MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

The Missing Input Problem

Laftchiev, Emil; Yan, Qing; Nikovski, Daniel N.

TR2020-172 December 16, 2020

Abstract

Rapid advances in information and communications technologies (ICT) have made it possible to deploy large collections of sensors to be used in traditional Supervisory Control and Data Acquisition (SCADA) systems and modern Internet of Things (IoT) installations. These sensors are intended for use in analytical formulas, AI algorithms, and traditional rulebased monitoring that determine optimal operation parameters, maintain smooth operation, or detect operation anomalies. Yet, advances in ICT do not always improve the reliability of data collection. Instead, the frequent use of consumer-grade sensors in IoT deployments, and an increasing array of customer choices often lead to inaccessibility of sensors or systematic absence of sensor readings. The lack of reliability in data collection leads to failures in the algorithms responsible for monitoring the system operation. We term this "the missing input problem", and discuss several state-of-the-art solutions. We specifically focus on the straightforward approach using standard imputation methods, as well as recent deep learning imputation methods. We show that none of the existing algorithms today perform very well in the face of missing sensors, and we outline several research directions that can lead to improvements.

IEEE Big Data

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

The Missing Input Problem

1st Emil Laftchiev
Mitsubishi Electric Research Labs
Cambridge, MA, USA 02139
laftchiev@merl.com

2nd Qing Yan
University of Chicago
Chicago, IL, USA 60637
yanq@uchicago.edu

3rd Daniel Nikovski
Mitsubishi Electric Research Labs
Cambridge, MA, USA 02139
nikovski@merl.com

***Abstract*—Rapid advances in information and communications technologies (ICT) have made it possible to deploy large collections of sensors to be used in traditional Supervisory Control and Data Acquisition (SCADA) systems and modern Internet of Things (IoT) installations. These sensors are intended for use in analytical formulas, AI algorithms, and traditional rule-based monitoring that determine optimal operation parameters, maintain smooth operation, or detect operation anomalies. Yet, advances in ICT do not always improve the reliability of data collection. Instead, the frequent use of consumer-grade sensors in IoT deployments, and an increasing array of customer choices often lead to inaccessibility of sensors or systematic absence of sensor readings. The lack of reliability in data collection leads to failures in the algorithms responsible for monitoring the system operation. We term this "the missing input problem", and discuss several state-of-the-art solutions. We specifically focus on the straightforward approach using standard imputation methods, as well as recent deep learning imputation methods. We show that none of the existing algorithms today perform very well in the face of missing sensors, and we outline several research directions that can lead to improvements.**

***Index Terms*—time series, missing sensor, imputation**

I. Introduction

Rapid advances in information and communications technologies (ICT) have made possible deployments of large collections of sensors used for remote monitoring, data collection, and device control. These developments have increased the size of traditional Supervisory Control and Data Acquisition (SCADA) systems and introduced a new type of installation termed the Internet of Things (IoT). Unfortunately, the larger number of sensors has not positively affected the reliability of the data collection process. There are several reasons for this. First, IoT deployments frequently use consumer-grade ICT components which can become faulty and/or inaccessible; second, both SCADA and IoT installations are affected by customer choices about the grade or level of deployment; and third, sensor deployments are affected by tasks such as routine maintenance that may render groups of sensors inaccessible for scheduled periods of time.

This creates a problem when the corresponding sensor readings are intended to be used in analytical formulas, AI algorithms, and traditional rule-based monitoring algorithms, whose goal is to monitor and optimize performance and detect anomalies. At test time, when the measurement is part of the input to an analytical formula or a rule, the corresponding output can only be computed if all arguments are known. When the measurement is an input argument to a statistical model such as an AI or ML algorithm, the output of the model is only predictable when all inputs are present, because the model is usually trained offline, with a full input set. Thus, the absence of a sensor severely affects the ability to automatically monitor and operate SCADA and IoT deployments, and can result in catastrophic failure of operation. For this reason, methods are needed for dealing with missing sensors values that result in a more graceful degradation of performance.

In general, dealing with missing data at random is a well-known problem in the field of machine learning, and various data imputation methods have been proposed over the years. The current problem of systemically missing sensors is different for two reasons. First, given the large number of deployed sensors, modern SCADA and IoT deployments may suffer sensor outages or inaccessibility for prolonged periods of time. Second, customer choices during the initial deployment may reduce the number of available sensors. When sensors are inaccessible for systematic reasons (e.g. choice made during deployment), or are faulty for extended periods of time, standard imputation methods fail because they can no longer leverage time-lag correlations within the sensor measurements to perform imputation. Thus, known solutions to the missing sensor problem, such as using imputed values, will typically fail. This will be especially noticeable when the target usage of a sensor is the computation of an analytical formula, or as an input to an AI/ML algorithm.

However, SCADA and IoT deployments exhibit the favorable circumstance in which there is often substantial redundancy and correlation among the readings of different sensors. Sometimes this is intentional, and sometimes it is not; often, by virtue of the ever-decreasing marginal cost of installing more sensors, a few extra sensors end up being installed. For example, identical temperature or vibration sensors can be installed at multiple points on the body of a large machine, with the hope that more sensors will give better indication when something goes wrong with the operation of this machine. This favorable circumstance makes data imputation methods that leverage cross-correlation between sensors good candidates for solving the problem of missing sensor data in IoT and SCADA installations via imputation.

However, while imputation methods based on cross-correlation between the sensors lead to a more graceful degradation of performance, the direct use of imputed val-

Laftchiev and Yan had equal contributions to this work. Yan contributed to this work during time spent at Mistubishi Electric Research Labs.

ues as inputs to functions is not optimal. This is because the imputation methods are not geared towards imputation of multiple systematically missing sensors. Many imputation methods essentially try to learn the covariance matrix across all sensor variables, but usually they only learn the covariance between strongly correlated variables. Ideally, the optimal solution would learn not just an approximation of the sensor covariance matrix, but a full probability distribution over the measured quantities. Such a distribution can be leveraged during exploitation when multiple sensors are systematically missing.

