Model for Time Series Imputation based on Average of Historical Vectors, Fitting and Smoothing

Anibal Flores[1], 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*—This paper presents a novel model for univariate time series imputation of meteorological data based on three algorithms: The first of them AHV (Average of Historical Vectors) estimates the set of NA values from historical vectors classified by seasonality; the second iNN (Interpolation to Nearest Neighbors) adjusts the curve predicted by AHV in such a way that it adequately fits to the prior and next value of the NAs gap; The third LANNf allows smoothing the curve interpolated by iNN in such a way that the accuracy of the predicted data can be improved. The results achieved by the model are very good, surpassing in several cases different algorithms with which it was compared.**

***Keywords*—Univariate time series imputation; average of historical vectors; interpolation to nearest neighbors**

I. Introduction

The prediction of climate change and similar events requires increasingly precise predictive models, currently the most accurate prediction models require large amounts of data, however, in the field of meteorology, most historical time series present missing values or NA values for multiple reasons, and this means that a large amount of data cannot be used in prediction processes.

This paper presents a new model for imputation of missing data in meteorological time series. Missing data or NA values in weather series are presented in different sizes, small-gaps, medium-gaps and big-gaps [1]; accuracy in completing this data is very important to carry out successful forecasting or prediction processes.

The proposal model is based on three algorithms: AHV, iNN and LANNf which are briefly described below:

AHV approach is inspired by CBRi and CBRm that to complete a set of NA values use the prior value and the next value to the block of NA values and based on this, all historical values that are between these two values are searched and averaged.

However, implementing CBRi or CBRm for big-gaps is a bit complicated, since it is difficult to find intermediate data between 11 and 30 cosecutive NAs in a historical time series, and using the similarity could introduce bias in the synthetic data. That is why obtaining historical vectors considering just the prior value to the block of NAs was chosen.

Average of Historical Vectors (AHV) is a simple algorithm that uses historical vectors to calculate an average vector, historical vectors are classified into two groups and in each group they are identified by a key value that corresponds to their prior value. In addition, in the first group we find the historical vectors for the fall and winter stations and in a second group we find the historical vectors for the spring and summer stations, taking into account that the first group temperatures tend to fall and in the second group temperatures tend to rise.

In general, the algorithms for estimating missing values or NA values perform a horizontal analysis of time series considering the values before or after a group of NA values, such as LANN [1], LANN++ [1], SMA [2][3], LWMA [2][3], EWMA [2][3], ARIMA [3], etc. In CBRi "in press" [4] and CBRm [5] for NA calculation it is proposed to use only two horizontal values prior and next, and the rest of the data is vertical. Fig. 1 shows the difference between traditional approaches (horizontals) and new approaches (verticals). In this work, AHV uses a vertical approach using one of the most traditional and basic imputation techniques, the mean.

Interpolation to Nearest Neighbors (iNN) is an interpolation or fitting algorithm that allows adjusting a predicted curve to two values. In an imputation process, these two values would correspond to the prior and next values in a time series with NA values. See Fig. 2.

The adjustment problem arises in AHV predictions since in some cases no historical vectors are found for the prior value of a gap of NAs and vectors of a key value or other prior similar to the current prior are used.

Once the curve of imputed values is adjusted, it is important to carry out smoothing, since the imputed and interpolated data are usually not. It has been observed in Deep Learning algorithms such as Long Short-Term Memory (LSTM) [6], Gate Recurrent Unit [7], and others such as Prophet [8] that the estimated or predicted values are very smoothed, hence the present work apply smoothing to the predicted and interpolated vector in such a way that the accuracy of the model is improved. LANNf is used for this process which is inspired by the LANN [1] imputation algorithm.

For the experiment in this work, medium-gaps and big-gaps are considered. For the first case the performance of known imputation techniques such as SMA [9], LWMA [10], EWMA [2] [11], Kalman ARIMA, etc is analyzed. For the second case, very known prediction techniques such as ARIMA, LSTM, GRU and Prophet are analyzed.

