Abstract

Completeness of data is vital for decision making and forecasting of Building Management System (BMS) datasets. Missing data can result in biased decision making down the line. This study creates a guideline for imputing the gaps in BMS datasets by comparing four methods: KNN algorithm, RNN, Hot Deck and LOCF. The guideline will contain the best method per gap and data classification. The four methods are from various backgrounds and are tested on a real BMS and KNMI dataset. The focus of this paper is not to impute every cell as accurate as possible but to impute trends back into the missing data. The performance is evaluated with Variance Error (VE) instead of RMSE to indicate the ability to impute trends. From preliminary results, concluded that the best K for KNN would be 5 for the smallest gap size and 100 for the bigger gaps. The results also concluded that RNN architecture GRU was best used for this research.

Keywords: Building Management System time series data, Imputation, KNN, RNN, Hot Deck, trend

Introduction

Missing data is a common occurrence in time-series data, for this specific case causes include faulty sensors or errors in data storage. Missing data can cause downstream applications to malfunction and can thus have serious consequences. Missing data in Building Management Systems (BMS) can cause underperforming building services e.g., lower comfort of living or higher power usage, or in worst-case scenarios building breakdown as system control decisions are based on the collected data.

Imputation methods evaluated in this paper are selected from previous research that has been done into the imputation of time series data. The methods that are selected for evaluation are: Last Observation Carried Forward (LOCF), K-Nearest Neighbour algorithm (KNN), Recurrent Neural Network (RNN) and Hot Deck (HD).

HD has been outperformed by machine learning in the past as seen in (Sree Dhevi, 2014) [1] but it is applicable due to the number of similar units available for study. The time-series imputation performance of different types of RNN's has been studied before in Che et al. (2018) [2]. The study concluded that when a Gated Recurrent Units (GRU) architecture is properly set up *"it pulled significantly ahead of non-deep learning methods"* [2].

Pazhoohesh et al. (2019) [3] found that for datasets where 10 to 30 %of the data is missing KNN algorithm does great compared to eight other methods. Poloczek et al. 2014 [4] analysed the use of KNN regression and FFIL and found that both did well for the study, but that KNN regression dominated other methods.

There are limited studies to clarify how to deal with missing data in BMS datasets. Previous research has focused on lighting and occupancy [3] data or created a generic framework for imputing data from multiple sensors [5]. In the case of (Zhang,2020) it is advised that a more generic plug-n-play framework is to be further studied. This study will not build on the framework created by (Zhang,2020) but tries to give a guideline as to use the methods evaluated in this paper. The research focused on imputing trends rather than accurately imputing data in a single moment in time.

This paper aims to evaluate and compare the imputation performance of the following methods: KNN algorithm, LOCF, RNN and Hot Deck. The imputation performance has been evaluated by making use of various criteria to facilitate the choice of the most suitable method for each scenario.

The method section will contain a description of the datasets, description of the pipeline, imputation methods and the criteria used for evaluation. The result section will present the imputation results and a recommended action for each data classification and gap size.

Methodology

Dataset description

BMS datasets store sensor data such as fluid temperature, power usage, flow rate, operational mode, solar radiation, and outdoor temperature. methods two datasets have been used, twenty-five weather stations from the Royal Netherlands Meteorological Institute (KNMI) [8] and BMS data of hundred-twenty residential Net-Zero energy houses. The NZEB BMS time series dataset contains data from 2019 and is supposed to have five-minute interval data measurements (105096 rows). The KNMI dataset contains data from 2018 to 2020 and is measured at hourly intervals (17545 rows). The only change made to the datasets was converting the timestamps to Python Date Time objects.

Columns selected for imputation

To get a general impression of imputation performance on other sensors and efficiency of research seven columns are selected from the two datasets to evaluate the imputation performance. The selected features from the BMS dataset are power usage (power), CO2 level measurements (CO2), heat pump flow temperature (flow_temp) and operational mode (op_mode). The features selected from the KNMI datasets are solar radiation (global radiation), temperature (temperature) and relative atmospheric humidity (Relative atmospheric humidity).
The columns were selected for the classification of data and the KNMI columns were also selected for the strong correlation between the features.

Table 1.

Title: Columns selected for imputation

Description: Columns with the dataset of origin, device, unit of measurement and classification.