We believe that the absence of sensors for extended periods, whether due to faults, inaccessibility, or customer choices, presents a significant problem in SCADA and IoT systems. This problem prevents physics-based analytical functions, statistical models, or rule-based methods from being fully utilized for anomaly detection and operation of the underlying physical processes. To demonstrate the problem in this paper, we experiment with the well-known solution of using existing imputation methods to show the ability of these models to impute missing sensor time series. We show that despite the large volume of existing algorithms, few demonstrate fidelity in reconstructing the missing data. This indicates that the methods are not able to capture the correct correlations between the existing sensors and are very susceptible to further sensor faults and inaccessibility. In addition to standard methods of imputation, we envision that latent space/distribution methods may provide a good approach to learning better relationships in the system. Thus, the review presented here also introduces novel deep learning latent space methods geared toward the imputation of randomly missing data.

II. Problem Formulation

Formally, the problem can be stated as follows. We observe a set X of N variables, $x_i \in X$, $i = 1, \ldots, N$. This set of variables is used to compute a function $y = f(\cdot)$ whose output y is used at test time for system monitoring and optimization. At training time, a complete data set $x_1, \ldots, x_N$ is available. This allows the learning or specification of an optimal $f(\cdot)$. At deployment time, one or more of the variables x_i, $i = 1, \ldots, M$ are missing, making it impossible to directly compute the function $f(\cdot)$.

In contrast to the standard imputation problem where the variables in X have only some measurements missing at random, here the variables in X are systematically missing due to a reduced set of sensors chosen during deployment, or the long term failure of existing sensors. A further complication is that in some cases, the dependency between the sensors might change over time. In all cases, it is assumed that there are dependencies across the variables in X through the underlying system which is being monitored.

In the purely imputation-based (IB) setting, the effect of missing sensors is handled as follows. Let $X^- \subseteq X$ be the set of missing variables of X. Correspondingly, let the sets of available variables be denoted by $X^+ = X \setminus X^-$. If a variable $x_i \in X$ is missing, its value $\hat{x_i} = g_i(X^+)$ is imputed from some or all available variables in X^+, where $g_i(\cdot)$ is an imputation procedure. It is preferable that $g_i(\cdot)$ be a procedure that can work with arbitrary sets of available variables, including an empty one. Using $g_i(\cdot)$, the estimated values $\hat{x_i}$ of the variables in X^- can be substituted for the true values in the expression $f(\cdot)$ for computing z, such that the imputation-based monitoring output becomes, for example, $\hat{y}_I = f(x_1, \ldots, x_{i-1}, \hat{x_i}, x_{i+1}, \ldots, x_M)$, for the case when only one variable x_i is missing.

III. Background Literature

In this section, we review existing imputation algorithms that could be useful in the direct imputation setting. We focus on embedded space algorithms and generative models, because of these models' ability to learn an abstract representation of the data, from which other industrial AI applications can be deployed. This is different from standard methods of imputation (which will be described later) that focus on choosing the correct statistic for randomly missing data in a given application. Here, we explore black-box methods that might be broadly applicable to the missing sensor problem such as: the (variational) autoencoder, HI-VAE [1], GP-VAE [2], and BRITS [3]. We also briefly discuss two other standard methods for imputation: regression-based imputation [4] and matrix factorization [5]. There are also other approaches, such as generative adversarial network (GAN) based models [6], which leverage learning at test time for imputation. Such approaches are most likely too slow for online exploitation and are not reviewed here.

A. AE/VAE

The autoencoder (AE) [7] is a neural learning method for learning representative subspaces or data features of lower dimension than that of the original data. An autoencoder is a symmetric network, shown in Fig. 1, which maps the input data, X, to a lower dimensional encoding z via an encoding function $E(X)$, and then from the lower dimensional space back to an estimate of the input data via a decoding function function $D(z)$. The autoencoder is trained using a reconstruction loss that compares the difference between the input data, X, and the reconstructed data, $\tilde{X} = D(E(X))$.

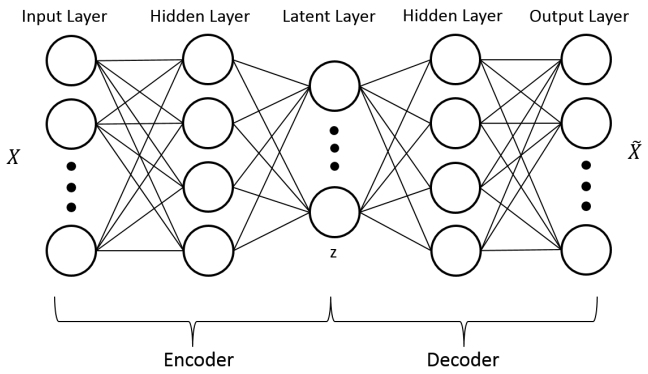

Fig. 1. An example of an autoencoder.

Building on the AE, the variational autoencoder (VAE) [8] learns a distribution in the embedded space and reconstructs the data from samples of this distribution. More specifically, the variational autoencoder is a latent variable model that learns the joint distribution $p(\mathbf{x}, \mathbf{z})$ where $\mathbf{x}$ in this paper represents a vector of all sensors x^i, and $\mathbf{z}$ is a vector of latent variables. Then, by marginalizing out $\mathbf{z}$, we could obtain the likelihood of $\mathbf{x}$: $\log p(\mathbf{x}) = \int p(\mathbf{x}, \mathbf{z}) d\mathbf{z}$. Unfortunately, this integral is usually intractable, making it hard to evaluate and optimize the likelihood. This leads to a proposed solution called variational inference. In particular, a VAE model is learned by maximizing the evidence lower bound (ELBO) as a surrogate to maximizing likelihood,

$$\text{ELBO} \triangleq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})} \log p_\theta(\mathbf{x}|\mathbf{z}) - KL(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})).$$

It is composed of an encoder to output the variational distribution $q_\phi(\mathbf{z}|\mathbf{x})$ and a decoder to provide the data distribution given latent variable $p_\theta(x|z)$. In practice, $q_\phi(\mathbf{z}|\mathbf{x})$ is set as a mean-field Gaussian distribution, and the encoder learns to map the data $\mathbf{x}$ to the parameters of this distribution. Typically, the prior distribution $p(\mathbf{z})$ is set as standard Gaussian $N(0, I)$. The prior can be viewed as a regularization on $q_\phi(\mathbf{z}|\mathbf{x})$. The distribution, $p_\theta(\mathbf{x}|\mathbf{z})$, could also take different forms for different types of $\mathbf{x}$. For example, for continuous variables, the distribution is typically chosen to be a multivariate Gaussian with a diagonal covariance matrix; for binary variables, the distribution is usually chosen to be a multivariate Bernoulli distribution with independent variables. The choice of independence type between variables is important because it restricts the ability of models to learn correlations among the inputs. However, in practice this has not been observed to be a great limitation.

