Application-Driven Sensing Data Reconstruction and Selection Based on Correlation Mining and Dynamic Feedback

Zhichuan Huang, Tiantian Xie, Ting Zhu, Jianwu Wang and Qingquan Zhang
University of Maryland, Baltimore County
{zhihu1, xtiant1, zt, jianwu, q}@umbc.edu

***Abstract*— As sensors spread across almost every industry, the Internet of Things (IoT) is going to trigger an era of big data. However, the abundance of available sensing data causes new challenges when building IoT applications. One main challenge is how to select proper data from large amount of sensing data for learning useful information efficiently. Existing approaches require developers to manage data for each specific application, which is very time consuming since the developers may not have enough knowledge about the dynamic changing data quality of different sensors. In this paper, we propose a data management middleware to learn the correlations between time series sensor data without prior knowledge. The learned correlation is then applied to select the useful sensor and reconstruct the incorrect data. To generalize the correlation models for each application, we utilize the dynamic feedback from the application to update the data selection and reconstruction. We evaluate our data management middleware in smart grids. The evaluation results show that our middleware can achieve better application performance with the help of dynamic feedback, data reconstruction and data selection.**

I. Introduction

More and more sensors are being deployed in the environment to provide sensing data to support various Internet of Things (IoT) applications. With these millions of IoT devices deployed in the environment and connected to Internet, the enormous volumes of sensor data will be generated, and trigger an era of big data for IoT [3]. Many IoT applications are built using a data-driven approach: try to deploy as many sensors as possible and utilize as much available data as they can. However, the challenges of more sensing data sometimes overweighs the benefits, especially when there is no clear indication on the usefulness of a type of sensor data for a specific application. More sensor and sensor data normally increases the budgetary cost of the application and complexity of data processing in the application. More troublingly, we have found that the enormous volumes of data from a large amount of sensors may lead to less correct or even incorrect information due to overfitting problem of learning models [13]. Therefore, in such an IoT big data era, it is increasingly critical and challenging to find the relevant and reliable sensing data that fits specific applications, which is also a core problem faced by most IoT applications.

To cope with the massive and unreliable sensor data in the IoT, we propose an application driven data management middleware between sensor data and applications. The middleware will dynamically learn the most relevant data for the application from the large amount of raw sensor data and reconstruct missing or incorrect data to be used in the application. The middleware can be used to not only build better applications by dynamically finding right types of sensor data, but also help application developers to decide which sensors are actually useful for their applications.

This paper focus on two challenges of building such a data management middleware. One challenge is how to have a generic approach that is able to identify the usefulness and correctness of each sensor data for different applications. To address this challenge, we use temporal and spatial correlations among different sensor data, which are learned without prior knowledge to select the useful sensor data and reconstruct missing or unreliable data. Another challenge is that the data requirements of an application at different time are varying due to the nature of time series sensor data. To address this challenge, we apply feedback based data management, which will learn from the performance evaluation of the application to obtain feedback. Then the feedback can be further applied to update the correlations among different sensor data. Specifically, we summarize our contributions as follows:

- We propose a data management middleware to find useful sensor data and reconstruct unreliable data for application. The temporal and spatial correlations are learned from sensor data without prior knowledge. The learned correlations are then applied to find the useful sensor data and reconstruct the data with low quality.
- To generalize our data management middleware for data reconstruction and selection, we utilize the dynamic feedback of different applications based on the performance of application. The dynamic feedback is then applied to update the data reconstruction and selection based on data correlation models.
- We evaluate our data management middleware using a case study in smart grids. The evaluation results show that our middleware can achieve better application performance with the help of dynamic feedback, data reconstruction and data selection.

The rest of the paper is organized as follows. The system overview is introduced in §2; detailed system design, and evaluation are provided in §3, §4, respectively; related work is discussed in §5; finally, we conclude our paper in §6.

