Feature Engineering in Big Data Analytics for IoT-Enabled Smart Manufacturing – Comparison between Deep Learning and Statistical Learning

Devarshi Shah*, Jin Wang*[+], Q. Peter He*[+]

*Department of Chemical Engineering, Auburn University, AL, 36849, USA
[+] Corresponding Author (Tel: 334-844-7602; email: qhe@auburn.edu)

Abstract

As IoT-enabled manufacturing is still in its infancy, there are several key research gaps that need to be addressed. These gaps include the understanding of the characteristics of the big data generated from industrial IoT sensors, the challenges they present to process data analytics, as well as the specific opportunities that the IoT big data could bring to advance manufacturing. In this paper, we use an inhouse-developed IoT-enabled manufacturing testbed to study the characteristics of the big data generated from the testbed. Since the quality of the data usually has the most impact on process modeling, data veracity is often the most challenging characteristic of big data. To address that, we explore the role of feature engineering in developing effective machine learning models for predicting key process variables. We compare complex deep learning approaches to a simple statistical learning approach, with different level or extent of feature engineering, to explore their pros and cons for potential industrial IoT-enabled manufacturing applications.

Keywords: Internet-of-Things, smart manufacturing, big data, data analytics, feature engineering, deep learning, statistical learning

1 INTRODUCTION

The emergence of the industrial Internet-of-Things (IoT) devices and ever advancing computation and communication technologies have fueled a new industrial revolution. Although the application of IoT devices in manufacturing processes has been limited, there have been high expectations that IoT will be a key enabler for making manufacturing systems smarter, safer and more efficient. Specifically, the small size and low cost of IoT devices make it possible to equip manufacturing systems with a large number of IoT sensors. As a result, the big data generated by the IoT devices and other sensors is expected to offer unprecedented opportunities to significantly advance smart manufacturing.

Big data is arguably a major focus for the ongoing worldwide transformation of advanced manufacturing. How to extract manufacturing intelligence from huge amount of process data to support evidence-based and timely decision-making will likely play a key role in determining the competitiveness, productivity growth and innovation of the manufacturing industry (Manyika et al., 2011). However, currently there are limited studies on the challenges associated with analyzing big data generated from IoT devices and how to address them (Zhong and Ge, 2018). This is likely due to the lack of IoT big data from real applications to work with. To address this challenge, we

have developed several IoT-enabled manufacturing testbeds in-house, including a flow system equipped with 5 IoT vibration sensors and a continuous stirred-tank reactor equipped with 28 IoT temperature sensors. With real big data collected from the IoT-enabled manufacturing testbeds, we can examine the characteristics of the IoT big data, the challenges they present for data analytics, and explore the potential of IoT devices for advancing smart manufacturing. In this paper, using IoT big data collected from the flow system testbed, we aim to answer the following two questions: first, can we extract useful information from IoT big data? Second, if yes, what would be an efficient way to obtain such information? In particular, with the renewed interests in artificial neural networks (ANN), we aim to examine the role of feature engineering by comparing two main data-driven modeling routes: deep learning vs statistical learning, with different level or extent of feature engineering.

Even without IoT sensors, databases of industrial manufacturing systems are already massive due to the pervasive use of computations and information systems in process operations. As a result, many data-driven machine learning algorithms have been developed to model different industrial process operations. One particular research area, process monitoring, has received significant interests because of its importance in maintaining long-term reliable operation of any system. Statistical process monitoring approaches, i.e., a group of statistical learning algorithms such as principal component analysis (PCA) and partial least squares or projection to latent structure (PLS), have enjoyed tremendous success in monitoring various industrial processes. In the era of smart manufacturing, the importance of process monitoring will only become greater; correspondingly, it is important to examine whether these statistical learning algorithms can be adapted to address the new challenges presented by IoT big data.

At the same time, with the availability of big data and significantly advanced computing power, ANN started to draw renewed research interests in the process systems engineering (PSE) community. Back in the 1990s, ANN had been studied extensively to learn hidden-patterns from input-output data, enabled by the reinvention of the backpropagation learning algorithm (Venkatasubramanian, 2019). However, the impact of ANN had been limited in PSE, partially due to the lack of powerful computing and the lack of large amount of data. Now with the significant advancement in computing power, development of deep neural network (DNN), as well as availability of big data in smart manufacturing, artificial intelligence (AI), DNN in particular, is again poised to make real impact to process operations.

In this work, with the IoT big data (~70GB) collected from flow system testbed, we explore the potential of deep learning approaches in handling the IoT big data, and compare deep learning approaches with a representative linear statistical learning approach, PLS. Specifically, we examine whether the big data collected from vibration IoT sensors can be utilized to model the testbed system and predict the pump motor speed (in revolution per minute or RPM) and water flow rate (in gallon per minute or GPM). In our previous research, we suggested that feature-based monitoring could offer an effective solution for next-generation process monitoring, which is also well positioned to address the 4V challenges associated with the IoT big data (He and Wang, 2018a, 2018b). In this work, we examine how feature engineering that is rooted in system knowledge and human learning could enhance both complex deep learning and simple statistical learning. It is worth noting that, for statistical learning, certain sample distribution (e.g., multivariate Gaussian) is assumed, while for machine learning, including DNN, data distribution is not assumed to be known or to follow certain form.

The rest of the paper is organized as follows. Section 2 presents the setup of the IoT-enabled flow system testbed; Section 3 briefly reviews the statistical learning approach and deep learning approaches examined in this work; Section 4 discusses feature engineering for machine learning; Section 5 presents the results of different modeling approaches, with an emphasis on the effect of feature engineering; finally Section 6 draws the conclusions.

2 DEVELOPMENT OF AN IoT-ENABLED MANUFACTURING TESTBED

One of the most versatile equipment in manufacturing industry are centrifugal pumps and compressors and the associated piping systems that move gas or liquid from one location to another. Therefore, in this work we chose a laboratory-scale pipe flow system where water flow is driven by a centrifugal pump and controlled by valves as the testbed. We equipped the testbed with five IoT vibration sensors (a.k.a. accelerometers) at different locations. The detailed setup of the testbed is discussed in Sec. 2.1. It is worth noting that vibration sensors have long been used for the monitoring of rotary machines and fluid flows in industrial processes. However, most of them are the conventional piezoelectric vibration sensors (Evans et al., 2004; Hu et al., 2019; Lannes et al., 2018), which are bulky, energy hungry and expensive (Jung et al., 2017). The vibration sensors used in this study are the new generation IoT vibration sensors based on micro-electro-mechanical systems (MEMS) technology. These sensors are small, cheap and with low power consumption rate. The IoT vibration sensors also have some drawbacks, including the reduced accuracy and error-prone data, which have been shown to pose significant challenges to data processing and analytics (Jung et al., 2017). It is also worth noting that by no means the testbed is a challenging problem that only IoT vibration sensors can address. On the contrary, it is a system simple enough that we can perform designed experiments with actual measurements, so that we can validate different modeling approaches based on the IoT big data.

2.1 Setup of the IoT-enabled manufacturing testbed

The IoT testbed was set up by non-invasively mounting five tri-axis accelerometers ADXL345 at different locations of a laboratory-scale pipe flow system as shown in Figure 1. The sensors are marked by red ellipses and numbered in the figure. In this testbed, water flows in the pipes, which is driven by a centrifugal pump and controlled by valves. The pump assembly contains a variable drive motor, pump impeller, impeller casing, coupling, electrical connections, and a knob for changing motor speed (in revolution per minute or RPM). The motor shaft and impeller shaft are connected by a coupling. The pump sucks water from a reservoir and pumps it back to the reservoir. The piping system has a suction valve and a discharge valve, but no bypass. The specific sensor locations are: sensor 1 – top of the water pipe bend; sensor 2 – the last section of the water pipe near the exit; sensor 3 – motor; sensor 4 – coupling; sensor 5 – impeller casing. There is one flow meter measuring the water flow rate (in gallon per minute or GPM), which can be changed by the discharge valve opening and/or pump motor RPM. The pump motor speed (RPM) and water flow rate (GPM) are displayed on a computer screen refreshing/updating every second. The details on the hardware and software protocols responsible for data acquisition, communication and storage can be find in (Shah, 2019).