Both the AE and VAE are algorithms that can be used for value imputation. To perform value imputation, the input is zero padded at the index of the missing values. The completed vector is mapped to the latent space/distribution, and the imputations are recovered from the reconstructed output. The idea is that the latent mapping will remain true even when the input is missing values. In practice, this is often not the case, and researchers have sought to improve on the models.

B. HI-VAE

HI-VAE (Heterogeneous Incomplete VAE) [1] is an approach that builds on the VAE to model heterogeneous data sets that are incomplete. To handle heterogeneous data, the authors propose to factorize the distribution modeled by the decoder and deploy different network layers for different input types. To model the incomplete data, a HI-VAE builds an encoder (recognition) network that takes the observed variables as input and outputs the corresponding latent distribution, and a decoder (generative) network transforms the latent distribution to the distribution of the observed and missing data. Missing data can occur both during the training stage and during the testing stage. During the training process, incomplete inputs are filled with zeros before being input into the (fully-connected) network to ensure missing variables are prevented from affecting the decoder output. The ELBO is only optimized over the observed variables because the missing variables are inaccessible. During the testing stage, HI-VAE takes a zero-padded incomplete data vector as input and returns imputed values sampled from the data distribution at the output of the decoder. To facilitate learning of the interaction of input variables, HI-VAE replaces the standard Gaussian prior with a Gaussian mixture distribution and introduces an intermediate homogeneous representation in the decoder.

C. GP-VAE

GP-VAE [2] is the first paper that specifically discusses imputation in the context of time series. The addition of time series, explicitly as a condition in this work, means that a fully factorized variational distribution would have independent components equal to the dimension of the latent space times the length of each time series window. In addition to having a large dimension, this distribution also does not accurately represent time series problems because consecutive points of a time series are not truly independent. To correctly account for the time lag information, GP-VAE uses an encoding network which consists of 1-D convolutional layers over each time window and a variational distribution with some off-diagonal covariance terms. These covariance terms account for the dependence in the time lag components of the time series. Samples of the latent distribution are transformed by another decoder network with several multi-layer perceptron blocks. Because of the special structure of the variational distribution, the prior distribution has to be tailored to time series data. GP-VAE applies Gaussian process with Rational Quadratic kernel as the prior to learn the dynamics of the time series in the latent space.

D. BRITS

An alternative to the embedded space and probabilistic models introduced so far is presented in BRITS [3]. The idea of this paper is to leverage the Recurrent Neural Network (RNN) machinery to impute missing values. Specifically, the authors augment the update equations of RNNs by imputing the missing dimensions of a data point x_t using the past hidden state of the RNN. Like HI-VAE, the authors calculate the training loss only over the observed values.

Dependency among the time series dimensions is added into the model via a feature estimation step. Here, the features are estimated from a complement state, which is the ordinary state with the imputed missing value calculated from the prior state of the network. This feature is combined into the state of the RNN. Lastly, to ensure stability, the authors run the model in a bi-directional fashion.

E. Regression-Based Imputation

A more traditional approach to imputation is regression-based imputation. In its most basic form, regression-based imputation takes the form of finding a regressor from the

observed sensor measurements to the missing sensor measurements. Because this regressor cannot be generated on-demand, all regressors with combinations of input and output sensors must be learned a-priori and stored. This means that storage complexity of this approach is proportional to the size of the power set of the number of system sensors. It is clear that such storage is prohibitively expensive for the large sensor deployments of modern SCADA and IoT sensors. Nonetheless, in this paper, we compare this performance as an upper bound on the performance of all algorithms.

To avoid storing all regressors, a round-robin approach can be used. Such an approach is implemented in scikit-learn under the name iterative imputer [9], building on the work of Buck [4] and on various imputation packages in R [10]. During training, the iterative imputer learns one regressor for each sensor from the remaining sensor set, and the mean of each sensor. During test time, all missing values are replaced with the mean of the sensor. Then, the values are imputed using the appropriate regressor in a prescribed imputation order. The imputation order is traversed a fixed number of times, which gives the method its name, the iterative imputer. We will include this imputation method in our experiments as well.

F. Matrix Factorization

A popular approach to imputation, particularly in the field of recommender systems, is matrix factorization. The idea of matrix factorization is to factor a data matrix $X_D \in \mathcal{R}^{d_1 \times d_2}$ into two low rank matricies, $U \in \mathcal{R}^{d_1 \times d}$ and $V \in \mathcal{R}^{d \times d_2}$, with dimension d equal to the rank of X_D. Here we can interpret U as capturing the correlation between points along dimension d_1 and V as capturing correlation between points along dimension d_2. For collected time series data, d_1 can represent the number of sensors and d_2 can represent the length of the recorded data T. Thus U captures the correlation among the recorded time series, while V captures the time lag correlation in each time series.

Having factored X_D into U and V, $\tilde{X}_D$ can be recovered by taking the dot product UV^T. When data is missing at random and the approximations of the matrices are correct, the reconstructed matrix should include good reconstructions of the missing values. For the case of missing sensors, this approach suffers from the inability to compute accurate low-rank matrices U and V. Thus, we expect the precision of the imputed values to suffer.

IV. Experimental Set-up

A practical issue that has hampered the adoption of AI algorithms in physical industries has been the lack of standardized data sets. In this paper, we want to compare the performance of several previously proposed models along with two baseline methods. To perform this comparison, we designed our own experiment, which mimics the type of industrial problem that can be commonly found.