II. Design Overview

To provide a generic sensing data management middleware for IoT applications, we propose the system design as shown in Figure 1, which includes three main components: sensing data reconstructor, sensor data selector, correlation mining and feedback controller. In summary,

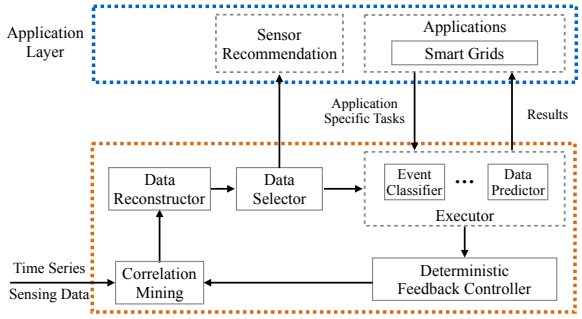


Figure 1: Design Overview

our system works as follows: (i) the data reconstructor identify unreliable data and reconstruct their values; (ii) the data selector find the relevant data for application and provide sensor recommendation for application layer; (iii) the correlation mining then takes the relevant data to learn the correlations among the data and provide the correlation levels for the executor; (iv) the executor takes in the relevant data, correlations and the application specific goal from different applications to obtain the results for application; (v) the results of the executor are then evaluated by feedback controller to identify the correlation between sensor data and application performance and generate feedback for data reconstructor to correct data and for data selector to update the relevant data over time.

• **Correlation Mining.** To reconstruct and select the sensing data, we investigate the correlation models among data. Specifically, we investigate i) temporal correlation; ii) correlation between different types of the sensors; and iii) correlation between the same type of sensors deployed in different places.

• **Data Reconstructor.** To deal with unreliable data problem, the data reconstructor uses the feedback from performance evaluation to identify the unreliable data in real time and reconstruct their values based on mined correlations.

• **Data Selector.** To deal with data selection problem, the data selector is introduced to find the subset of the data that is relevant to the specific application. The key idea is to utilize the feedback from performance evaluation to learn the relevance between the data and the performance evaluation of the application.

• **Executor.** The executor runs application specific tasks based on reconstructed and selected data and mined correlations. Each IoT application has its own specific goals and tasks, such as which values to predict and which events to classify. These tasks will be able to run within our executor without any modification. We believe our executor can have better or similar performance by reconstructing unreliable data and only selecting relevant sensing data. Executor results will be sent to both feedback controller and application itself.

• **Feedback Controller.** The feedback controller interpret the relationship between the selected data and the application performance using the selected data. For deterministic applications, we investigate the correlation between data errors and performance of the application to provide feedback for the data selector to find the relevant data for the application.

III. Data Management Components

In this section, we will introduce the detailed design of our data management middleware. First, we present how the temporal and spatial correlations among time series sensor data can be learned without prior knowledge and how to evaluate the Pearson correlation coefficient among time series data. Then, we explain how our data reconstructor and data selector can be used to correct misreading data, find the relevant data and recommend sensors for application layer based on the learned correlations. Finally, we elaborate how the feedback controller can utilize the performance evaluation to provide dynamic feedback.

A. Data Correlation Mining

In this paper, we use correlation (or, similarity) to discover groups of objects with similar behaviors and discover potential signatures of time series sensor data by correlations.

For correlation notation, we use lowercase bold letters for column vectors $(\mathbf{x}, \mathbf{y}, \cdots)$ and uppercase bold for matrices $(\mathbf{X}, \mathbf{Y}, \cdots)$. The Euclidean norm of $\mathbf{x}$ is $||\mathbf{x}||$ and the size of $\mathbf{x}$ is $|\mathbf{x}|$. We denote a time series sensor data $\mathbf{x}$ as an indexed collection of random variables X, $\mathbf{x} = \{X(1), X(2), \cdots, X(t), \cdots\}$. In this paper, we use $\mathbf{x}(i, j)$ to represent the sequence $\{X(i), \cdots, X(i+j-1)\}$, which starts with $X(i)$ with length of j. The covariance of two random variables X, Y is defined as $\mathrm{Cov}[X, Y] = \mathrm{E}[(X - \mathrm{E}[X])(Y - \mathrm{E}[Y])]$. If $\{X_1, X_2, \cdots, X_m\}$ is a group of m random variables, their covariance matrix $\mathbf{C} \in \mathbf{R}^{m \times m}$ is the symmetric matrix defined by $c_{ij} = \mathrm{Cov}[X_i, X_j], 1 \leq i, j \leq m$. To quantify the correlation between two sequences of data, we use Pearson correlation in this paper. The Pearson correlation between two sequences of data $\mathbf{x}_i$ and $\mathbf{x}_j$ can be calculated as:

$$\rho_{ij} = \frac{|\mathrm{Cov}[\mathbf{x}_i, \mathbf{x}_j]|}{\mathrm{Var}[\mathbf{x}_i]\mathrm{Var}[\mathbf{x}_j]} \quad (1)$$

$\mathrm{Cov}[\mathbf{x}_i, \mathbf{x}_j]$ is the correlation between sequences $\mathbf{x}_i$ and $\mathbf{x}_j$. $\mathrm{Var}[\mathbf{x}_i]$ and $\mathrm{Var}[\mathbf{x}_i]$ are the variances of the two sequences. The Pearson correlation of two vectors calculated through Equation (1) is ranging from 0 to 1, and higher value means that the correlation between two vectors is stronger. In the following sections, we will introduce how to identify the temporal and spatial correlations in time series sensor data and how these correlations can be learned without prior knowledge.