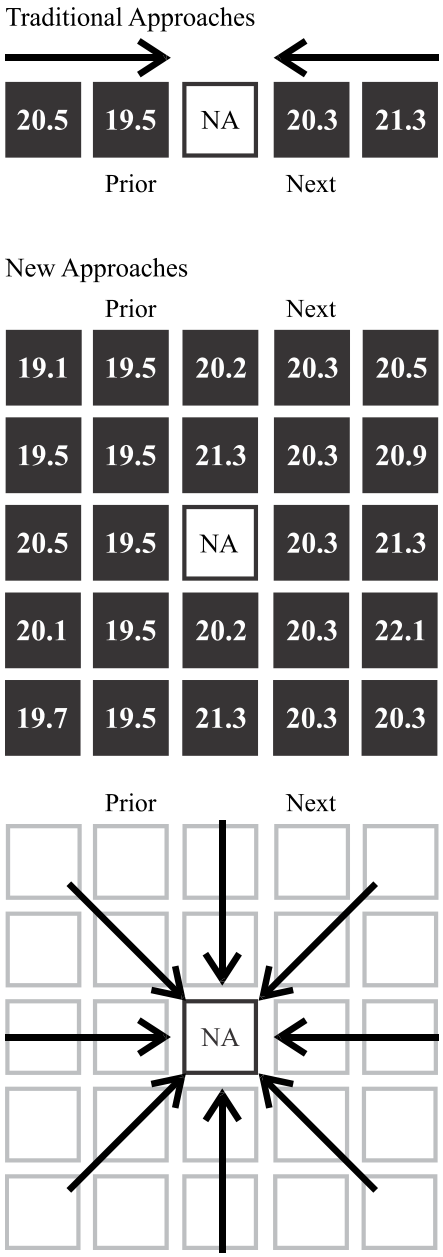

Fig. 1. Horizontal vs Vertical Approaches.

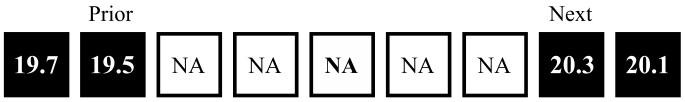

Fig. 2. Prior and Next Values.

This paper has been organized as follows: a summary of related work on univariate time series imputation is shown in the second section. In the third section some concepts are developed that will allow a better understanding of the proposal made at work. The fourth section describes the proposal model and each of its elements and algorithms implemented. The fifth section shows the results achieved and a comparison with several techniques. In the sixth section, the conclusions reached at the end of the present work are shown, finally some weaknesses of the proposed model are described and that could be overcome in future works.

II. Related Work

This section shows the reviewing results of different techniques or algorithms for univariate time series imputation which is detailed next.

Traditionally, quite simple techniques have been used for imputation of time series; these include the mean, median and mode, which currently are not recommended due to the risk of introducing bias to the time series.

Last Observed Carried Forward (LOCF) [12] is also a fairly simple technique, which involves replacing an NA value with the last observed value of the time series [13].

Baseline Observation Carried Forward (BOCF) [14] is similar to the LOCF; it replaces NA values with the non-missing baseline observation of the time series.

Hot-deck [15] [16], is an algorithm that replaces an NA value with an existing value of time series randomly. For example, if time series have ten values with an NA value, hot-deck, randomly select a value from the existing nine values and replace the NA value. For comparative analysis in this work VIM R package is used to implement hot-deck imputation.

Another set of imputation techniques that have been used frequently are those based on moving averages [3], among them Simple Moving Average (SMA) [3] [9], Linear Weighted Moving Average (LWMA) [3] [9] and Exponential Weighted Moving Average (EWMA) [3] [9] [11]. All of them use a parameter k that establishes the number of elements to calculate the average that replaces the NA value. In the case of SMA, the average is calculated considering only the elements established in parameter k without assigning any weight. In the case of LWMA, a linear weight is assigned to each element that will be used to calculate the average. And in the case of EWMA, the weight is assigned exponentially to each element used to calculate the average. For comparative analysis, moving average based algorithms are implemented using the imputeTS package of R language.

Kalman filter [17], also known as Linear Quadratic Estimation (LQE), is an algorithm that uses a series of measurements observed over time, which contains statistical noise and other inaccuracies, and produces estimates of unknown variables that tend to be more accurate than those based on a single measurement. Autoregressive Integrated Moving Average (ARIMA) [18] [19]integrated with Kalman filter produces good results in regression processes. Also, impute TS package of R language implements Kalman ARIMA imputation with a special setting called auto.arima [9] that produces optimal results.

