY IMPUTATION WITH ANOTHER TECHNIQUES

Technique	RMSE			
	Short-Gaps	Medium-Gaps	Large-Gaps	Avg
LSTM LSTM LSTM LSTM	2.3773	2.5066	**2.5097**	**2.4645**
GRU GRU GRU GRU	3.5501	3.5335	3.5599	3.5478
LANN	1.2291	**1.6532**	4.0342	2.3055
SMA	1.4195	1.7944	3.6377	2.2838
LWMA	**1.1995**	1.6765	3.5758	2.1506
EWMA	1.2691	1.6852	3.5984	2.1842
ARIMA-KALMAN	1.2518	2.3963	3.3447	2.3309

As noted in the previous tables, the small difference between the RMSEs obtained by the models based on recurrent neural networks for short-gaps, medium-gaps and large-gaps should be highlighted. That is, the RMSE varies very little and it costs almost the same to impute 1 or 2 values than 30.

Likewise, it is also important to highlight that for short-gaps the imputation techniques of the state of the art offer very good results, while their performance is diminished in medium-gaps and much more in large-gaps, where RNN models offer the best results.

VII. CONCLUSIONS

The effectiveness of six models based on recurrent neural networks in nine case studies was analyzed, and in seven of them at least one model based on recurrent neural networks outperformed other imputation techniques of the state of the art, so we conclude that models based on recurrent neural networks are highly recommended to be implemented for univariate time series imputation especially for medium and large gap-sizes.

The results achieved show that not all models achieve optimal results, so it is important to implement not only one model but several in such a way that the most appropriate model can be chosen for the problem to solve.

In the three time series analyzed, the LSTM-based models show greater heterogeneity in their results compared to GRU-based models whose results are more homogeneous.

VIII. FUTURE WORK

In the present work it was experimented with models based on recurrent neural networks, differentiating them only by the number of layers and the number of neurons in each layer, for future works it would be important to be able to implement hybrid models that contain both LSTM and GRU layers, since it has been seen in different works that hybrid models for certain time series produce better results than non-hybrid models. Likewise, it can be experimented with other parameters such as the number of epochs, the batch-size, the training data size, the optimizer, etc.

Likewise, the results achieved by the RNN-based models for the wind direction and wind velocity time series, despite exceeding the state-of-the-art techniques, are not optimal (they have a high RMSE) so they could be improved by increasing the size of the training data or adding more variables to the model.

REFERENCES

[1] S. Moritz, T. Sardá, T. Bartz-Beielstein, M. Zaefferer and J. Stork, “Comparison of different methods for univariate time series imputation in R,” arxiv.org, 2015.

[2] A. Flores, H. Tito and C. Silva, “Local Average of Nearest Neighbors: Univariate Time Series Imputation,” International Journal of Advanced Computer Science and Applications, vol. 10, nº 8, pp. 45-50, 2019.

[3] S. Moritz, T. Bartz-Beielstein, “imputeTS: Time Series Missing Value Imputation in R,” The R Journal, vol. 9, nº 1, pp. 207-2018, 2017.

[4] R. Hyndman & G. Athanasopoulos, Forecasting: principles and practice, Melbourne, Australia: OTexts, 2018.

[5] Y. LeCun, Y. Bengio & G. Hinton, “Deep learning,” Nature, vol. 521, pp. 436-444, 2015.

[6] C. Kyunghyun, V. Bart, G. Caglar, B. Dzmitry, B. Fethi, S. Holger & B. Yoshua, “Learning phrase representations using RNN enconder-decoder for statistical machine traslation,” arxiv.org, pp. 1-15, 2014.

[7] A. Flores, H. Tito and C. Silva, “Local average of nearest neighbors: univariate time series imputation,” International Journal of Advanced Computer Science and Applications, vol. 10, nº 8, pp. 45-50, 2019.

[8] A. Flores, H. Tito & C. Silva, “CBRm: case based reasoning approach for imputation of medium gaps,” (IJACSA) International Journal of Advanced Computer Science and Applications, vol. 10, nº 9, pp. 376-382, 2019.

[9] A. Kowarick and M. Templ, “Imputation with R Package VIM,” Journal of Statistical Software, vol. 74, nº 7, 2016.

[10] M. Jahan Khan, H. Hayat and I. Awan, “Hybrid case-base maintenance approach for modeling large scale case-based reasoning systems,” Human-centric Computing and Information Sciences, vol. 9, nº 9, 2019.

[11] A. Flores, H. Tito & C. Silva, “CBRi: a case based reasoning-inspired approach for univariate time series imputation. In Press,” de 6th IEEE Latin American Conference on Computational Intelligence LA-CCI, Guayaquil, Ecuador, 2019.

[12] A. Flores, H. Tito & D. Centty, “Model for time series imputation based on average of historical vectors, fitting and smoothing,” (IJACSA) International Journal of Advanced Computer Science and Applications, vol. 10, nº 10, pp. 346-352, 2019.

[13] R. Pascanu, T. Mikolov, Y. Bengio, “On the dificulty of training recurrent neural networks,” de 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013.

[14] M. Paco, C. López Del Alamo & R. Alfonte, “Forecasting of meteorological weather time series through a feature vector based on correlation,” de 18th International Conference Computer Analysis of Images and Patterns CAIP 2019, Salerno, Italy, 2019.

[15] W. Gail, G. Yoav & Y. Eran, “On the practical computational power of finite precision RNNs for language recognition,” arxiv.org, pp. 1-9, 2018.