1) Temporal Correlation: Since a type of time series sensor data is usually collected by monitoring a certain event, if there is no temporal correlation within a time series sensor data, it is unlikely that the sensor data will be useful for IoT applications. If more than two partial sequences of data have strong correlations among them, we then denote these

sequences as one signature of the data. These signatures are the unique information of the time series sensor data and can be very useful for IoT applications. Therefore, we design a temporal signature detection algorithm on the data set.

2) Spatial Correlation: Spatial correlation is another type of correlations that widely exist in time series sensor data since the same type of sensors may be deployed in different locations to monitor the same event. For example, weather conditions within certain geographical areas are usually very similar. In this case, one sensor data can be potentially applied to reconstruct the other sensor data.

To accurately capture the correlation between two time series sensor data, we need a time-evolving correlation model that should be able to capture time-evolving correlations between two time series sensor data. An intuitive solution is to calculate the Pearson correlation of two sequences. However, this correlation model of time series is too simple and has two problems: (i) it cannot capture more complex relationships, and (ii) it is too sensitive to transient changes, often leading to very dynamically changing correlation values, which is not useful for applications. To address these two problems, for two sequences data with window size w, we not only calculate the correlation between these two sequences but also the correlation of sequences that are near t. In detail, we will consider exponential window: we use all the windows before t, specifically $\mathbf{x}_i(\tau, w)$ for $t-k+1 \leq \tau \leq t$, and we weigh those sequences that close to t more, by multiplying each window by a factor of β.

Given a time series $\mathbf{x}_i$, the local correlation estimator $\rho^e_{ij}(t, w)$ of two time series data $\mathbf{x}_i$ and $\mathbf{x}_j$ at time t using a exponential window is defined as:

$$\rho^e_{ij}(t, w) = \sum_{\tau=1}^{t} \rho_{ij}(\tau, w)\beta^{t-\tau} \tag{2}$$

B. Data Reconstruction and Selection

With the more and more collected sensor data, the complexity of data processing will increase exponentially. Meanwhile, more data may bring more errors and cause overfitting problem. In this section, we first introduce how to utilize the correlation mining results to reconstruct missing readings. Then, we learn the relevance of the data for data selection.

1) Data Reconstruction: Based on our experiences and other work, the sensor data collection may suffer from missed readings and unreliable readings. In Sections III-A1 and III-A2, we show that how temporal and spatial correlation among different sensor data can be learned. In this section, we present how utilize these correlations to reconstruct the sensor data. To utilize these correlations, we first propose to define the temporal correlation score $P_i(t, w)$ for a sequence data $\mathbf{x}_i(t, w)$ as:

$$P_i(t, w) = \sum_{\tau=t}^{w} \mathbf{p}_i(\tau)/w \tag{3}$$

$\mathbf{p}_i(\tau)$ is the Pearson correlation between the data and signatures. $\mathbf{x}_i(t, w)$ has strong temporal correlation when $P_i(t, w)$ is close to 1 and weak temporal correlation when $P_i(t, w)$ is close to 0. For spatial correlation score, it can be directly obtained by calculating their Pearson correlation with exponential window based on Equation 2. With the temporal and spatial correlation models, we can reconstruct the missing data as follows:

$$\bar{\mathbf{x}}_i(t, w) = \hat{P}_i(t, w)\mathbf{s}_i + \sum_{k=1}^{m} \mathbf{x}_k(t, w)\hat{\rho}^e_{ik}(t, w) \tag{4}$$

$\hat{P}_i(t, w)$ is the estimated temporal correlation score between the signature and missing data and $\hat{\rho}^e_{ik}(t, w)$ is the estimated Pearson correlation with exponential window. Due to the missing data, we are not able to get $P_i(t, w)$. However, we can estimate $\hat{P}_i(t, w)$ based on known sensor data sequence $\mathbf{x}_i(t-w, w)$. The estimated temporal correlation score can be calculated as:

$$\hat{P}_i(t, w) = \max_{\tau \in [1,w]} P_i(t-\tau, w) \tag{5}$$

Also, we can estimate $\hat{\rho}^e_{ik}(t, w)$ based on known sensor data sequences $\mathbf{x}_j(t-w, w), j \neq i$. The estimated spatial correlation can be calculated as:

$$\hat{\rho}^e_{ik}(t, w) = \sum_{\tau=1}^{w} \rho^e_{ik}(t-\tau, \tau)/w \tag{6}$$

Finally, the reconstructed data $\bar{\mathbf{x}}_i(t, w)$ can be calculated based on estimated $\hat{P}_i(t, w)$ and $\hat{\rho}^e_{ik}(t, w)$.