We created an experimental test bed based on a cooling loop, shown in Fig. 2. The cooling loop is created using a Thermaltake C360 hard tubing PC water cooling kit. The loop consists of two heat sinks attached to 40W silicone heaters, one pump, and one cooling radiator cooled by three 120mm fans. The loop is equipped with the following sensors: temperature sensors after each component (a total of 4), 2 flow sensors, and 5 room air temperature sensors that are placed at the corners of the test bed and in the middle of the test bed. The cooling and heating of the loop, as well as the data collection of the loop, are controlled via an Arduino MEGA 2560 with a custom built IO board. The two heating commands and the three fan speed commands are sampled from a multivariate Gaussian distribution with a random covariance matrix. The heat commands, x_i, are discretized as $x_i = 0$ if $x_i < 0$ else 1. The generated fan speed commands are truncated to integer levels from 0 to 300. Each command (heating and cooling) is issued to the loop for 2 minutes, and data samples are continuously collected at 3-second intervals.

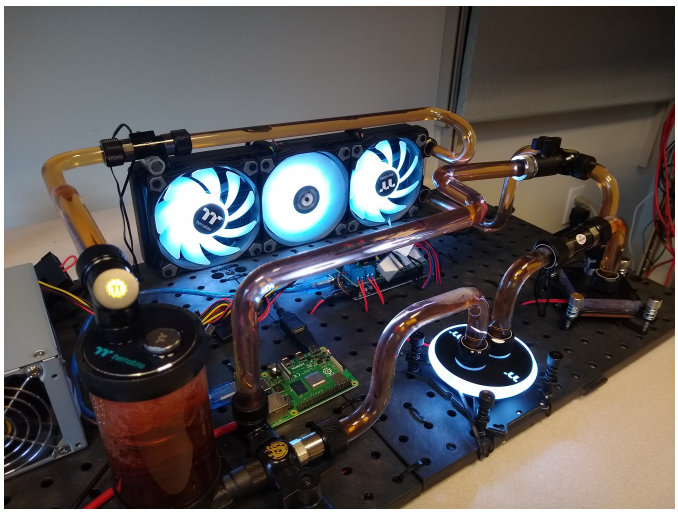

Fig. 2. Cooling Loop Test bed.

The experimental test bed described here can be used to collect an unlimited amount of data. Here, we use a 4-hour data collection window, generating the loop control commands randomly. The resulting data set contains 7271 data samples. This data set is used below in model evaluation.

V. Numerical Results

A. Standard Imputation Methods

In addition to the imputation methods discussed in the background section, in this work we also test all easily implementable imputation methods using scikit-learn [9] and the Python library fancyimpute. In particular, we evaluate simple imputation methods such as mean, median, and most frequent value; the kNN imputation method that averages the k-Nearest Neighbors to impute the missing values; SoftImpute, a method developed by Mazumder et. al. [11] which computes the SVD of the data matrix and applies a soft threshold to the singular values; and Iterative SVD [12], which performs imputation via SVD but does not apply soft thresholding.

B. Data Subset Selection

In the context of missing values of sensor measurements, we observe that the majority of imputation methods simply predict the mean of a given sensor measurement, the average of other sensors that are highly correlated, or a linear combination of the existing sensors. Thus, those methods can be used on systems with redundant sensors. However, in this paper, we would like to explore the performance of imputation methods when there are correlated (possibly highly correlated) sensors, thus learning how much of the underlying system is learned from the sensor measurements. In the data set collected for this paper, the following sensors are available: Left Fan Speed, Middle Fan Speed, Right Fan Speed, Loop Flow, Fluid Temperature at Heater 2, Fluid Temperature After Cooling, Fluid Temperature after Heater 1, Fluid Temperature after Loop Pump, Room Temperature on Control Board, Room Temperature Upper Right of Test Bed, Room Temperature Lower Left of Test Bed, Room Temperature Upper Left of Test Bed, Room Temperature Lower Right of Test Bed, Heat 1 Command, and Heater 2 Command. From this set of sensors, we select the following subset for our imputation experiments: Left Fan Speed, Middle Fan Speed, Loop Flow, Fluid Temperature at Heater 2, Fluid Temperature After Cooling, Room Temperature on Control Board, Room Temperature Upper Left of Test Bed, Heat 1 Command, and Heater 2 Command.

In particular, we use a subset of measurements that is redundant in only one other sensor. Thus, this provides the opportunity to examine the imputation algorithms to determine if they are learning the physical properties of the system or simply learning the direct correlation between sensors.

C. Experimental Results

Using the data set collected for this experiment and the subset of sensors specified above, we first perform an experiment using the standard imputation methods specified in section V-A. In this experiment, we train an imputation method and then sequentially remove each of the sensor time series, impute their values, and calculate the normalized root mean square error (NRMSE) of the imputed time series with respect to the true time series. NRMSE is normalized by the standard deviation of the true time series and reveals how well a method performs with regard to using simply the mean value on the training set for imputation. The experiment is repeated in a 5-fold cross-validation. The average error is reported for all sensors with a mean and standard deviation. The results for standard imputation methods are shown in Table I.

The results of standard imputation methods are illustrative. These approaches show that no approach outperforms mean imputation and that a simple statistic like NRMSE does not tell us the full information about an imputation method. We first evaluate the NRMSE measurment by comparing the KNN imputation, median impuation, and most frequent imputation to mean imputation. All comparison methods perform worse than mean imputation on average as well as when taking into account the standard deviation of the error. In fact, median and most frequent imputation results in an NRMSE over five times worse then mean imputation. Yet these results also show the drawback of using a single statistic like NRMSE to report imptuation of entire sensor measurements. For comparison, we plot the Fluid Temperature after Cooling and Room Temperature on Control Board normalized in Fig. 3. From this plot it is clear that a mean imputation would have small NRMSE but would not capture any of the system dynamics. Thus mean imputation would not be a good approach when the goal of imputation is to input the imputed values into a monitoring algorithm such as those described in section I.