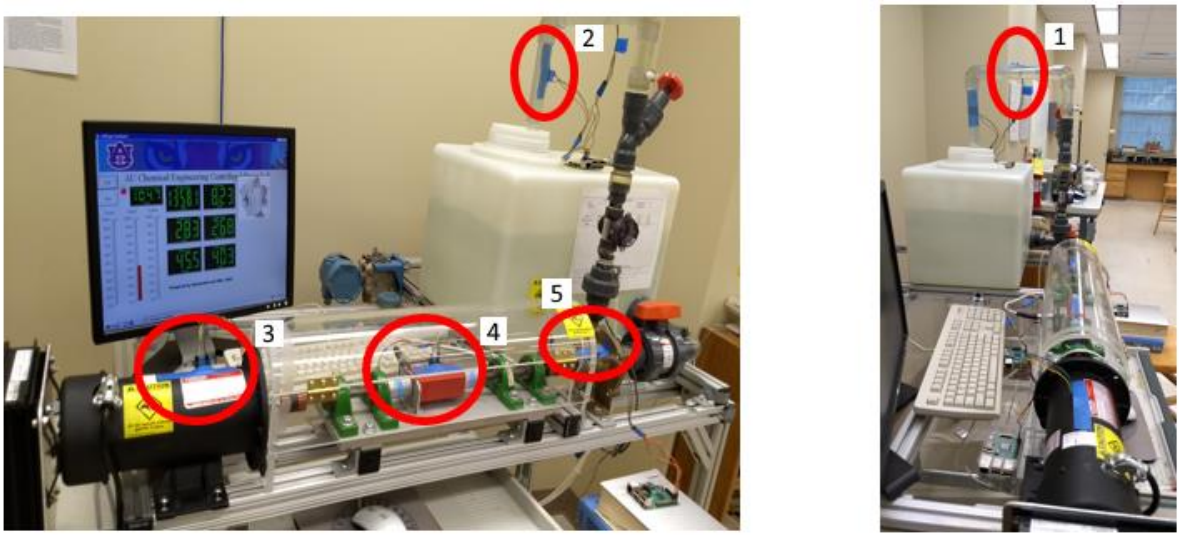


Figure 1. The IoT testbed with five tri-axis ADXL345 vibration sensors mounting at different locations of a laboratory-scale water flow system driven by a centrifugal pump. The sensors are marked and numbered.

2.2 Experiments and data collection

For the testbed, the pump operation range is from 1500 RPM to 2500 RPM. At 1500 rpm, the minimum flow rate that can be measured reliably by the flow meter is 5 gpm. At the other end, the maximum flowrate that can be achieved at 2500 rpm and maximum discharge valve opening is around 16 gpm. Process data collected include both the vibration signals from IoT sensors and pump motor speed (RPM) and water flow rate (GPM) from computer screen display. The accelerometers are triple axis accelerometers and thus measure vibration signals in x, y and z directions of Cartesian coordinate system. The RPM and GPM readings were captured by an IoT video camera and stored as video streams. To cover the full range of the potential operation conditions, we design a series of 85 experiments and Table 1 lists the conditions corresponding to these experiments. Each experimental condition corresponds to one combination of motor speed and flow rate by adjusting motor speed and discharge valve. For each experimental condition, data were collected for 10 min after the systems reached relative steady-state, which results in about 3×10^6 readings (1600 Hz × 60 ×10 min × 3 directions) from IoT vibration sensors and 1800 (3 Hz × 60 ×10 min) readings from the IoT camera for RPM and GPM. Note that the motor speeds and flow rates listed in Table 1 were approximate as they drifted during the course of the experiments and the real-time readings from the computer screen were used as the actual values.

Table 1 Experiments performed on the testbed

# of Conditions	Motor speed (rpm)	Flow rate (gpm)
3	1500	5, 7, 9
3	1600	5, 7, 9
4	1700	5, 7, 9, 11

4	1750	5, 7, 9, 11
4	1800	5, 7, 9, 11
4	1850	5, 7, 9, 11
4	1900	6, 8, 10, 12
4	1950	6, 8, 10, 12
5	2000	5, 7, 9, 11, 13
5	2050	5, 7, 9, 11, 13
5	2100	6, 8, 10, 12, 14
5	2150	6, 8, 10, 12, 14
5	2200	6, 8, 10, 12, 14
5	2250	6, 8, 10, 12, 14
5	2300	7, 9, 11, 13, 15
5	2350	7, 9, 11, 13, 15
5	2400	7, 9, 11, 13, 15
5	2450	7, 9, 11, 13, 15
5	2500	8, 10, 12, 14, 16

2.3 Characteristic of IoT vibration data

Although the IoT vibration data collected from the testbed have all the 4V characteristics (i.e., volume, variety, velocity and veracity) of big data, in this section we focus only on veracity, i.e., the quality of the data. Since the quality of the data usually has the most impact on process modeling, data veracity is often the most challenging characteristic of big data that we have to address in order to ensure the quality of the obtained model. Specifically, the IoT big data we obtained from the testbed have unequal or variable sample intervals; they are corrupted by high level noises; and they contain significant missing values. In addition, the type and quality of the internet connection affect data quality as well. These characteristics are briefly discussed below and more details can be found in (Shah, 2019; Shah et al., 2019a).

Although the IoT vibration sensor sampling rate is set to be 1600 Hz, the actual sampling rate, i.e., the rate based on when data is reported and stored in the database, is ultimately determined by the rate of data acquisition and communication. Therefore, the actually sampling rate varies from sample to sample due to CPU time variations. For most traditional data analytics or modeling techniques that deal with time series data, equal sampling interval or frequency is assumed. Obviously, this could be addressed by resampling (or down-sampling) of the raw data, but such treatment could distort the information contained in the raw data, and the effect of such resampling has not been examined. In addition, because of the high sensitivity of the accelerometer used in this work, the signals obtained are very noisy. Although denoising methods such as various filtering approach are available, the effect of denoising on signal distortion and information loss can vary depending on the method and associated parameters, as well as the signal itself. Another

challenge is the large chunk of missing data. This might be due to occasional connection failures or communication delays between the micro-controller and the sensor. Missing measurements is associated with most sensors, no matter IoT or traditional sensors. However, the frequency and duration of miss data segments seem to be higher and longer for IoT sensors than the traditional ones. Finally, as wireless internet technology is fundamental to the concept of IoT-enabled smart manufacturing, the effect of network connection on the quality of IoT data is of significant importance. To examine the effect of wireless connection on data characteristics, we compared static noises collected from an IoT vibration sensor that was fixed on a non-mechanical stationary surface over wireless connection vs wired connection. Our results indicate that noises over wireless network show much wider distribution, suggesting an additional layer of noise due to wireless connection.

3 BRIEF REVIEW OF MODELING APPRAOCHES COMPARED IN THIS WORK

In this work, one research goal is to examine the performance of deep learning approaches on analyzing IoT big data, and compare it with a commonly used simple statistical learning approach. The other research goal is to examine the effectiveness of feature engineering on different machine learning approaches. Specifically, we examine the performance of different machine learning approaches in predicting the motor rotation speed (RPM) and water flow rate (GPM) using big data generated by IoT vibration sensors. For deep learning, we examine a recurrent neural network (RNN) approach, specifically long short-term memory (LSTM), for modeling the raw data collected by the IoT vibration sensors; and a conventional feed-forward deep neural network (DNN) for modeling the frequency spectra (i.e., features) extracted from the sensor measurements. For statistical learning, we choose the most commonly used PLS, which is a simple linear machine learning approach. Here we provide a brief review of different machine learning methods and feature engineering.