2) Data Selection: With the correlation between sensing data and performance, we then need to translate these correlation into sensor/data selection. If a time series sensor data show strong temporal correlation and have specific patterns, then the time series sensor data should have higher probability to be selected in later data selector and has higher impact in the executor. In the meanwhile, if two time series sensor data have strong spatial correlation, it means that there are containing similar information that is redundant, thus we can only select one of them in data selector. Therefore, in this paper, we use the temporal and spatial correlation to decide which time series sensor data is more important to the application. A naive solution is select a group of the sensor data with highest temporal correlation. However, each sensor data are also spatially correlated, if multiple sensor data has strong spatial correlation, then it is highly possible that they all have high temporal correlations. Thus, in this case, the sensor data in the selected group will be very similar and may not be able to show all the information contained in these sensor data. To avoid this scenario, we propose a data selection algorithm based on temporal and spatial correlation among sensor data and the detailed process is shown in Algorithm 1.

Algorithm 1 Data Selection Algorithm

Input: $\mathbf{x}_1(t,w), \cdots, \mathbf{x}_m(t,w)$
Output: $\mathbf{C}_2$
$\mathbf{C}_1 = \{\mathbf{x}_1(t,w), \cdots, \mathbf{x}_m(t,w)\}$, $\mathbf{C}_2 = \emptyset$;
while $\mathbf{C}_1 \neq \emptyset \vee P_i(t,w) > c_i$ **do**
  Fetch the data with highest $P_i(t,w)$;
  for every data sequences in $\mathbf{C}_2$ **do**
    if $\rho_{ij}(t,w) < \rho_{th}$ **then**
      Update $\hat{\mathbf{x}}_i(t,w)$;
      Remove $\mathbf{x}_i(t,w)$ from $\mathbf{C}_1$ and add $\hat{\mathbf{x}}_i(t,w)$ to $\mathbf{C}_2$;
    end if
  end for
end while

C. Feedback Controller

The feedback controller in our data management middleware will learn from the performance of the application to generate feedback for data reconstructor and selector. With the correlation between data errors and performance, the feedback will be transformed into data requirements for data selection process. In this paper, we need to decide what probability of different correlations is related to the application performance change. For deterministic applications, since the performance of application is best when the sensor data is 100% correct. Thus, we can investigate the correlation between data errors and performance difference to understand the data requirement of the application. In details, the deterministic feedback can be dynamically updated with the following steps:

Step 1: The application layer provides feedback error, $e(t)$, from the application layer, the the system calculates the $(\Delta e(t))$ and store it into a FIFO queue $Q(t)$ of size K.

Step 2: After the feedback, the program will calculate the changes of errors, $\alpha(t)$, using the following function:

$$\alpha(t) = \sum_{i=0}^{K-1} \Delta e(t-i) - \sum_{i=1}^{K} \Delta e(t-i) \tag{7}$$

Step 3: Based on the error $\alpha(t)$, input them into a function to find a parameter $\beta(t)$, which is used to increase the correlation threshold that filters out unrelated features if their correlation is less than the threshold.

$$\beta(t) = 1 - 0.01 * tanh(\alpha(t)) \tag{8}$$

where $tanh()$ is the hyperbolic tangent function, which is the hyperbolic analogue of the Tan circular function used throughout trigonometry.

Step 4: If $\alpha(t) > 0$ (positive errors) which means the error is increasing. Then we will apply $\beta(t)$ to the correlation threshold $c_i(t)$ of those data not currently selected to increase their chance of being selected.

Step 5: If $\alpha(t) < 0$ (negative errors), which means the error is decreasing. Then we will apply $\beta(t)$ to the correlation threshold $c_i(t)$ of those data selected to decrease their chance of being selected.

Step 6: In order to remove the zigzagging effect of threshold, when $\alpha(t) < 0$, the correlation threshold $c_i(t)$ will be increased more slowly. which means, $c_i(t) = c_i(t-1) * (\beta(t) + d)/(d+1)$, where $d \geq 1$ and is constant.