Column name	Dataset	Device	Unit of measurement	Classification
Temperature	KNMI	-	C (in 0.1c)	Interval
Global Radiation	KNMI	-	j per cm2	Ratio
Humidity	KNMI	-	%	Ratio
Flow_temp	BMS	Alklima Heat Pump	C	Interval
op_mode	BMS	Alklima Heat Pump	0-6 modes	Nominal
Power	BMS	Smartmeter	W	Ratio
C02	BMS	C02 Sensor	PPM	Ratio

Pipeline

A pipeline has been developed to evaluate the performance of imputation methods under the same reproducible conditions. The pipeline performed the following tasks: loading the data, creating gaps, imputing the artificial gaps, calculating imputation performance, and storing the evaluation results. The pipeline code and trained models can be found in the appendix.

Gap creation

To evaluate the performance of each imputation method artificial gaps are created in both datasets. The gaps come in different sizes to evaluate the performance of each imputation method the amount of sequential missing data. Gaps are created along the rules stated in the table below and are generated using a set random seed. The set random seed is also used to determine gap location and the size of the gap. The gap sizes and locations are the same for every feature and method tested.

Table 2.

Title: BMS artificial gap rules
Description: BMS gap sizes with minimum size, maximum size, and percentage of total missing data.

Nr.	Min_size	Max_size	% Of data
1	5 min	60 min	15
2	1 hour	6 hours	4
3	6 hours	24 hours	1.5
4	24 hours	72 hours	0.5
5	72 hours	168 hours	0.01

Table 3.

Title: KNMI artificial gap rules
Description: KNMI gap sizes with minimum size, maximum size, and percentage of total missing data.

Nr.	Min_size	Max_size	% Of data
1	1 hour	6 hours	15
2	6 hours	24 hours	5
3	24 hours	72 hours	1.5
4	72 hours	168 hours	0.005

Imputation methods

Four imputation methods are compared in this paper: Hot Deck, Recurrent Neural Network (RNN), Last Observation Carried Forward (LOCF) and K-Nearest Neighbour algorithm (KNN). The methods are selected from previous literature and aim to have a wide scope of imputation approach to facilitate each method's characterizations, advantages, and disadvantages.

KNN algorithm

KNN algorithm is a nonparametric imputation method that works by taking the average of a gap's K-number of neighbours. Treating every neighbouring value equally KNN would make it more vulnerable to outliers. To mitigate this KNN is set up to weigh the nearer neighbours of a gap heavier than further away values.

According to the research, in Pazhoohesh et al. (2019), KNN achieved the best result when selecting the K-number in proportion to the percentage of data missing. The K-values that were tested in this paper [3] are 1, 2, 4, 6, 8 and 10, with 1 or 2 being best at 10% and 4 the best at 30% missing. For this paper, the K-value has been redetermined because of the difference in the data used.

The K-values tested are: 1,5,10,15,20,100. The K-value selection was done by evaluating the results gotten from imputation using the Variance Error. From the results of the evaluation, it can be concluded that K=5 is best for the gap size 1 and K=100 for gap sizes 2 to 5.

Last Observation Carried Forward

Last Observation Carried Forward works by filling in the gap with the last valid before the gap observation forwards. LOCF can introduce substantial bias in datasets that do have high volatility in values [6]. Columns such as power usage will most likely suffer the most from this due to the unpredictability in data which is expected to worsen with larger gaps in the data. However, LOCF is still in common use to this day and has been compared before in time-series imputation performance [3-5].

Hot deck

Introduction to Hot Deck

Hot-deck imputation is a method for handling missing data in which each missing value from a recipient is replaced with an observed value from a similar unit (the donor). This method applies perfectly to this study since there are multiple units (different houses or different weather stations' data).

This method is a well-known method, but the theory behind the Hot Deck is not as well developed as that of other imputation methods, leaving researchers with limited guidance on how to apply it. The main challenge will be selecting donors.

In some versions, the donor is selected randomly from a set of potential donors, which is called the donor pool. In other, more deterministic, versions a single donor is identified, and values are imputed from that case usually, the "nearest neighbour" based on a dataset-dependent metric (i.e.: the mean when imputing temperature time series).

Implementing the donor selection

In theory

In the case of this research, the donor selection was based on pattern recognition. It works by taking an extract containing data before and after a series of missing values (a gap) found in the recipient. To find the best matching segment of data from a donor, the recipient's extract would then be compared to similarly-sized extracts from the same time period in a donor.

Using the difference in the mean of the donor's extracts and recipient's extract, the values from the donor's extracts can be shifted towards those of the recipient except when imputing categorical data. The sum of the absolute difference between the extracts can now be used to sort the comparisons: the smaller the sum, the better the pattern matches.

The operation can then be repeated throughout each donor of the donor pool, for each gap, to find the best possible match before finally importing data into the recipient.