3.1 Statistical learning – partial least squares or projection to latent structure (PLS)

PLS is a well-known and well established statistical learning method for finding fundamental linear relations between observed variables and responses. Due to its simplicity, PLS is arguably the most commonly applied statistical modeling approach in industrial applications. Let $X \in R^{n\times p}$ be the input matrix and $Y \in R^{n\times q}$ be the output matrix, where n, p and q are the number of samples, input variables (usually process variables) and output variables or responses (usually quality variables), respectively. In PLS modeling, l latent variable pairs in X and Y (denoted as T and U) are first extracted such that each pair of them capture the maximum variance in X and Y matrices while maximizing the correlation between them.

$$X = TP^T + E$$

$$Y = UQ^T + F$$

The final linear regression model between X and Y can be written as

$Y = X\beta + F$ where $\beta = PBQ^T$

PLS only has one tuning parameters: the number of latent variable pairs l. In this work, l is determined through cross-validation.

3.2 Deep learning approaches

3.2.1 Deep neural network (DNN)

Unlike the ANN of the 1990s, which typically had only one hidden layer of neurons, a deep neural network (DNN) has multiple hidden layers (Figure 2). Because the search space and hence the computation increase exponentially with the number of layers, DNN was only made possible by recent algorithmic advancement in training strategies coupled with technological advancement in computing power. ANN, including DNN, usually have 4 main components: neurons, connections & weights, activation functions and learning rule. In standard DNN, flow of information is always forward to the next layers (as shown in Figure 2) and thus is also called feed-forward DNN.

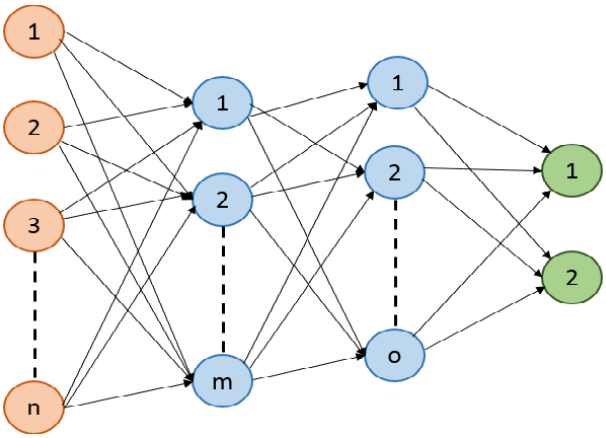


Figure 2. Schematic diagram of a feed-forward DNN with two hidden layers

3.2.2 Long short-term memory (LSTM) Network

A feed-forward DNN only captures the static relationship between input and output, as it has no notion of temporal order. However, most industrial processes are dynamic systems, whose current output is affected by both the current input and the system history. Therefore, the data generated by dynamic systems are time-series, and the temporal order of the samples plays an important role in capturing the system dynamics. Recurrent neural net (RNN) is capable of capturing such dynamics by taking both the current input and what it has seen previously. Because the output now depends on what has occurred before, the network behaves as if it has "memory" (Rumelhart et al., 1986). Long short-term memory (LSTM) network is an improved version of RNN by introducing a cell which can store the value over arbitrary time intervals, and three gates (i.e., input gate, output gate and forget gate) regulate the flow of information into and out of the cell (Hochreiter and Schmidhuber, 1997). As a result of the flexible memory unit, LSTM networks are well suited for predicting output based on time-series data.

3.2.3 Tuning parameters of neural networks

For neural networks, there are several hyperparameters that have to be determined before network training could be conducted. The major ones are: network structure (i.e., the number of hidden layers and the number of neurons for each layer), number of epochs, batch size and network weight initialization. Commonly used method to tune these hyperparameters are manual search, grid

search, random search, and Bayesian optimization (Bergstra et al., 2011; Nielsen, 2015). However, each method has its pros and cons. In this work, extensive manual search was conducted to optimize the neural network performance.

3.3 Brief review of feature engineering

Feature engineering involves the transformation of given feature/variable space, typically using mathematical functions, with the objectives of reducing modeling error (Khurana et al., 2018). The importance of feature engineering has long been recognized (Bengio et al., 2013) (Khurana et al., 2018; Wind, 2014). There even has been a famous claim that "Applied machine learning is basically feature engineering" (Ng, 2013). On the other hand, the challenge of feature engineering has also long been recognized (Anderson et al., 2013) (Khurana et al., 2018). Currently there is no well-defined basis for performing effective feature engineering. Here we summarize different feature engineering approaches into three categories: explicit feature engineering, implicit feature engineering and knowledge-guided feature engineering.

3.3.1 Explicit or direct feature engineering

Explicit or direct feature engineering uses some mathematical transformation functions, a.k.a. constructor functions, to transform the given features into a new set of features. For example, Khurana et al., (2018) propose a transformation graph that enumerates the space of feature options based on the following twenty-two transformation functions: Log, Square, Square Root, Product, Z-Score, Min-Max-Normalization, Time-Binning, Aggregation (using Min, Max, Mean, Count, Std), Temporal window aggregate, Spatial Aggregation, Spatio Temporal Aggregation, k-term frequency, Sum, Difference, Division, Sigmoid, BinningU, BinningD, NominalExpansion, Sin, Cos, Tanh. Other methods construct new features by applying similar functions, such as FICUS (Markovitch and Rosenstein, 2002), FEADIS (Dor and Reich, 2012), and Cognito (Khurana et al., 2016). Most of the explicit or direct feature engineering methods discussed here can be automated to become automated or automatic feature engineering (AFE), which is an area of active research in the machine learning field.

3.3.2 Implicit or indirect feature engineering

Some machine learning methods perform feature engineering implicitly or indirectly. Examples in this category are kernel methods, which are among the most popular feature engineering methods. Kernel methods have been incorporated in different algorithms such as kernel SVM for classification, kernel PCA for dimensionality reduction, and kernel K-means for clustering. Kernel methods use implicit feature engineering by working with the similarity of the data points in the input space, but not the explicit representation in the transformed feature space (Wang et al., 2017). This is because the kernel trick is used to make problems solvable without ever having to explicitly map the original data into a very high dimensional kernel space. There are some key weaknesses of kernel transformations, including the space- and time-inefficiency, susceptible to poor results with increasing irrelevant input features, and lack of interpretability of the kernel features even when extracted explicitly (Wang et al., 2017). Another class of machine learning method that performs feature engineering indirectly are dimensionality reduction or feature combination methods, such as PCA, which aim at mapping or combining the raw features into a lower-dimensional space with fewer features.

3.3.3 Knowledge and human learning guided feature engineering

Although the explicit features generated using the mathematical transformation functions are mostly interpretable, they may not carry any physical meaning. This may not be important for some applications, such as image classification or computer vision, speech recognition, music classification, and machine translation. However, for most engineering applications, especially process engineering applications, the physical meanings of the features are important for making diagnostic inferences about specific predictions. In this aspect, domain knowledge and human learning through data visualization and exploration can play a significant role. In addition, our previous work with feature-based process modeling suggest that quite often the key of developing an effective data-driven process model is to identify features that can effectively characterize the behavior of a process or system (He and Wang, 2018). For example, auto- and cross-correlations of selected process variables can capture key system dynamics and lead to enhanced process monitoring performance (He and Wang, 2011; Wang and He, 2010). Other knowledge-guide feature engineering, such as landmark features (Wold et al., 2010); profile-driven features (Rendall et al., 2017); geometry based features (Wang et al., 2017), have also been proposed in the PSE community.

4 COMPARISION BETWEEN DEEP LEARNING AND STATISTICAL LEARNING FOR IoT BIG DATA ANALYSIS

For the IoT-enabled pipe flow testbed, we collected IoT vibration sensor data (i.e., the input variables or predictors), as well as the pump motor speed (RPM) and water flow rate (GPM) data (i.e., the outputs or responses) under 85 different operation conditions. However, the input and output variables were collected at drastically different frequencies. The IoT vibration sensor data were collected at roughly 1600Hz, while the sampling frequency of RPM and GPM was only 3 Hz after image processing. To handle the different sampling frequencies between input and output data, instead of down sampling the IoT data, we paired 800 IoT samples with one pair of RPM and GPM readings extracted from the images. As the vibration sensor measures vibration in x, y and z directions, the input vector corresponding to one output pair (RPM and GPM) consists of 2400 variables (800×3) per sensor.