LANN and LANN+ [1], they are two fairly simple algorithms based on moving averages that produce good results in the imputation of short-gaps (1 or 2 consecutive NAs). In this work, these are just compared and evaluated for the study cases corresponding to medium-gaps (3 to 10 consecutive NAs).

CBRi "in press" [4] and CBRm [5] are algorithms inspired by Case Based Reasoning and instead of taking advantage of the horizontal characteristics of a time series, these exploit the historical vertical values between the prior and next value in a block of NA values. Both are analyzed and compared in médium-gaps study cases.

For big-gaps, prediction algorithms that use large amounts of historical data are usually used, in this case algorithms such as ARIMA [3], PROPHET [8], LSTM [6] [20] and GRU [7] [21] will be implemented for analysis. Python language with keras and tensorflow libraries is used.

III. Theoretical Background

A. Time Series

A time series is the result of observing the values of an X variable over time. For example: the minimum daily temperature of the city of Lima, The total monthly sales of a given product, the number of visits per hour of a website, etc.

A common use of time series is its analysis for prediction and forecasting. Time series are studied in different areas such as signal processing, econometrics, statistics, biology, etc. Some features or characteristics of time series are: trends, cycles of seasonality and non-seasonality, pulses and steps, and outliers.

B. Missing Data

Depending on what causes missing data, the gaps will have a certain distribution. Understanding this distribution may be helpful in two ways [2]. First, this knowledge can be used to select the most appropriate imputation algorithm to complete the NA values. Second, this knowledge can help implement an imputation model with a set of training data and a set of test data to determine the RMSE to replace known NA values of the same time series; once the model is evaluated, it must be decided whether it is suitable for the imputation process of unknown NA values.

Missing data is classified into three categories: Missing Completely at Random (MCAR), Missing at Random (MAR) and Not Missing at Random (NMAR). The process of completing NA-gaps in time series is sometimes complicated, since the underlying mechanisms are unknown [2].

C. Univariate Time Series

This term refers to a time series that consists of single observations recorded sequentially over successive time periods. Although a univariate time series is usually considered as one column of observations, time is in fact an implicit variable [2]. Traditional techniques such as SMA, LWMA, EWMA, ARIMA and others usually just work with time series values, instead another forecasting techniques such as Prophet, LSTM, GRU in addition to the time series values they use the recording date, this undoubtedly makes their accuracy much better than moving average techniques.

IV. Proposal Model

The proposal model has four modules: Time Series, NA Calculation, Fitting - Smoothing, and Testing. Fig. 3 shows a graphical view of the proposal model and every module of this is described below.

A. Time Series

This module contains several functions that allow the pre-processing of the time series from which the historical vectors will be implemented. For the present study, the same time series was chosen as in [1], "in press" [4] and [5] so that the comparison of results is more appropriate.

Also, this module contains a getVectors() function which allows to extract historical vectors from the chosen time series.

Two vectors were considered for the vector base: The first Q1, contains time series extracted from March 23 to September 22 (Fall and Winter seasons in Peru). The second Q2, contains time series extracted from September 23 to March 22 (Spring and Summer seasons in Peru). Table I shows the algorithms used in Time Series Module.

TABLE. I. Algorithms for Retrieving Historical Vectors

```
function initTemp()
{   t=15.0;
    for(i=0;i<=200;i++)
    {   temv.push(parseFloat(t.toFixed(1)));
        t+=0.1;
    }
}
function initBase(ts,idx)
{   Q1=new Array(200);
    Q2=new Array(200);
    getVectors(ts,idx);
}
function getVectors(ts,idx)
{   nQ=Q1.length;
    nts=ts.length;
    total=30;
    for(i=0;i<nQ;i++)
    {   value=parseFloat(temv[i]);
        cad1="";//fall-winter
        cad2="";//spring-summer
        finj=nts-total;
        for(j=0;j<finj;j++)
        {   if(value==parseFloat(ts[j]))
            {   ini=j+2;
                fin=j+total;
                if(idx[j]<184)
                {   cad1+=ts[j+1];
                    for(k=ini;k<=fin;k++)
                        cad1+="*"+ts[k];
                    cad1+="/";
                }
                else
                {   cad2+=ts[j+1];
                    for(k=ini;k<=fin;k++)
                        cad2+="*"+ts[k];
                    cad2+="/";
                }
            }
        }
        Q1[i]=cad1;
        Q2[i]=cad2;
    }
}
```