In application

This donor selection method has been applied in two versions for this paper.

Whereas the first iteration had a focus on precision, using interpolation to have the most accurate value between two data measurements, for example. The second iteration focused on improving processing time, by vectorizing the search algorithm.

But the processing time improvements had a negligible cost in precision. Which was even more diminished by the ability to compare the recipient's extract to superior amounts of data from each donor for every gap (equivalent to a month plus the gap size).

Recurrent Neural Network

Recurrent Neural Networks have been proven to perform well when working with time-series data [7] and in [2] it pulled significantly ahead of non-deep learning methods. RNN's benefits from having an internal memory unlike other NNs this helps them to preserve context which will be useful with the imputation of BMS time-series data. The internal memory of the RNN architecture is useful for the purpose of imputing time-series data as the missing values will highly depend on the trend before and after a gap.

Two different architectures of RNNs were compared on performance in time series imputation: Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU). Both architectures use long-short term memory, but the key difference is that LSTM uses three gates; forget, input and output whilst GRU uses two; update and reset. Another difference between the two is that GRU exposes the entire memory including hidden layers whilst LSTM keeps that hidden.

The two architectures were compared using a genetic algorithm, random configurations are generated by changing the GRU or LSTM. Architectural parts that were randomized are the number (1 – 5) and size (2 – 100) of hidden layers, input sequence size (2 – 12) and the loss-function (MSE or Huber).

The best architecture configuration found during testing was a GRU RNN configured with 1 hidden layer of size 95, input sequence length 12 and the using the MSE loss function. The final GRU was configured as listed before in addition a fully connected layer was added. This was done to transform the GRU layer output into prediction. The GRU-based RNN was trained for every feature that was to be imputed.

The implementation of the current GRU-based model has two major limitations. It only imputes one value at a time based on the X-number of preceding values. This means that with a sufficiently large gap it will use its own values to impute further. Using previously imputed values can result in biases in the imputation since one imputation error will impact all following imputations. Another limitation is that the current GRU RNN version trained using Mean Squared Error (MSE) may not line up with the goal of imputing trends back into missing data.

Table 4.

Title: List of methods included in this paper

Description: Methods used with an abbreviation, category a short description and Python library of origin.

Method	Abbreviation	Category	Description	Library used
Forward filling	LOCF	Simple	Use the last cell before the gap to fill a gap	Pandas.DataFrame.fillna
KNN regression	KNN	Simple	Take the weighted average K-number of nearest neighbours.	Sklearn.impute.KNNImputer
GRU RNN	RNN	Neural Network	NN considers past values to impute missing data.	**None**
Hot deck	HD	Statistical	Take data from a different unit with a similar trend.	**None**

Imputation evaluation criteria

The aim of this paper is to create a selection of the most suitable imputation methods for certain scenarios with data classifications and gap sizes. To select the best method for each scenario evaluation criteria are required for this paper the selected criteria are Variance Error (VE) and Root Mean Squared Error (RMSE).

VE is used to give insight into the imputation method's ability to impute trends back into the missing data as that is one of the focal points of our research. The VE is calculated by calculating the difference in variance between the original and imputed data for each gap and then averaging it out if multiple gaps are present. To get the difference in variance in a gap pandas.var has been used.

In previous literature [] RMSE has been used to evaluate the performance of imputation on time series data. RMSE is calculated to give a comparison point for imputations done in this paper compared to results in previous research. The RMSE was calculated by taking the square root of the Mean Squared Error.

Results and discussion

With the developed pipeline and datasets as described in the methodology, an experiment will run with the settings of Tables 2 and 3 to determine the best imputation method per gap size and data classification.

The pipeline will evaluate each imputation method based on the evaluation metrics listed under the evaluation criteria per gap and feature. The imputation performance will mainly be evaluated based on the VE metric as mentioned in the evaluation criteria. RMSE is also calculated to evaluate imputation performance as seen in previous literature.

Based on the information in Table 7 several conclusions can be made by comparing the performance of the imputation method over various gap sizes and data classifications:

- The RMSE results show that the imputation performance as judged by traditional metrics is poor. The reasons could include that imputing using previous trends with HD is harder due to the volatileness of the data and the RNN bias. In the ratio features, a high bias was observed from the RNN imputations. This can be addressed by switching the RNN architecture to a bi-encoder-decoder sequence-based imputer.

- In all KNMI features selected for imputation HD was the best performing method. This might be because KNMI data is similar to each other in trends. In interval data, it is observed that HD performs 7.25% better on KNMI data but 58% worse in VE for BMS data when compared to RNN. The RMSE performance is worse too HD has a 6.75% better performance in KNMI interval data and a 128% worse performance in BMS interval data compared to RNN.