Data collected under all the conditions were used for model training and testing. Both training and testing sets contain samples from all conditions. For each condition, the first 8 minutes of collected data were used for training, and the last 2 minutes of data were used for testing. Samples in the training set are randomized and further divided into calibration (70%) & validation (30 %) sets. In addition, the segments with large missing chunks of data were removed from training and testing. In the end, the total number of samples used for calibration, validation and testing are 48,073, 20,604 and 16,488, respectively.

Although vibration signals from all five IoT sensors were collected, our analyses show that data from sensor 4 (i.e., the one attached to the coupling) is the most relevant to both RPM and GPM due to its central location between motor and water pipe. When data from all five sensors were used to predict RPM and GPM, the results from PLS were slightly worse than those using data from sensor 4 alone. This is probably due to the fact that the benefit of extra information added by extra sensors is outweighed by the cost of added noise and increased curse of dimensionality. We did not test DNN using data from all five sensors as it would take a long time and significant computing resources to train the models and find optimal hyperparameters. Therefore, in this work we only included the data collected from sensor 4 for analysis to simplify the comparison.

However, data collected from other sensors do contain valuable information. For example, we have demonstrated that sensor 4 can be reconstructed using the other sensors should the malfunction occur to sensor 4 (Shah, 2019). These results are not included in this work as they are outside of the scope of this work. In addition, we suspect that sensors 1 and 2 are probably better candidates for predicting GPM since they are directly attached to the pipe and are far away from the motor. On the other hand, sensor 3 might be better in predicting RPM since it is directly attached to the motor. Similarly, sensor 5 might be the best for detecting pump problems. These hypotheses are from the viewpoint of signal decoupling, which require further validation.

In this section, we compare deep learning and statistical learning approaches on their capabilities in handling the challenges presented in IoT big data. In addition, we examine the effect of feature engineering on the performance of both modeling approaches. Specifically, we first apply LSTM and PLS modeling approaches on the raw IoT sensor data to see if they could extract relevant information from the vibration signals. Next, based on system information, we perform feature engineering, i.e., extracting frequency content from the raw, time-domain vibration signals. DNN and PLS are then applied to predict RPM and GPM using samples' frequency spectra. Finally, guided by system information and human learning through data visualization and exploration, we perform additional feature engineering by identifying informative features for building the prediction models. To evaluate different machine learning methods, the performance of different methods are evaluated using root mean squared error (RMSE) of the test set, as defined below.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(\hat{y}-y)^2}{n}} \quad (1)$$

where y and $\hat{y}$ are the predicted and measured RPM or GPM, respectively. n is the number of samples in the test set.

4.1 Modeling using the raw data collected by the IoT sensors

First, we examine if LSTM and PLS could capture the hidden relationship between the inputs (vibration signal) and output (RPM and GPM). As discussed before, the raw data collected by IoT sensors are not equally spaced, but LSTM requires equally spaced time series data. To address this, the sample segments with large missing chunks of data were removed. Without the segments of missing data chunks, the variation in sampling frequency was relatively small and we simply used the raw data directly for model training and testing. In this way, we could use the models' prediction performance to indirectly examine whether the modeling approach could effectively handle such small abnormality.

The tuning of LSTM network (i.e., determining the hyperparameters) was through extensive manual search, and the final selected LSTM network has 3 hidden layers: the first two layers consist of fully connected LSTM units, and the third layer consists of fully connect neurons. The initial weights of the LSTM network was assigned randomly following a uniform distribution. Table 2 lists the value of the hyperparameters for LSTM. For PLS, the only tuning parameter is the number of latent variables, which is selected through cross-validation. The numbers of latent variables for the RPM and GPM models were 8 and 1, respectively.

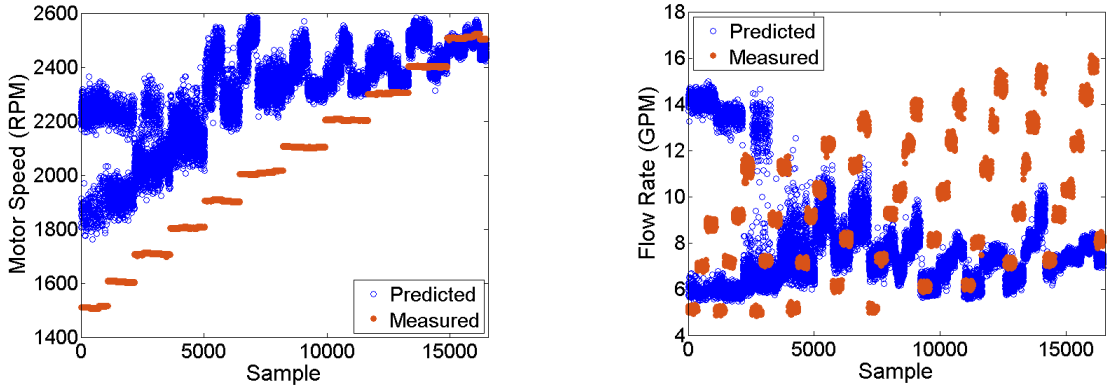
Table **2**. Hyperparameters for LSTM

Hyperparameter	Value
# of LSTM units in HL1	200
# of LSTM units in HL 2	100
# of neurons in HL 3	100
# of epoch	200
Batch size	801

Figure 3 and 4 compare the predictive performance of the LSTM network and PLS on the testing data set, with RPM in (a) and GPM in (b). The RMSE's of different modeling approaches are reported in Table 3, where the first block row corresponds to the results using the raw IoT big data. The other results are discussed later.

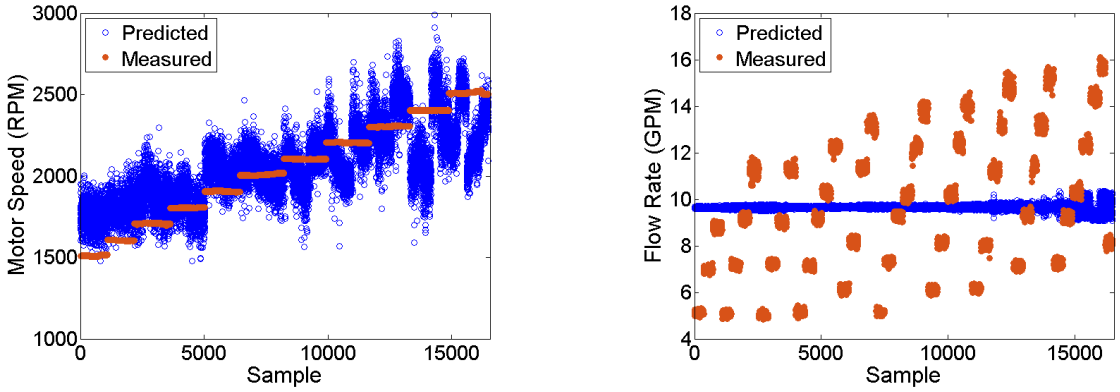
Table 3. RMSE of different modeling approaches with different level of feature engineering

		RMSE(RPM)	RMSE(GPM)
Models using raw IoT big data	LSTM	335.99	3.99
	PLS	195.97	2.96
Models using whole frequency spectra	DNN	12.21	0.34
	PLS	13.16	0.78
Model using selected features	PLS	0.21	0.09



(a) (b)

Figure 3. Deep learning from raw IoT data: (a) Measured and LSTM-predicted motor speed (RPM); (b) Measured and LSTM-predicted flow rate (GPM)



(a) (b)

Figure 4. Statistical learning from raw IoT data: (a) Measured and PLS-predicted motor speed (RPM); (b) Measured and PLS-predicted flow rate (GPM)

Figure 3 and 4 clearly indicate that neither modeling approach was able to effectively capture the correlation between the vibration signal and the system output (RPM and GPM). For the LSTM network, since it is capable of capturing any nonlinear relationship, its poor performance could be attributed to the high noises in the data, and perhaps unequally spaced data. It is possible that a different network structure could yield better prediction performance. However, it is extremely time-consuming to train and tune the LSTM network because of the long time series of the input data. For PLS, the poor performance could be explained by the fact that PLS is a linear modeling approach and cannot capture the likely nonlinear relationship between the input and output data. For example, the number of latent variables selected for the GPM model is only one, and the PLS prediction is basically the average of GPM's across all conditions. In addition, the high noise level in the input data could be another reason for poor prediction performance of PLS, reflected by the highly noisy prediction of RPM.