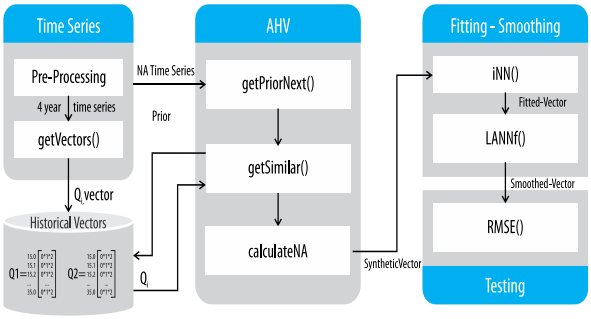

Fig. 3. Proposal Model.

B. Average of Historical Vectors (AHV)

This module was implemented through a function called AHV that uses several functions that will allow to calculate the NA values, among these functions we have getPriorNext (), getSimilar() and calculateNA(), which are described below:

getPriorNext() is a function that allows to calculate the prior and next values from the time series with NA values.

getSimilar () is a function that allows to obtain the historical vectors of 30 days having the prior value as a key value.

The data returned by the getSimilar() function is converted in vectors and then into a matrix M whose columns will be averaged to obtain the average vector with the estimated NA values. Equation (1) is used to calculate NA values.

$$NAs = \sum_{j=0}^{m-1}(\sum_{i=0}^{n-1} Mij)/n \quad (1)$$

TABLE. II. AHV ALGORITHM

```
function AHV(prior,ix)
{
    totalf=30;
    posf=ts.length-1;
    i=0;
    index=ix+1;
    avector=new Array();
    while(i<totalf)
    {   data=getSimilar(prior,ix);
        dat=data.split("/");
        ndat=dat.length-1;
        M=new Array();
        for(j=0;j<ndat;j++)
        {    da=dat[j].split("*");
             M.push(da);
        }
        nda=20;
        ix+=nda;
        if(ix>=365)
             ix=0;
        for(jj=0;jj<nda;jj++)
        {    s=0.0;
             for(ii=0;ii<ndat;ii++)
                  s+=parseFloat(M[ii][jj]);
             mean=s/ndat;
             avector.push(mean.toFixed(4));
        }
        current=mean.toFixed(1);
        i+=nda;
        posf=ts.length-1;
    }
    return avector;
}
```

The code corresponding to the AHV algorithm is shown in Table II.

C. Fitting - Smoothing

This module allows improving the accuracy of the estimated vector by AHV algorithm using two algorithms, the first is iNN (interpolation to Nearest Neighbors) and the second is LANNf, which are described below.

- Interpolation to Nearest Neighbors (iNN). It is an algorithm that approximates an estimated vector of NA values towards the prior and next values of a block of NA values.

This approach consists of:

1) Calculate the difference *d* between prior and next values, through equation (2).

$$d = (prior - next) \quad (2)$$

2) Determine the *t* factor from equation (3).

$$t = d/nna \quad (3)$$

Where *nna* is the gap-size. The *t* factor is used to calculate the values of the first NA and Last NA. See Fig. 4.

The *t* factor is subtracted or added according to the value of *d*, which determines the trend of the curve.

3) Once the first fitted NA and last fitted NA values are calculated, the differences between these values and those estimated in the NA Calculation block are determined using equation (4) and equation (5).

$$d1 = first\_NA - first\_fitted\_NA \quad (4)$$

$$d2 = last\_NA - last\_fitted\_NA \quad (5)$$

4) Based on the differences *d1* and *d2*, the *k* factor is calculated using equation (6).

$$k = |\frac{d1-d2}{nna-1}| \quad (6)$$

The k factor is used to adjust the non-adjusted NA values. An adder *s* is initialized on *d1* and each non-adjusted NA element is iterated. If *d1<d2* the adder *s* increases in *k*, otherwise if *d1>d2* the adder *s* decreases in *k*.

Fig. 5 shows an example of an estimated 30-day curve fitted with iNN.