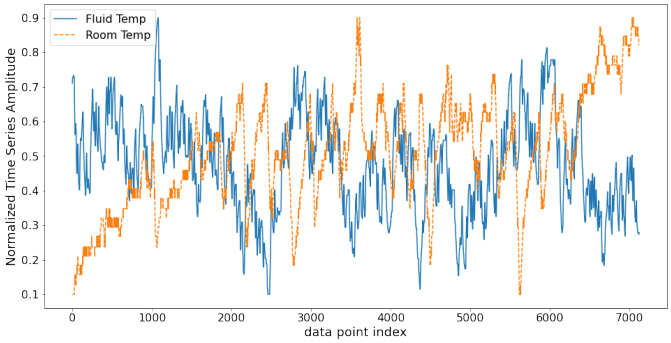

Fig. 3. Sample Temperature Time Series

Next we perform experiments with the more complex, modeled imputation methods. We experiment with three methods that attempt to model the data matrix in a low dimensional space, and the iterative imputer regression method that models the missing time series from the observed time series. The results for these experiments are shown in Table II. Interestingly, we note that the soft imputation outperforms all other matrix based imputation methods, but it is still inferior to simply using the iterative imputer which uses a regressor to model the missing time series. Because we have already established the inadequacy of judging imputation performance by NRMSE, we now plot the imputation performance for SoftImpute (Fig.4), Iterative SVD (Fig.5), and Matrix Factorization (Fig.6) on the Fluid Temperature of Heater 2 and for the Regression method. Note here we choose a fluid temperature sensor because it is a continuous sensor which is the easiest type of sensor to impute. From these plots we see that regressive imputation has the best performance; however, iterative SVD appears to learn the time series, including its dynamics, up to an offset factor.

The plots shown in Figures 4-6 show that, qualitatively, only regression imputation and iterative SVD imputation demonstrate the ability to learn time series dynamics of the missing time series. This appears to fit our intuition about the methods. Methods that substitute an average value for the missing time series do not capture any of the time series dynamics. Methods that try to learn an imputed data matrix via matrix factorization are designed for the case when values are missing at random and do not perform well when a whole time series is missing. On the other hand, methods that focus on finding strong correlations in the data learn well how to relate one sensor time series to another.

TABLE I
IMPUTATION ERROR FOR STANDARD IMPUTATION APPROACHES.

	Mean	Median	Most Freq.	kNN-1	kNN-2	kNN-3	kNN-4	kNN-5
Loop Flow	1.05	4.98	4.87	1.42	1.41	1.40	1.39	1.38
Left Fan Speed	1.00	5.14	5.62	1.32	1.30	1.29	1.28	1.27
Middle Fan Speed	1.01	5.17	5.51	1.18	1.17	1.16	1.15	1.15
Heat 1 Command	1.03	7.51	7.51	1.34	1.33	1.32	1.32	1.31
Heat 2 Command	1.03	7.59	7.59	1.28	1.27	1.25	1.25	1.24
Fluid Temperature (Heater 2)	1.04	5.19	10.52	0.71	0.70	0.69	0.69	0.69
Fluid Temperature after Cooling	1.04	5.28	5.76	0.76	0.75	0.74	0.74	0.73
Room Temperature on Control Board	1.11	5.45	5.27	1.08	1.08	1.07	1.07	1.06
Room Temperature Upper Left of Test Bed	1.08	5.27	5.34	1.18	1.17	1.17	1.17	1.17
Overall μ	0.94	5.16	5.8	1.03	1.02	1.01	1.00	1.00
Overall σ	0.33	2.06	2.66	0.43	0.43	0.43	0.42	0.42

TABLE II
IMPUTATION ERROR FOR MODELED IMPUTATION APPROACHES.

	SoftImpute	Iterative SVD	Iterative Imputer	Matrix Fact.
Loop Flow	1.10	2.80	1.05	2.91
Left Fan Speed	1.19	2.49	0.88	1.83
Middle Fan Speed	1.22	2.24	0.77	1.95
Heat 1 Command	1.35	1.55	1.00	1.73
Heat 2 Command	1.33	1.60	1.03	1.50
Fluid Temperature (Heater 2)	1.37	3.27	0.23	2.12
Fluid Temperature after Cooling	1.39	3.15	0.23	2.57
Room Temperature on Control Board	1.56	3.19	0.71	2.24
Room Temperature Upper Left of Test Bed	1.30	4.04	0.81	3.04
Overall, $\mu \pm \sigma$	1.18 $\pm$0.43	2.43 $\pm$1.15	0.67 $\pm$0.38	1.99 $\pm$0.86

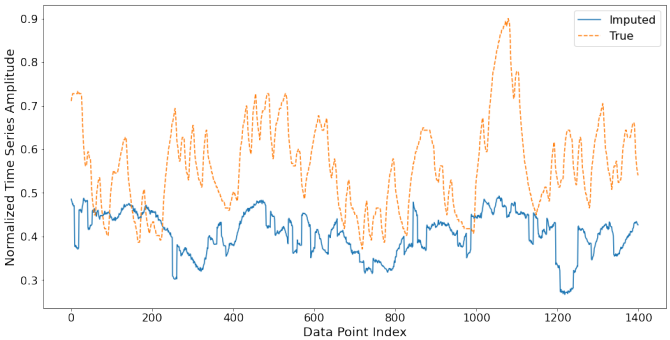

Fig. 4. Sample imputation of a time series using SoftImpute.

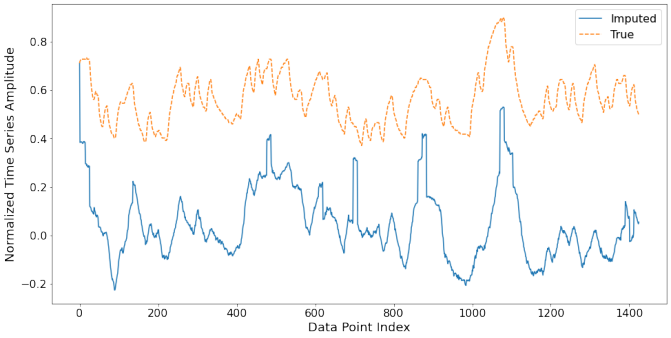

Fig. 5. Sample imputation of a time series using Iterative SVD.