However, it is interesting to notice that although both modeling approaches fail to capture the underlying correlation between input and output data, PLS appears to perform better than LSTM. Specifically, the PLS prediction of RPM roughly followed the trend of experimental measurement, although very noisy, while the LSTM predictions of both RPM and GPM showed little correlation with the measurement. These results suggest that simply applying deep learning tools on available big data may not guarantee better performance than that of simpler, or even linear, but more robust statistical learning methods.

4.2 Feature engineering via projecting time series data into frequency domain

The modeling results from the previous section clearly indicate that the available deep learning and statistical learning approaches cannot predict RPM and GPM from raw vibration signal. Our previous work suggests that feature-based modeling, especially features that can capture the process characteristics, could offer an effective solution to address the challenges associated with IoT big data (He and Wang, 2011; Q.P. He and Wang, 2018; Shah et al., 2019b; Suthar et al., 2018; Wang and He, 2010). In feature-based modeling, features generated from process data, instead of process data themselves, are paired with output data to build the model. In this section, we explore feature engineering to improve the prediction performance from different models.

Because the pump operation is inherently periodic, and higher rotation speed would result in higher vibration frequency, it makes sense to extract frequency spectra as process features from the raw vibration signal. In addition, most of the noises contained in the vibration signal would be in higher frequency and hence could be easily separated from the frequency resulted from pump rotation. Therefore, we expect that using the frequency spectra as the features to predict RPM and GPM would result in improved prediction performance.

Fast Fourier transform (FFT) is the most commonly applied approach to extract frequency spectrum from time-series data. However, FFT requires equally spaced samples. When this condition is not satisfied, there is no guarantee that FFT would yield reliable frequency spectrum. To address this challenge, one could implement some preprocessing steps such as interpolation and signal binning to make the data equally spaced. However, these approaches all have their limitations and could distort the information contained in the raw data. For this consideration, we choose a robust least-squares spectral analysis approach, Lomb's algorithm, to extract the frequency spectrum from the raw data. Lomb's algorithm is commonly used in astronomy community, which estimates a signal's frequency spectrum based on a least squares fit of sinusoids to data samples (Lomb, 1976). The most significant advantage of Lomb's algorithm in this study is that it can handle unequally spaced samples without introducing information distortion or down-sampling.

Figure 5 shows the spectra of 500 randomly selected samples corresponding to different conditions, which covers frequency range from 1 to 800 Hz with a resolution of 0.2. Figure 5 confirms that most information clustered around low frequency range (30 to 300 Hz) with distinct peaks, while the rest of the spectrum, especially the high frequency range, are mainly noises.

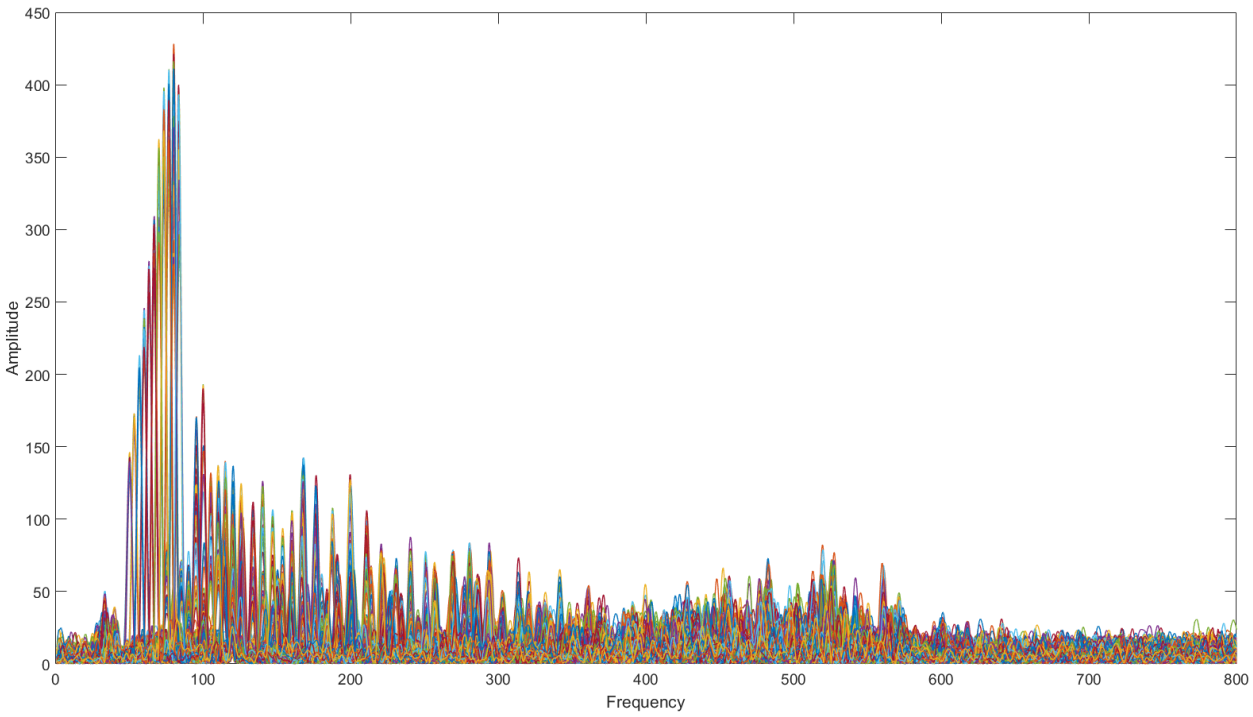
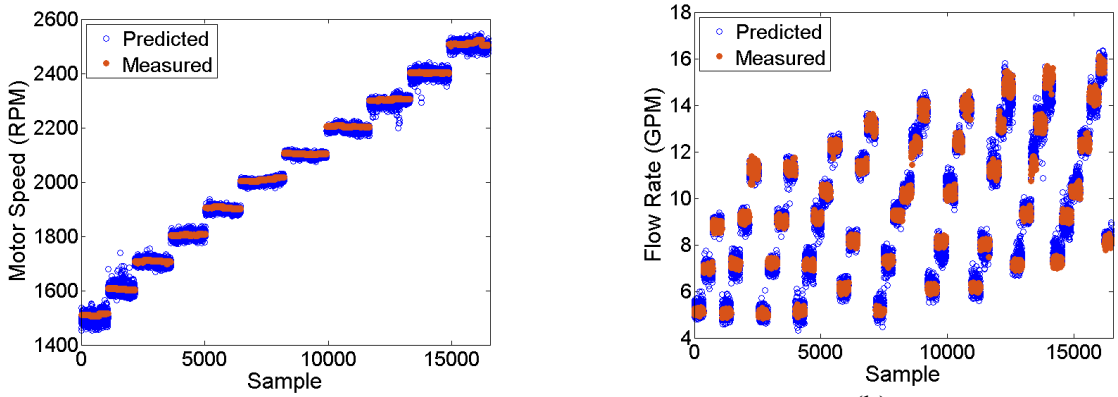


Figure 5. Spectra of 500 samples after Lomb transform.

4.3 Modeling using frequency spectra

With the whole frequency spectrum as process features to predict RPM and GPM, we again compare the modeling performance of a deep learning approach with a statistical learning approach. Since the frequency spectrum is no longer a time series sequence, a standard DNN would be sufficient for the modeling purpose. In this exercise, we chose a DNN with 4 hidden layers, and each layer has 4,000, 2,000, 1,000 & 300 neurons respectively. The input layer has 12,000 neurons, which correspond to the spectra input from all three directions, and the output layer has 2 neurons. The activation function for hidden neurons was rectified linear unit (ReLU). The number of epochs was set as 100, and the batch size was 600. The other hyperparameters were kept at recommended values. For the statistical learning approach, we still use PLS. The number of latent variables selected for the RPM and GPM modes are 29 and 13, respectively.