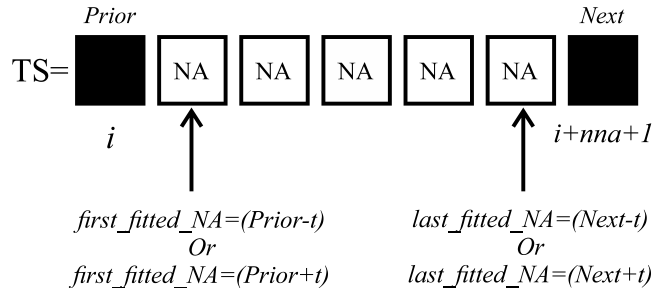

Fig. 4. First and Last NA Re-Calculation.

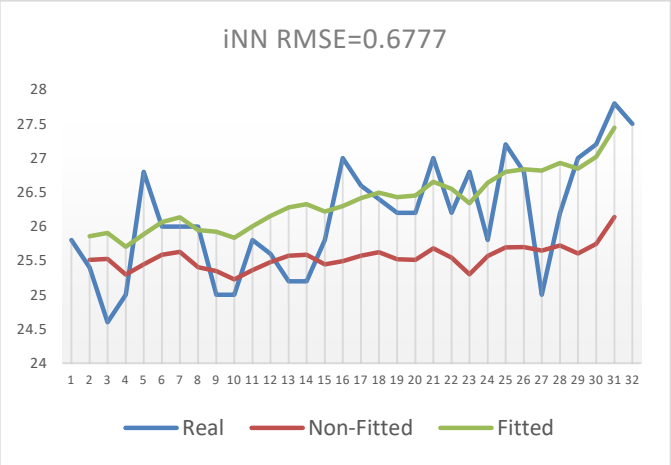

Fig. 5. Comparison of Non-Fitted Curve vs Fitted Curve.

The iNN algorithm is shown in Table III, which it receives as inputs:

- ts the time series to interpolate or adjust
- prr the prior value
- nxt the next value
- nna the number of NA values

TABLE. III. iNN ALGORITHM

```
function iNN(ts,prr,nxt,nna)
{   first=0;
    last=nna-1;
    d=prr-next;
    t=Math.abs(d/nna);
    if(d>0)
    {   prr1=prr-t;
        nxt1=nxt+t;
    }
    else
    {   prr1=prr+t;
        nxt1=nxt-t;
    }
    d1=ts[first]-prr1;
    d2=ts[last]-nxt1;
    dd=d1-d2;
    k=Math.abs(dd/(nna-1));
    predi=new Array();
    predi.push(prr1);
    s=d1;
    for(z=1;z<last;z++)
    {
        if(d1>d2)
            s-=k;
        else
            s+=k;
        fllw=parseFloat(ts[z])-s;
        predi.push(fllw);
    }
    predi.push(nxt1);
    return predi;
}
```

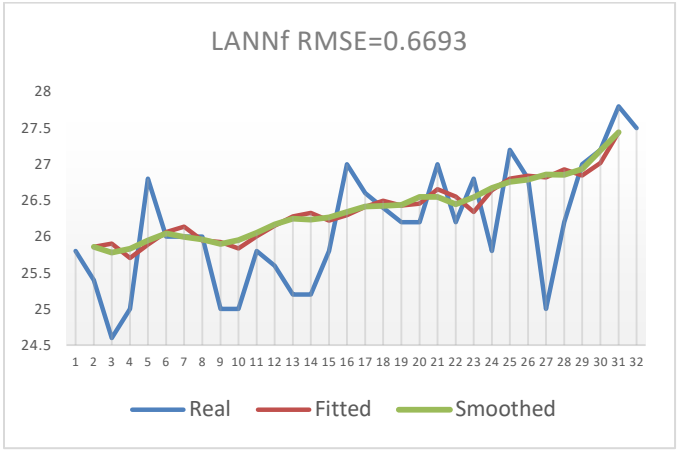

Fig. 6. Comparison of Fitted Curve vs Smoothed Curve.

- Local Average of Nearest Neighbors Filter (LANNf). It is an algorithm inspired by LANN [1] that allows smoothing a curve, for each estimate it uses three values of the vector to be smoothed, recalculating the intermediate or second value with the average of the first and third. Fig. 6. Shows a comparison between a fitted curve versus a smoothed curve using LANNf.

D. Testing

This module implements a RMSE function that calculates the Root Mean Squared Error to estimate the performance and accuracy of the proposal model. RMSE is calculated with Equation (7).

$$RMSE = \sqrt{\frac{\sum_{i=0}^{n-1}(Pi-Ri)^2}{n}} \quad (7)$$

The LANNf algorithm is shown in Table IV.