Next, we evaluate using imputation based on regression. This is motivated by the relatively good qualitative performance demonstrated by the iterative imputer with respect to

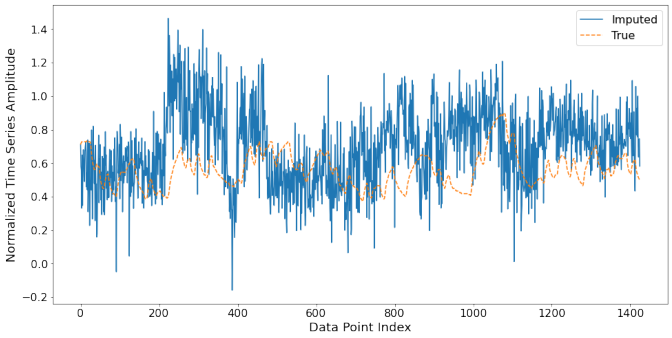

Fig. 6. Sample imputation of a time series using Matrix Factorization.

NRMSE. Specifically, we are interested in how much regression models for regression-based imputation can learn beyond the relationship to the sensor with the highest correlation. To evaluate the performance of the regression-based method, we plot a sample of the Fluid Temperature of Heater 2 time series when only this sensor is missing, in Fig. 7, and when both fluid temperature sensors are missing, in Fig. 8. We observe that when a highly correlated sensor exists in the observed data, regression-based imputation can almost exactly impute the missing time series by copying the correlated sensor. However, when the correlated sensors are both imputed at the same time as in Fig. 8, we observe that imputation suffers. This shows that regression based imputation likely identifies the most correlated sensor and then imputes based on this sensor. This is further illustrated in Fig. 13, which shows that the average NRMSE of the imputed sensors degrades as a function

of the number of missing sensors.

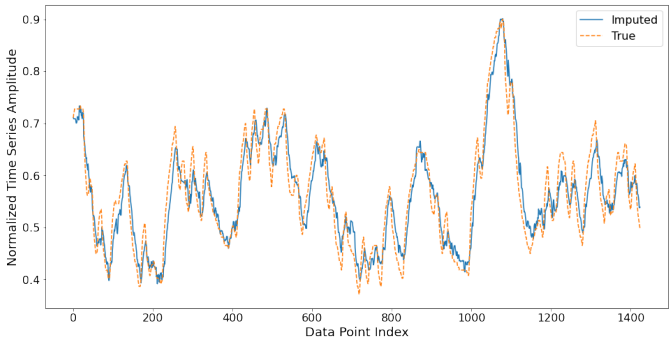

Fig. 7. Regression-based imputation with one highly correlated sensor.

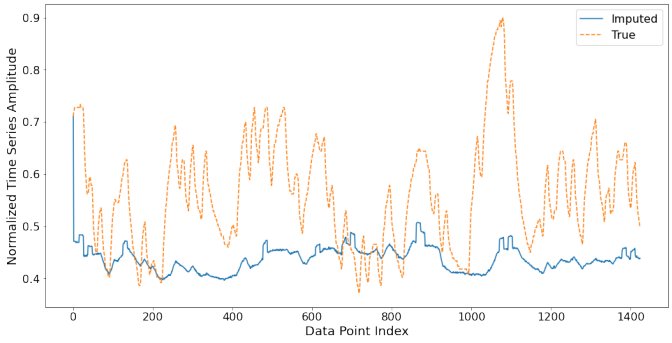

Fig. 8. Regression-based imputation without the presence of highly correlated sensors.

It is clear that standard methods of imputation are not sufficient to address the challenge of missing sensors when the purpose of imputation is to use the imputed time series in monitoring algorithms. We evaluate next the ability of modern deep learning methods to impute missing time series. In particular, we would like to find out if deep learning methods can learn to model the instantaneous correlation between the time series and the time lag correlation between and within the time series. We focus on four models: AE, VAE, HI-VAE, and GP-VAE. The results of imputing a single sensor for these methods is shown in Table III.

Observing the mean NRMSE and standard deviation of the RNMSE in Table III shows that the deep learning methods appear to outperform mean imputation, but continue to exhibit poorer performance than standard regression based methods. It is notable that standard AE/VAE models have performance on par with HI-VAE and GP-VAE because the latter two methods are specifically developed for heterogeneous data (HI-VAE) and time series data (GP-VAE). Thus, it appears that neither accounting for heterogeneity nor accounting for time lag has benefited these models. We suspect that this is the case because these algorithms were still developed for the case of randomly missing data and do not fully learn the underlying system processes.

Next, we evaluate the performance of HI-VAE when a single fluid temperate sensor is missing and when both fluid temperate sensors are missing. The results are shown in Fig. 9 and 10. We observe that while HI-VAE is capable of mostly imputing the missing time series for one sensor, it is unable to do so when all correlated sensors are removed. In fact, its latter performance is on par with the performance of the regression-based imputation, which suggests that the more complex model does not benefit this problem.

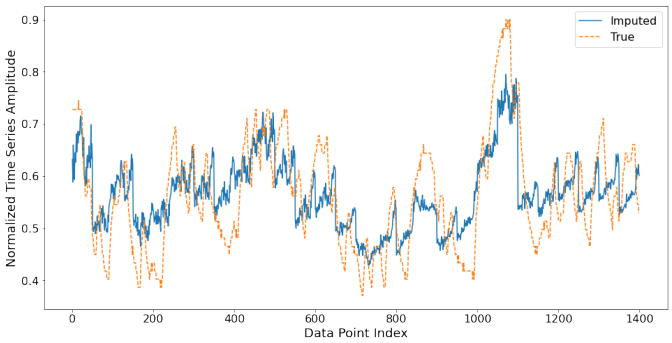

Fig. 9. Imputation of HI-VAE for one missing sensor when another highly correlated sensor is present..