Figure 6 and 7 compare the prediction performance of DNN and PLS on the testing data set, with RPM in (a) and GPM in (b). The RMSE's of different modeling approaches are reported in Table 2. These results clearly show that by using the process features (i.e., the whole frequency spectrum) as the inputs to build the model, both modeling approaches achieved significantly improved performance. For example, for RPM prediction, the RMSE's from models based on the frequency features are only 3.6% and 6.7 % of those from the raw data-based models for deep learning approach and statistical learning approach respectively. The performance of DNN is slightly better than PLS in predicting RPM and noticeably better in predicting GPM. This is likely due to the DNN's capability in capturing nonlinear relationships. It is also interesting to notice that the number of the latent variables for PLS models both increased, especially for the GPM model. This suggests that frequency spectrum generated from the raw vibration data can help separate noises from signals.



(a) (b)

Figure 6. Deep learning from frequency features: (a) Measured and DNN-predicted motor speed (RPM); (b) Measured and DNN-predicted flow rate (GPM)

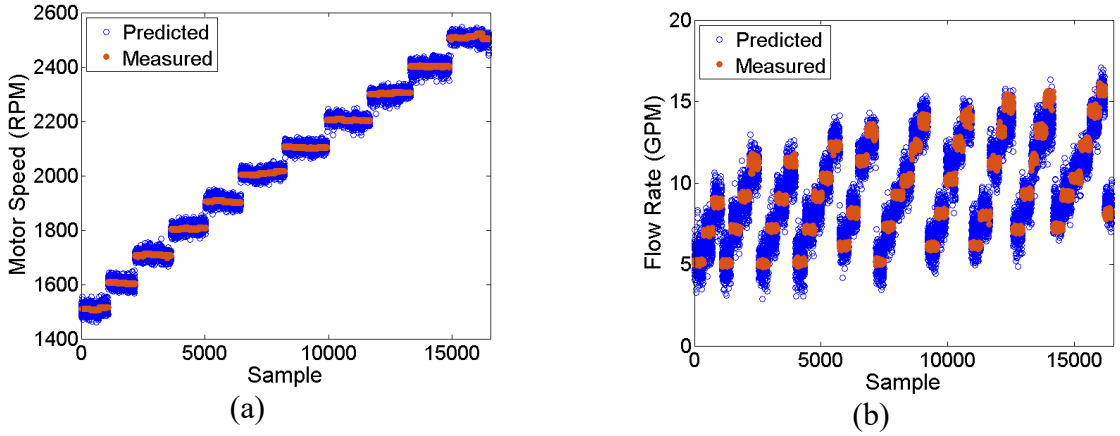


Figure 7. Statistical learning from frequency features: (a) Measured and PLS-predicted motor speed (RPM); (b) Measured and PLS-predicted flow rate (GPM)

Finally, it is important to note that although DNN achieved better prediction performance than PLS, there are several limitations associated with deep learning approaches, which are particularly relevant to industrial applications. In this case, the DNN model has four hidden layers, with each consisting of hundreds to thousands of neurons. Such complex network structure makes model interpretability extremely difficult. Second, DNN has a large number of hyperparameters, which makes model tuning challenging and time consuming. Finally, although the DNN model predicts RPM and GPM well, it does not reveal any fundamental understanding on how the frequency content determines the system output. For industrial applications, model interpretability is crucially importance, as it enables diagnostic inferences and offers key insights for troubleshooting under faulty conditions. On the other hand, although the PLS model performs slightly worse than DNN in terms of prediction performance, there are several advantages of PLS models for industrial applications. These advantages include the robustness of the PLS model under adversarial disturbances or attacks, and straightforward and fast model training with only one tuning parameter. In addition, the significantly simpler model structure and fast computation enables recursive model update. Last but not least, the simple and linear structure enables the model to be easily interpretable. For example, by examining the model coefficients, we can identify the frequencies that contribute the most to the prediction of RPM and GPM.

4.4 Further feature engineering through data visualization and exploration

Because of the simplicity, robustness and interpretability of the PLS models, it is highly desirable to develop a PLS model that could accurately predict the RPM and GPM. In the previous section, when using the whole frequency spectrum as the input to predict RPM and GPM, the prediction performance was improved significantly compared to using the raw vibration data. However, the predictions were still not ideal, especially the flow rate predictions. Figure 5 showed that most of the frequency spectrum are still quite noisy, except the peaks at the low frequency range. The noisy part of the frequency spectrum might be the reason for the noisy prediction generated by the PLS models. If only the informative peaks were included for model building while the irrelevant noisy parts were removed, we would expect to further improve model performance.

But can we identify the relevant information from the whole spectrum? One obvious answer is to use a variable selection algorithm to identify the relevant frequencies, and many methods are available for variable selection for soft sensor development (Lee et al., 2019; Mehmood et al.,

2012; Peres and Fogliatto, 2018; Wang et al., 2015). In this work, we are interested in exploring if knowledge gained by human learning, e.g., through data visualization and exploration, can be used to guide further feature engineering. For this purpose, we examine RPM and GPM modeling separately.

For RPM modeling, it is a common knowledge that for pumps, the motor rotation speed (i.e., RPM) determines the peak locations of the vibration frequency spectrum. This is confirmed by Figure 8, where the spectra with the same RPM but different GPM are grouped together using the same color. Figure 8 shows that the frequency spectra of the same RPM share the same set of peak frequencies, while different GPM's only affect the heights of the peaks. This observation suggests that we might only need the frequency of the dominant peaks to predict RPM. Indeed, when only the frequencies of the dominant peaks were included in the PLS model to predict RPM, the model prediction was further improved, by a large degree, as shown in Figure 9. The RMSE was reduced drastically from 13.16 to 0.21 as shown in Table 2 earlier. The PLS modeling was implemented through a binary matrix that marks the location of the highest peak from each sample spectrum. Details can be found in (Shah, 2019).

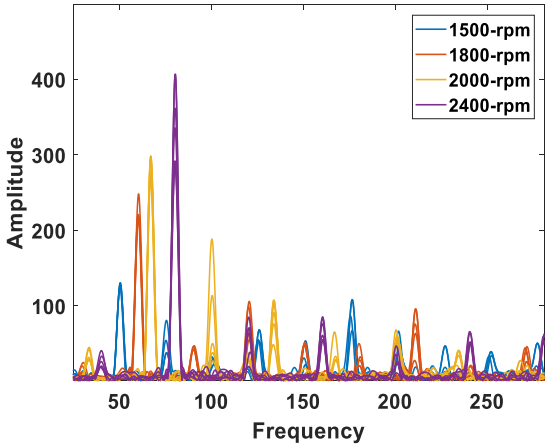


Figure 8. Data visualization and exploration: spectra of the same RPM but different flow rate share the same frequency of the highest peak.

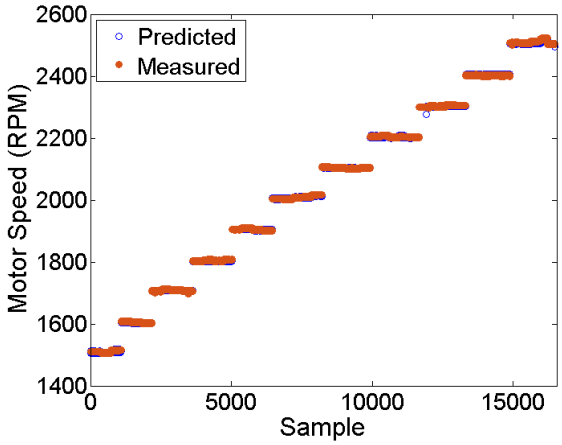


Figure 9. Measured and PLS-predicted RPM when the frequencies of the highest peaks were used as features

To improve GMP prediction, we further explore system knowledge and human learning for feature selection. For the variable speed pump used in this study, depending on the output flow requirement (i.e., the load), the pump usually runs in different stages of RPMs, which are called operation stages. For such operations, we could use an ensemble of models to capture the behavior of different operation stages separately. In other words, we could bin the data that share the same RPM together for GPM modeling. The logic behind the data binning is that by reducing the range of system response covered by the data, a linear PLS model could better approximate the nonlinear relationship between the frequency spectrum and GPM, due to the reduced input space. Since we can predict RPM very accurately, we can identify the operation stage (i.e., RPM) first, then use the corresponding stage model for GPM prediction.