TABLE. IV. LANNF ALGORITHM

```
function LANNf(ts)
{   nts=ts.length-1;
    i=1;
    while(i<nts)
    {   prr=parseFloat(ts[i-1]);
        nxt=parseFloat(ts[i+1]);
        ts[i]=((prr+nxt)/2).toFixed(4);
        i+=1;
    }
    return ts;
}
```

V. RESULTS AND DISCUSSION

In this section, the accuracy of the proposal model is compared with different techniques described in Related Work section; the comparative results show the performance of the proposal on medium and big gaps.

Table V and Fig. 7 show the results achieved by the proposal model compared to other well-known techniques. As it can be seen, the proposal model is always among the best ones.

TABLE. V. COMPARISON WITH OTHER UNIVARIATE IMPUTATION TECHNIQUES (90-DAYS) IN MEDIUM-GAPS

Technique	RMSE (NAs 80%)	RMSE (NAs 65.55%)	RMSE (NAs 54.44%)
AHV, iNN & LANNf	**0.7038**	**0.7251**	**0.8381**
CBRm	**0.6844**	0.8050	0.8968
CBRi	0.8086	0.8112	0.8905
LANN	0.8422	0.8198	0.9053
LANN+	0.8276	0.7339	0.8608
Hotdeck	1.4337	1.6323	1.4996
SMA (k=1)	0.8324	**0.7035**	**0.8403**
LWMA (k=4)	**0.7673**	**0.7083**	**0.8106**
EWMA (k=4)	0.7682	0.7456	0.8535
ARIMA Kalman	5.4275	6.7383	2.6836

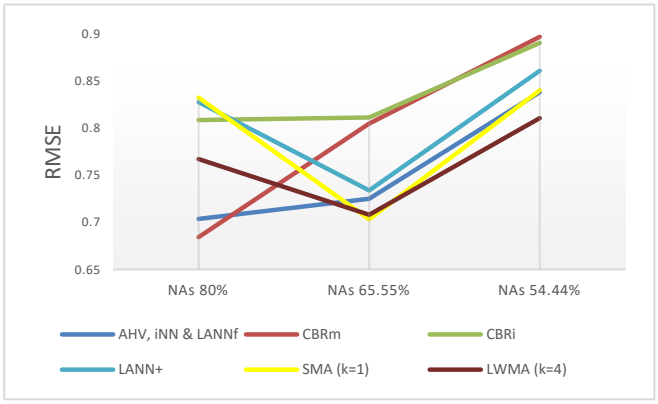

Fig. 7. Comparison with other Techniques (Medium-Gaps).

According to Table VI and Fig. 8, it can be seen that the proposal model, like the case of medium-gaps, for big-gaps is also among the best.

TABLE. VI. COMPARING WITH OTHER UNIVARIATE IMPUTATION TECHNIQUES IN BIG-GAPS

Technique	RMSE GAP-SIZE:11	RMSE GAP-SIZE: 21	RMSE GAP-SIZE: 30
AHV, iNN & LANNf	0.6175	0.6783	**0.6693**
ARIMA	0.6748	1.0424	1.4165
Prophet	**0.5991**	**0.6477**	0.7652
LSTM	0.6156	0.6820	0.7579
GRU	0.6749	0.6503	0.7262

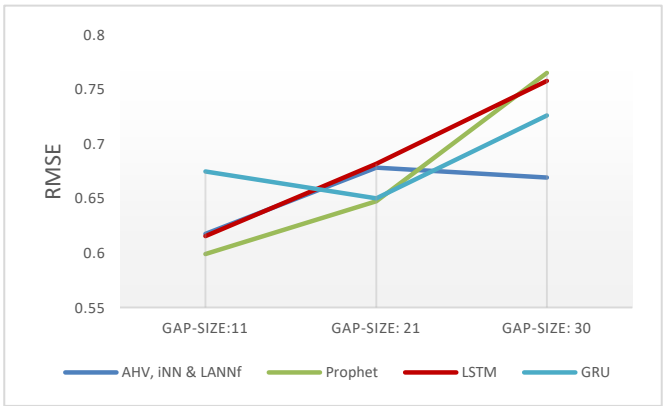

Fig. 8. Comparison with other Techniques (Big-Gaps).