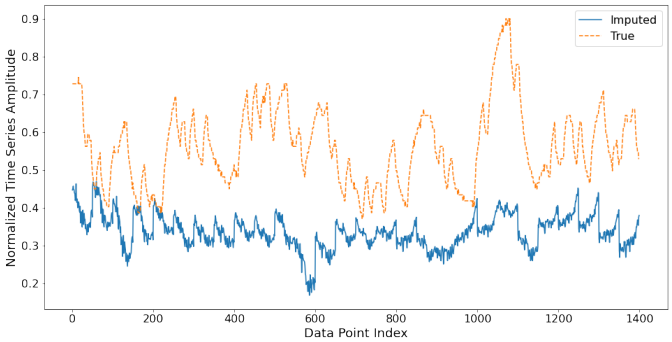

Fig. 10. Imputation of HI-VAE for both missing temperature sensors, i.e. when both correlated sensors are missing.

We repeat this evaluation for GP-VAE in Figs. 11 and 12. Interestingly, here we observe that while the initial reconstruction of the missing sensor is poorer than HI-VAE and regression-based imputation, its imputation when both correlated sensors are removed shows that the model learned at least the low frequency behavior of the system. This suggest that the time series component of the model is important in this application.

Lastly, we plot the degradation of the performance of the HI-VAE, GP-VAE, the iterative imputer, and standard least squares linear regression with respect to the number of missing sensors. Here the least squares linear regression is compared by fitting a model from the existing sensors to the missing sensors for each combination possible given the number of missing sensors. The results reported are the average of all missing sensor combinations for a given number of missing sensors. The results are shown in Fig. 13. We observe all methods have a significant drop in NRMSE with just one

TABLE III
IMPUTATION ERROR FOR MODELED IMPUTATION APPROACHES.

	AE	VAE	HI-VAE	GP-VAE
Loop Flow	1.10	1.05	1.05	1.04
Left Fan Speed	0.95	1.00	1.01	0.93
Middle Fan Speed	0.87	0.90	0.95	1.00
Heat 1 Command	1.19	1.11	1.33	1.15
Heat 2 Command	1.04	1.10	1.20	1.18
Fluid Temperature (Heater 2)	0.44	0.31	0.62	0.51
Fluid Temperature after Cooling	0.46	0.31	0.50	0.43
Room Temperature on Control Board	0.88	0.87	0.99	0.86
Room Temperature Upper Left of Test Bed	1.00	1.05	0.87	0.97
Overall, $\mu \pm \sigma$	0.88 $\pm$.025	0.86 $\pm$0.30	0.95 $\pm$0.24	0.89 $\pm$0.25

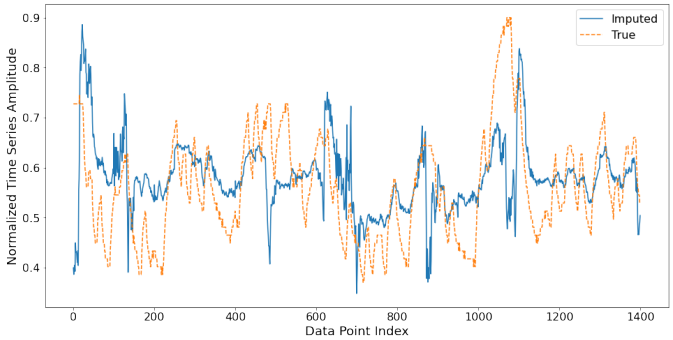

Fig. 11. Imputation of GP-VAE for one sensor with a highly correlated sensor present.

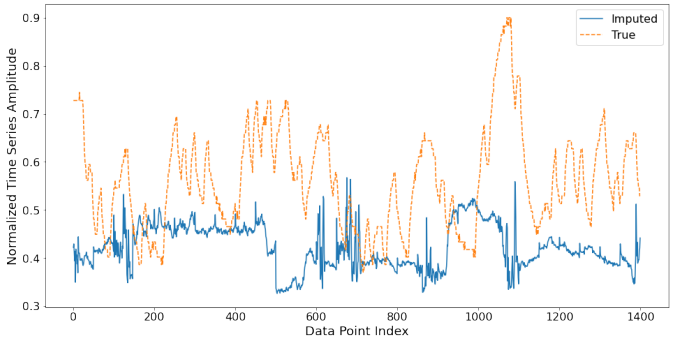

Fig. 12. Imputation of GP-VAE for both missing temperature sensors, i.e. when both correlated sensors are missing.

missing sensor. Linear regression outperforms all other methods, with the iterative imputer outperforming HI-VAE until half of the sensors are missing. Thus, HI-VAE has a more graceful degradation in the presence of missing sensors than the iterative imputer. Last among all of the methods is GP-VAE with the fastest degradation of performance of all the methods.

VI. DISCUSSION AND RESEARCH CHALLENGES

The previous section explored the ability of classical imputation methods and state-of-the-art deep learning methods to impute missing time series. Recall here that the goal of this paper is to evaluate the simple imputation approach to correcting for missing sensors in SCADA or IoT deployments

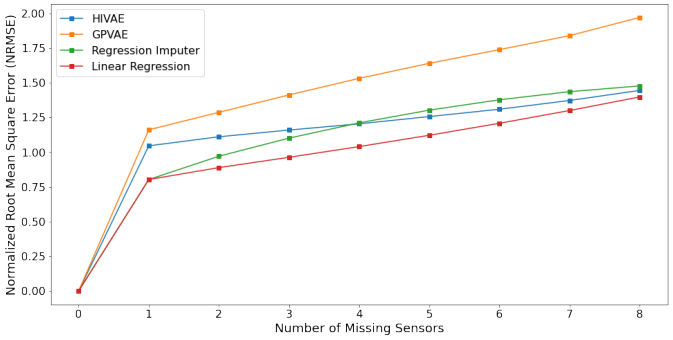

Fig. 13. Degradation of imputation method as the number of missing sensors increases.

as described in section II. The setting in which these methods are tested is very different from the classical setting of imputation, for which all the reviewed methods have been proposed. This is because when data is missing at random, the existing data can provide both temporal context and correlation context among the recorded sensors.