The feasibility of such a data binning approach is confirmed by Figure 10, which provides a zoomed-in view of sample spectra that share the same RPM but different GPM. Figure 10 shows that all sample spectra with the same RPM share the same peak frequency, and different GPM's are reflected in different peak heights. In addition, each sample spectrum only contains small amount of useful information (i.e., the peaks) while most of the spectral segment are noises. Clearly, if the noisy parts can be removed from the input data, the model performance would be further improved.

To isolate the informative peaks that are not drowned by the noises, we propose to use the inverse of the coefficient of variation (ICV) computed from the spectra of all the training samples. For each frequency *f*, the inverse coefficient of variation is defined as $(f) = \frac{mean(A_f)}{var(A_f)}$, where A_f is the amplitude of the spectrum at frequency *f*. If $ICV(f) \geq ICV_{th}$, then frequency *f* would be included in the input for model development. In this work, we set the threshold of ICV as 2.4, which corresponds to 98% confidence level that A_f represents a meaningful peak, instead of random noise. Figure 12 shows the zoomed in view of the ICV at different frequencies for samples from a given condition (2,400 RPM and 7 GPM), for all three directions (x, y and z). The results highlighted the consistency among three directions, and confirmed that majority of the spectrum indeed only contains noise and should not be included as features.

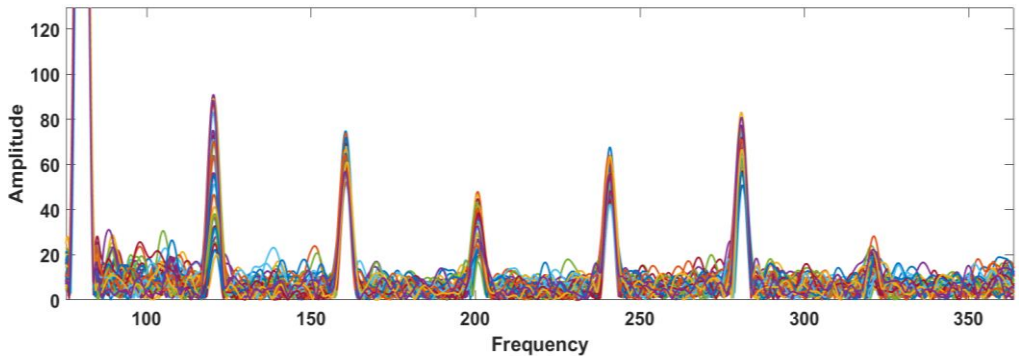


Figure 10. Data visualization and exploration: spectra of the same RPM but different GPM share the same peak locations (i.e., frequencies) but with different peak heights. Besides peaks, the spectra contain significant noises close to the baseline.

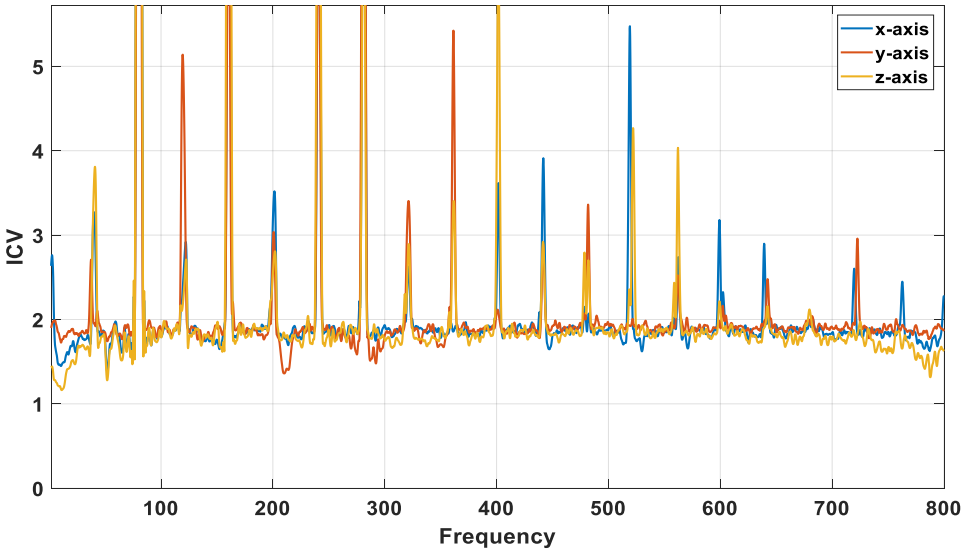


Figure 11. ICV identifies informative features (i.e., the significant and consistent peaks across all training samples of the same conditions)

Next a PLS model was developed to predict GPM using only the selected informative features (i.e., significant and consistent peaks) as input variables for each RPM. The model prediction results are shown Figure 12. It clearly showed that with feature engineering and selection, the prediction performance of the PLS model was significantly improved. Similar to the RPM model, the model predictions agree very well with the experimental measurements. Again, the RMSE was drastically reduced from 0.78 to 0.09 as shown in Table 2 earlier. It is worth noting that the features extracted have clear linear relationships with RPM and GPM, as indicated by the excellent performance of the linear PLS models. Therefore, we did not test any nonlinear statistical modeling approach such as nonlinear PLS. In addition, there are limitations associated with nonlinear PLS. For the methods that use nonlinear transformation to convert an original nonlinear problem into a linear one, such as polynomial PLS, it can be difficult to identify the proper transformations. For kernel PLS, the model becomes a 'black-box' model with limited possibility to interpret the results with respect to the original data.

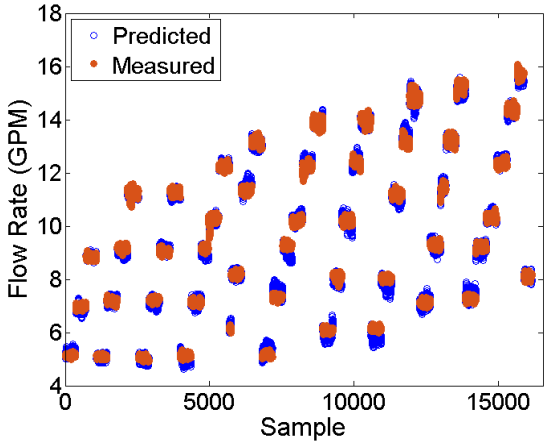


Figure 12. Measured and PLS-predicted flow rate (in GPM) when only consistent peaks across all training samples of the same conditions were included

5 CONCLUSIONS

In this work, we use data collected from an IoT-enabled manufacturing testbed to demonstrate the challenges associated with analyzing IoT big data. In addition, we compared the performances of both complex deep learning and simple statistical learning models with different level/extent of feature engineering in modeling the IoT testbed system. The comparison results highlight the importance of feature engineering and feature selection in developing successful data-driven machine learning models, especially for IoT big data applications. Specifically, we showed that rote application of machine learning, especially deep learning, to raw IoT data yielded erroneous models due to the 4V challenges, especially veracity or the data quality, presented by the IoT big data. In addition, we showed that feature engineering guided by system information and human learning can be a highly effective way to address the 4V challenges of the IoT big data. The robust and informative features generated through feature engineering and selection resulted in accurate models that can be used for process monitoring and control. Finally, our results suggest that for industrial IoT-enabled manufacturing, there are still unsolved challenges for deep learning modeling approaches, including tuning, training and interpretability of DNN with multiple hidden layers and many hyperparameters. In comparison, simple statistical learning approaches, combined with feature engineering, especially that involved somewhat extensive human learning through data visualization and exploration, achieved superior performance. The performance is truly remarkable considering the significant messiness of the IoT big data and the fact that no data cleaning or preprocessing was performed. There are many advantages of simple statistical learning models for industrial applications, including simple tuning, training and easy interpretability. Nevertheless, with DNN's strong capability in modeling nonlinear process behavior, and future advancements in making DNN more interpretable, there is undeniable potential of deep learning for process systems engineering applications. We have witnessed significant advancements in DNN applications for process design and optimization, and we will probably see DNN applications for process operations in the not too distant future.