VI. CONCLUSION

In imputation processes of máximum temperature time series with medium-gaps (from 3 to 10 consecutive NAs), of the three proposed problems and among 10 techniques, the proposal model was among the best: in one problem it was third and in the other two it was second.

For big-gaps imputation, in three different problems the proposed model was always among the three best, in two cases it was third and in one case it was the best.

Therefore, according to the results achieved, the proposed model is highly recommended for imputation processes of medium and big-gaps.

VII. FUTURE WORK

There are several improvements that can be implemented in this work; some of them are mentioned below:

AHV only uses historical data to estimate NAs vertically; it could be complemented with the traditional horizontal mode. In addition, the vertical imputation technique could be any of the known SMA, LWMA, EWMA, ARIMA, KALMAN, etc., since as the average is known, it is one of the most basic and risky techniques due to the bias that can be inserted in time series.

Also, instead of working with only two stations, the 4 known stations could be included for this type of time series.

REFERENCES

[1] Flores, H. Tito, 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.

[2] S. Moritz, A. Sardá, T. Bartz-Beielstein, M. Zaefferer, J. Stork, "Comparison of different Methods for Univariate Time Series Imputation in R," arxiv.org, 2015.

[3] S. Moritz, "Package ImputeTS," cran.r-project.org, 2019.

[4] A. Flores, H. Tito, C. Silva, "CBRi: A Case Based Reasoning-Inspired Approach for Univariate Time Series Imputation. In Press," de IEEE Latin American Conference on Computational Intelligence, Guayaquil, Ecuador, 2019.

[5] A. Flores, H. Tito, C. Silva, "CBRm: Case based Reasoning Approach for Imputation of Medium Gaps," International Journal of Advanced Computer Sciences and Applications, vol. 10, nº 9, pp. 376-382, 2019.

[6] S. Hochreiter, J. Schmidhuber, "Long short-term memory," Neural computation, vol. 9, nº 8, pp. 1735-1780, 1997.

[7] C. Kyunghyun, V. Bart, C. Caglar, B. Dzmitry, B. Fethi, S. Holger, B. Yoshua, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," arXiv, 2014.

[8] S. Taylor, B. Letham, "Forecasting at scale," PeerJ Preprints, 2017.

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

[10] S. Moritz, A.Sardá, T. Bartz-Beielstein, M. Zaeffer, J, Stork, "Comparison of different methods for univariate time series imputation in R," arxiv.org, 2015.

[11] E. Rantou, "Missing Data in Time Series and Imputation Methods," University of the Aegean, Samos, 2017.

[12] N. Bokde, M. Beck, F. Martinez, K. Kulat, "A novel imputation methodology for time series based on pattern sequence forecasting," Pattern Recognition Letters, 2018.

[13] A. Zeileis, G. Grothendieck, "zoo: S3 infrastructure for regular and irregular time series," Journal of Statistical Software, vol. 14, nº 6, 2005.

[14] K. Kaiser, O. Affuso, T, Beasley, D. Allison, "Getting carried away: A note showing baseline observation carried forward (BOCF) results can

be calculated from published complete-cases results," PMC US National Library of Medicine, 2012.

[15] A. Kowarick, M. Templ, "Imputation with the R package VIM," Journa of Statistical Software, vol. 74, nº 7, 2016.

[16] T. Aljuaid, S. Sasi, "Proper imputation techniques for missing values in data sets," de International Conference on Data Science and Engineering (ICDSE), Cochin, India, 2016.

[17] P. Zarchan, H. Musoff, Fundamentals of kalman filtering, American Institute of Aeronautics and astronautics, 2000.

[18] Z. Wang, Y. Lou, "Hydrological time series forecast model based on wavelet de-noising and ARIMA-LSTM," de IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Chengdu, China, 2019.

[19] Y. Du, "Application and analysis of forecasting stock price index based on combination of ARIMA model and BP neural network," de Chinese Control and Decision Conference (CCDC), Shenyang, China, 2018.

[20] S. Kumar, L. Hussain, S. Banarjee, M. Reza, "Energy load forecasting using deep learning approach-LSTM and GRU in spark cluster," de IEEE, Kolkata, India, 2018.

[21] J. Chung, C. Gulcere, K. Cho, Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," arxiv.org, 2014.