- VE and RMSE don't always give the same results in 8 out of 27 columns (op_mode excluded) different methods ended up on top. In CO2 sensor data gap 3 and 4, the difference between VE and RMSE is especially stark. When visualizing data it is visible that HD gets the original trend mostly right RNN gets at better RMSE score because on average it stays closer to the original data.

- Both RNN and HD try to impute trends back into the data

- The

Conclusion

This paper proposes a guideline to impute BMS nZEB data based on gap size and data classification. The problem of missing data in BMS is becoming a bigger problem in an era where buildings depend on big data. Previous studies have been done into imputing BMS time series data; this paper tries to build on that by creating a comprehensive guideline to follow for certain scenarios. To create a guideline 4 methods were chosen from previous literature: GRU RNN, Hot Deck, KNN algorithm and LOCF. During the research imputing trends back into missing data became the focal point of this study which is why Variance Error was Used instead of a more traditional metric like Root Mean Squared Error. The results of the experiment are as follows when formatted into a guideline:

Table no. 5
Title: Guideline for what method to use based on VE
Description: Guideline, method listed per Gap type and data classification

	Gap type 1.	**Gap type 2.**	**Gap type 3.**	**Gap type 4.**	**Gap type 5.**
Nominal	HD	HD	HD	HD	HD
Ratio	HD	HD	HD	HD	HD
Interval	RNN	RNN	RNN	RNN	RNN

Table no.6
Title: Guideline for what method to use based on RMSE
Description: Guideline, method listed per Gap type and data classification

	Gap type 1.	**Gap type 2.**	**Gap type 3.**	**Gap type 4.**	**Gap type 5.**
Nominal	HD	HD	HD	HD	HD
Ratio	KNN	HD	HD	HD	HD
Interval	RNN	RNN	RNN	RNN	RNN

The GRU RNN used for comparison in this paper has limitations that should be solved for a true comparison of RNN's ability to impute trends. The solutions to the limitations would be to use a sequence-to-sequence encoder such as an encoder-decoder structure and another non-error-based training metric should be used instead of MSE. This would allow the RNN to impute less biased sequences of data at once without using its own generated data.

Future work

In future work, the focus of research should be less on evaluating imputation with metrics based on the error but its impact on forecasting using imputed data. The effect on forecasting performance ought to be evaluated as it can provide a more complete view of imputation performance.

The data sets used for this study contain only numerical data and no ordinal data. To get a full view of the imputation performance on text-based categorical data further research is required.

The GRU RNN architecture used in the research had clear limitations based on how it was set up. To evaluate the full potential of imputation using RNN the architecture should be changed to an encoder-decoder sequential imputer-based design. This would remove the bias of imputing using its own imputed values.

Field	Method	HD	RNN	KNN	LOCF
	Gap type 1				
Temperature	**VE**	60.13	63.73	92.126	92.701
	RMSE	14.07	12.31	38.19	22.352
FLOW_TEMP	**VE**	8.67	7.82	10.76	10.76
	RMSE	5.29	3.44	5.34	6.79
op_mode	**VE**	0.08	0.07	0.08	0.08
	RMSE	0.49	0.51	0.45	0.56
Global Radiation	**VE**	337.621	369.87	620.279	620.85
	RMSE	25.16	29.846	66.47	53.02
Humidity	**VE**	15.38	15.45	20.71	20.80
	RMSE	6.78	6.19	14.52	9.88
Power	**VE**	137217	165879	158763	158763
	RMSE	686.24	794.46	632.54	800.01
C02	**VE**	372.92	409.15	420.02	420.02
	RMSE	44.41	42.11	35.29	43.03
	Gap type 2				
Temperature	**VE**	264.83	290.635	604.13	609.78
	RMSE	17.61	17.19	38.72	45.14
FLOW_TEMP	**VE**	32.892	19.262	39.59	39.631
	RMSE	8.03	3.45	8.43	10.28
op_mode	**VE**	0.28	0.33	0.33	0.33
	RMSE	0.84	0.87	0.83	0.99
Global Radiation	**VE**	1086.88	1427.15	2902.85	2904.88
	RMSE	32.52	45.92	67.81	91.24
Humidity	**VE**	51.67	59.07	108.11	108.60
	RMSE	9.08	9.45	15.12	18.78
Power	**VE**	357663	463033	494539	504114
	RMSE	895.32	1238.05	1176.21	1181.55
C02	**VE**	1393.59	1656.68	1782.42	1747.18
	RMSE	53.07	67.18	78.15	74.81
	Gap type 3				
Temperature	**VE**	328.88	381.24	901.32	912.82
	RMSE	14.39	17.18	37.874	45.98
FLOW_TEMP	**VE**	53	28.47	65.99	65.99
	RMSE	10.35	3.68	9.26	11.58
op_mode	**VE**	0.62	0.53	0.63	0.63
	RMSE	1.20	0.95	0.96	1.34
Global Radiation	**VE**	1138.15	2380.38	4392.39	4396.39
	RMSE	28.90	57.54	68.1	96.89
Humidity	**VE**	69.05	89.98	176.42	177.53