In this paper, we are interested in recovering information that would have been present in a missing time series. It appears that if only a single sensor is missing at test time, and there exists a strong correlation between the missing sensor and other measurements, then the standard approach of using regression from other existing sensors is the best way of recovering the missing information. Yet, the performance of this approach quickly deteriorates as the number of missing sensors with correlation increases.

Modern deep learning methods offer an approach to imputation that holds the promise of learning generative models of the system. The promise of these models is that learning will go beyond the simple first order correlation that is learned in regression-based models. Unfortunately, these models do not appear to fulfill this promise to date. In the course of this study, we identify four difficulties in the state-the-art that necessitate further research.

A. Deeper learning

The experiments in this paper show that, at the moment, classical and deep learning methods learn only the closest correlation among the sensor variables. In addition, the present generation of methods does not emphasize learning over

subsets of sensors and instead only focuses on imputing missing values at random. Solutions are needed that emphasize learning system behavior from subsets of relevant sensors. We can think of this as methods that enforce a learning regularization such that the relevant sensors contribute to the prediction of a derived variable.

B. Training Loss

With regard to the deep learning models used for imputation, an important research challenge is to choose the correct loss function used in learning. In the course of this work, we experimented extensively with the mean squared loss and dynamic time warping (DTW). We found that the added complexity of the DTW did not yield improved time series imputation. In addition, we experimented with adding a loss component that measured the MSE between the imputed and true Fourier domain of the time series window. This also did not yield improved learning in the models.

The challenge of determining the proper time series loss stems from the fact that the sensor data encountered in industry is not always continuous. In fact, in the experiments for this work, the data contained continuous, categorical, and discrete data. The treatment of such heterogeneous time series data is still very much an open question in the field of deep time series analysis.

C. Evaluation Metric

Another challenge we observe is how to evaluate the imputed time series. In the computer vision community, the Frechet Inception Distance (FID Score) [13] has been developed to evaluate the quality of generated images. No such metric exists for time series. Yet, without such a metric, many imputation methods appear the same. For example, we note that the mean imputation has the same NRMSE as the regression based imputation. On the surface these methods are identical, but by plotting the two imputation methods as we have done here, we observe that a large amount of temporal information is lost in mean imputation but preserved when using regression. Thus, research is needed in quantifying the quality of generated or imputed time series.

D. Target Task

Lastly, there is a need for benchmark tasks and data sets on industrial time series data. These tasks could serve as the derived variables in the problem discussed in this paper. Several candidate tasks exist: anomaly detection, regime classification, and prediction/forecasting. Yet, unlike in the field of computer vision where researchers have compiled benchmark data sets and innovate around mutually agreed benchmark tasks, the field of time series industrial data remains fragmented with multiple candidate tasks and unpublished data sets. To motivate increased interest and to speed development in this field, its is critical that these benchmarks be established by the research community.

VII. Conclusion

In this work, we introduced the missing sensor problem. We evaluated experimentally the suitability of existing classical approaches of imputation and cutting edge approaches in deep learning in addressing the missing sensor problem. The setting for this evaluation is the straightforward imputation of the missing time series information. We demonstrated that the methods defining the state-of-the-art were not able to recover the missing data, particularly when multiple sensors were missing, and we challenge the research community to innovate by discussing the relative strengths and weaknesses of the proposed algorithms.

References

[1] A. Nazábal, P. M. Olmos, Z. Ghahramani, and I. Valera, "Handling incomplete heterogeneous data using vaes," *Pattern Recognition*, p. 107501, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320320303046

[2] V. Fortuin, D. Baranchuk, G. Raetsch, and S. Mandt, "Gp-vae: Deep probabilistic time series imputation," in *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, S. Chiappa and R. Calandra, Eds., vol. 108. Online: PMLR, 26–28 Aug 2020, pp. 1651–1661. [Online]. Available: http://proceedings.mlr.press/v108/fortuin20a.html

[3] W. Cao, D. Wang, J. Li, H. Zhou, Y. Li, and L. Li, "Brits: Bidirectional recurrent imputation for time series," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, ser. NIPS'18. Red Hook, NY, USA: Curran Associates Inc., 2018, p. 6776–6786.

[4] S. F. Buck, "A method of estimation of missing values in multivariate data suitable for use with an electronic computer," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 22, no. 2, pp. 302–306, 1960. [Online]. Available: http://www.jstor.org/stable/2984099

[5] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[6] Y. Luo, X. Cai, Y. ZHANG, J. Xu, and Y. xiaojie, "Multivariate time series imputation with generative adversarial networks," in *Advances in Neural Information Processing Systems 31*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds. Curran Associates, Inc., 2018, pp. 1596–1607. [Online]. Available: http://papers.nips.cc/paper/7432-multivariate-time-series-imputation-with-generative-adversarial-networks.pdf

[7] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[8] D. P. Kingman, "Variational inference and deep learning: A new synthesis," Ph.D. dissertation, University of Amsterdam, 2017.

[9] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[10] S. van Buuren and K. Groothuis-Oudshoorn, "mice: Multivariate imputation by chained equations in r," *Journal of Statistical Software, Articles*, vol. 45, no. 3, pp. 1–67, 2011. [Online]. Available: https://www.jstatsoft.org/v045/i03

[11] R. Mazumder, T. Hastie, and R. Tibshirani, "Spectral regularization algorithms for learning large incomplete matrices," *Journal of Machine Learning Research*, vol. 11, no. 80, pp. 2287–2322, 2010. [Online]. Available: http://jmlr.org/papers/v11/mazumder10a.html

[12] O. Troyanskaya, M. Cantor, G. Sherlock, T. Hastie, R. Tibshirani, D. Botstein, and R. Altman, "Missing value estimation methods for dna microarrays," *Bioinformatics*, vol. 17, pp. 520–525, 07 2001.

[13] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 6626–6637. [Online]. Available: http://papers.nips.cc/paper/7240-gans-trained-by-a-two-time-scale-update-rule-converge-to-a-local-nash-equilibrium.pdf