IoT sensors are based on solid-state technologies such as complementary metal-oxide-semiconductor (CMOS), micro-electro-mechanical systems (MEMS), and nano-optics. In addition, they are produced through cost-efficient semiconductor manufacturing. Therefore, they have many advantages compared to the traditional sensors, such as small size, cheap price, low power consumption, sampling at very high frequency, durable to harsh environment, and can usually be installed non-invasively. As a result, they offer a unique opportunity to instrument manufacturing systems in massive numbers. Although the data quality may present some challenges to the existing data analytics tools, this study demonstrates that these challenges can be addressed with proper feature engineering and selection. Therefore, we envision that IoT sensors will have great potential in many manufacturing applications, such as to achieve improved process monitoring and control.

ACKNOLEDGEMENT

The authors thank the financial support from National Science Foundation under grant CBET #1805950.

REFERENCES

Anderson, M.R., Antenucci, D., Bittorf, V., Burgess, M., Cafarella, M.J., Kumar, A., Niu, F., Park, Y., Ré, C., Zhang, C., 2013. Brainwash: A Data System for Feature Engineering., in: Cidr.

Bengio, Y., Courville, A., Vincent, P., 2013. Representation learning: A review and new perspectives. IEEE Trans. Pattern Anal. Mach. Intell. 35, 1798–1828.

Bergstra, J.S., Bardenet, R., Bengio, Y., Kégl, B., 2011. Algorithms for Hyper-Parameter Optimization.

Dor, O., Reich, Y., 2012. Strengthening learning algorithms by feature discovery. Inf. Sci. (Ny). 189, 176–190.

Evans, R.P., Blotter, J.D., Stephens, A.G., 2004. Flow rate measurements using flow-induced pipe vibration. J. Fluids Eng. 126, 280–285.

He, Q.P., Wang, J., 2018. Statistical process monitoring as a big data analytics tool for smart manufacturing. J. Process Control 67, 35–43. https://doi.org/10.1016/j.jprocont.2017.06.012

He, Q. Peter, Wang, J., 2018. Statistics Pattern Analysis: A Statistical Process Monitoring Tool for Smart Manufacturing, in: Computer Aided Chemical Engineering. https://doi.org/10.1016/B978-0-444-64241-7.50340-2

He, Q.P., Wang, J., 2011. Statistics pattern analysis: A new process monitoring framework and its application to semiconductor batch processes. AIChE J. 57, 107–121. https://doi.org/10.1002/aic.12247

Hochreiter, S., Schmidhuber, J., 1997. Long Short-Term Memory. Neural Comput. 9, 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Hu, J., Peng, H., Liu, T., Yao, X., Wu, H., Lu, P., 2019. A flow sensing method of power spectrum based on piezoelectric effect and vortex-induced vibrations. Measurement 131, 473–481.

Jung, D., Zhang, Z., Winslett, M., 2017. Vibration analysis for iot enabled predictive maintenance, in: 2017 IEEE 33rd International Conference on Data Engineering (ICDE). IEEE, pp. 1271–1282.

Khurana, U., Samulowitz, H., Turaga, D., 2018. Feature engineering for predictive modeling using reinforcement learning, in: Thirty-Second AAAI Conference on Artificial Intelligence.

Khurana, U., Turaga, D., Samulowitz, H., Parthasrathy, S., 2016. Cognito: Automated feature engineering for supervised learning, in: 2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW). IEEE, pp. 1304–1307.

Lannes, D.P., Gama, A.L., Bento, T.F.B., 2018. Measurement of flow rate using straight pipes and pipe bends with integrated piezoelectric sensors. Flow Meas. Instrum. 60, 208–216.

Lee, J., Flores-Cerrillo, J., Wang, J., He, Q.P., 2019. Consistency-enhanced evolution for variable selection can identify key chemical information from spectroscopic data. Ind. Eng. Chem. Res. submitted, submitted.

Lomb, N.R., 1976. Least-squares frequency analysis of unequally spaced data. Astrophys. Space Sci. 39, 447–462.

Manyika, J., Chui, M., Brown, B., Bughin, J., Dobbs, R., Roxburgh, C., Byers, A.H., Institute,

M.G., 2011. Big data: The next frontier for innovation, competition, and productivity.

Markovitch, S., Rosenstein, D., 2002. Feature generation using general constructor functions. Mach. Learn. 49, 59–98.

Mehmood, T., Liland, K.H., Snipen, L., Saebo, S., 2012. A review of variable selection methods in Partial Least Squares Regression. Chemom. Intell. Lab. Syst. 118, 62–69.

Ng, A., 2013. Machine Learning and AI via Brain simulations. Accessed May 3, 2018.

Nielsen, M.A., 2015. Neural Networks and Deep Learning.

Peres, F.A.P., Fogliatto, F.S., 2018. Variable selection methods in multivariate statistical process control: A systematic literature review. Comput. Ind. Eng. 115, 603–619.

Rendall, R., Lu, B., Castillo, I., Chin, S.-T., Chiang, L.H., Reis, M.S., 2017. Profile-driven Features for Offline Quality Prediction in Batch Processes, in: Computer Aided Chemical Engineering. Elsevier, pp. 1501–1506.

Rumelhart, D.E., Hinton, G.E., Williams, R.J., 1986. Learning representations by back-propagating errors. Nature 323, 533–536.

Shah, D., 2019. Big Data Analytics and Its Applications in Soft Sensor for Smart Manufacturing. Auburn University.

Shah, D., Wang, J., He, Q.P., 2019a. An Internet-of-things Enabled Smart Manufacturing Testbed, in: IFAC-PapersOnLine. Florianopolis, Brazil, pp. 562–567. https://doi.org/10.1016/j.ifacol.2019.06.122

Shah, D., Wang, J., He, Q.P., 2019b. A feature-based soft sensor for spectroscopic data analysis. J. Process Control 78, 98–107.

Suthar, K., Shah, D., Wang, J., Peter He, Q., 2018. Feature-based Virtual Metrology for Semiconductor Manufacturing, in: Computer Aided Chemical Engineering. https://doi.org/10.1016/B978-0-444-64241-7.50342-6

Venkatasubramanian, V., 2019. The promise of artificial intelligence in chemical engineering: Is it here, finally. AIChE J 65, 466–478.

Wang, J., He, Q.P., 2010. Multivariate Statistical Process Monitoring Based on Statistics Pattern Analysis. Ind. Eng. Chem. Res. 49, 7858–7869. https://doi.org/10.1021/ie901911p

Wang, R., Edgar, T.F., Baldea, M., 2017. A geometric framework for monitoring and fault detection for periodic processes. AIChE J. 63, 2719–2730.

Wang, S., Aggarwal, C., Liu, H., 2017. Randomized feature engineering as a fast and accurate alternative to kernel methods, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, pp. 485–494.

Wang, Z., He, Q.P., Wang, J., 2015. Comparison of variable selection methods for PLS-based soft sensor modeling. J. Process Control 26, 56–72. https://doi.org/10.1016/j.jprocont.2015.01.003

Wind, D.K., 2014. Concepts in predictive machine learning. Tech. Univ. Denmark 1–129.

Wold, S., Kettaneh-Wold, N., MacGregor, J.F., Dunn, K.G., 2010. Batch Process Modeling and MSPC, in: Brown, S., Tauler, R., Walczak, B. (Eds.), Comprehensive Chemometrics.

Elsevier, pp. 163–197. https://doi.org/10.1016/B978-044452701-1.00108-3

Zhong, R.Y., Ge, W., 2018. Internet of things enabled manufacturing: A review. Int. J. Agil. Syst. Manag. 11, 126–154.