	RMSE	7.64	10.26	14.75	18.64
	VE	579467	1.04165e+06	1.36275e+06	1.36607e+06
Power	**RMSE**	997.66	1593.56	1725.46	1906.12
	VE	3862.98	4751.04	5272.42	5282.86
C02	**RMSE**	109.88	93.08	113.50	115.28
	Gap type 4				
	VE	415.57	407.244	1220.58	1228.37
Temperature	**RMSE**	14.67	18.23	42.08	52
	VE	54.81	38.209	73.699	73.701
FLOW_TEMP	**RMSE**	9.59	3.95	9.37	11.75
	VE	0.47	0.67	0.78	0.78
op_mode	**RMSE**	1.20	0.98	0.96	1.34
Global	**VE**	735.97	1715.78	4221.93	4222.48
Radiation	**RMSE**	22.94	64.08	65.86	97.46
	VE	67.07	95.857	187.31	187.40
Humidity	**RMSE**	7.12	11.46	15.21	19.94
	VE	760667	1.81155e+06	1.96192e+06	1.96194e+06
Power	**RMSE**	843.932	1581.06	1759.17	1992.42
	VE	6316.95	8877.94	9311.84	9315.94
C02	**RMSE**	115.84	107.93	123.39	142.02
	Gap type 5				
	VE	63.54	30.94	75.85	75.85
FLOW_TEMP	**RMSE**	9.12	3.83	9.48	12.60
	VE	0.15	0.52	0.598	0.598
op_mode	**RMSE**	0.89	0.92	1.59	0.92
	VE	746953	1.83388e+06	2.079629e-6	2.079629e-6
Power	**RMSE**	778.901	1489.48	1628.55	2218.56
	VE	5956.65	9578.38	9902.27	9902.33
C02	**RMSE**	105.32	105.44	114.78	131.26

References

1. Sree Dhevi, A. T. (2014). Imputing missing values using Inverse Distance Weighted Interpolation for time series data. 2014 Sixth International Conference on Advanced Computing (ICoAC). https://doi.org/10.1109/icoac.2014.7229721

2. Che, Z., Purushotham, S., Cho, K., Sontag, D., & Liu, Y. (2018). Recurrent Neural Networks for Multivariate Time Series with Missing Values. Scientific Reports, 8(1). https://doi.org/10.1038/s41598-018-24271-9

3. Pazhoohesh, M., Pourmirza, Z., & Walker, S. (2019). A Comparison of Methods for Missing Data Treatment in Building Sensor Data. 2019 IEEE 7th International Conference on Smart Energy Grid Engineering (SEGE). https://doi.org/10.1109/sege.2019.8859963

4. Poloczek, J., Treiber, N. A., & Kramer, O. (2014). KNN Regression as Geo-Imputation Method for Spatio-Temporal Wind Data. Advances in Intelligent Systems and Computing, 185–193. https://doi.org/10.1007/978-3-319-07995-0_19

5. Zhang, L. (2020). A Pattern-Recognition-Based Ensemble Data Imputation Framework for Sensors from Building Energy Systems. Sensors, 20(20), 5947. https://doi.org/10.3390/s20205947

6. Little, R., & Yau, L. (1996). Intent-to-Treat Analysis for Longitudinal Studies with Drop-Outs. Biometrics, 52(4), 1324–1333. https://doi.org/10.2307/2532847

7. Ma, J., Cheng, J. C., Jiang, F., Chen, W., Wang, M., & Zhai, C. (2020). A bi-directional missing data imputation scheme based on LSTM and transfer learning for building energy data. Energy and Buildings, 216, 109941. https://doi.org/10.1016/j.enbuild.2020.109941

8. KNMI. (n.d.). KNMI - Uurgegevens van het weer in Nederland. KNMI - Hourly weather data of Dutch climate. https://www.knmi.nl/nederland-nu/klimatologie/uurgegevens