Step 7: The correlation threshold $c_i(t)$ is then applied to each data to select data of high correlation and send them to the application.

With the above steps, the correlation threshold $c_i(t)$ used for data selection will be updated based on the application performance. Note that the data with higher correlation is not always selected since we need to consider the data redundancy, which is evaluated by spatial correlation.

IV. Experimental Evaluations

To evaluate our proposed data management middleware, we conduct extensive experiments with one case study in smart grids. In this case study, we consider a smart grid, illustrated in Figure 2, consists of generation technology (e.g., local electricity generators). To ensure compatibility with the traditional power grid, we adopt the architecture, which is similar to the one used in a traditional power grid. Within the smart grid, sensors are deployed in each home to collect energy related data (e.g., power, voltage and frequency) and send to controller. The controller then decides the generations to balance the power demand and supply in real-time based on energy consumption in next few hours. However, based on four years' experiences of energy monitoring, the sensor reading can be missed and unreliable, thus, the data management middleware proposed in our paper will be applied to the collected raw data and provide reconstructed and selected data for demand prediction.

We deploy eGauge power meters at individual homes to collect the energy consumption data every minute. In our simulation, we use the power consumption traces that we collected from 726 homes (located either in Texas, Massachusetts or California) for more than one year. An example of deployment is shown in Figure 3(a). To make the figure easy to follow, we only show the aggregated power consumption for 12 months in Figure 3(b). It can be found that the power consumption for different days varies significantly and is higher in summer while lower in winter.

Baselines. We compare two baselines in the evaluations: i) Data management without data selection (WoDS), in which we use all the sensor data to perform application specific tasks; and ii) Data management with data selection (WDS) but no dynamic feedback, which is used to show the performance gain by data selection. In all the experiments, our propose data management middleware is referred as data selection with dynamic feedback (WDS + FB).

Metrics. Because we predict the aggregated power consumption in smart grid. Thus, we use two metrics to evaluate the performance of our approach: i) **MAPE (Mean Absolute Percentage Error)** and ii) **RSME (Root Square Mean Error)**. MAPE and RMSE both measure the differences between outputs compared with ground truths. The lower the measure, the closer the methods outputs to the ground truths, and the better it performs. RMSE emphasizes on larger errors compared with MAPE.

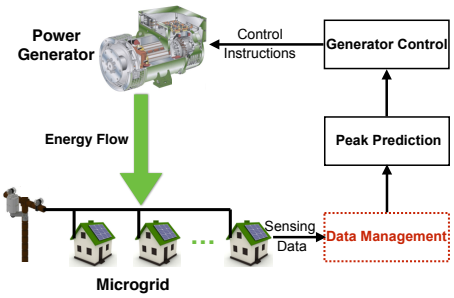
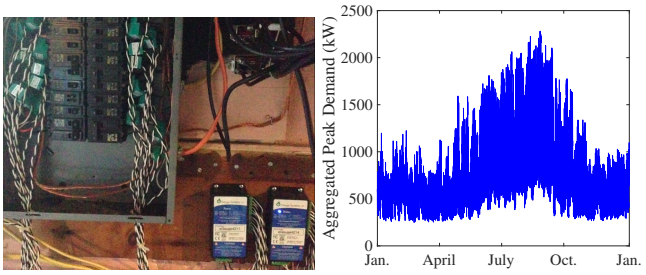


Figure 2: Architecture of A Smart Grid

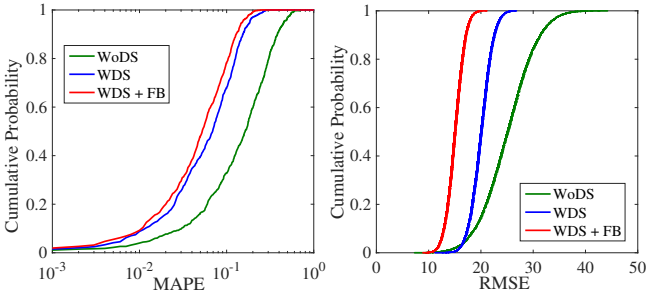


(a) Energy Monitoring (b) Aggregated Peak Demand

Figure 3: Sensor Deployment and Data Collection

1) Evaluation Results: **Overall Performance.** The MAPE and RMSE of demand prediction are shown in Figure 4(a) and 4(b) respectively. For aggregated demand prediction, we utilize both temporal and spatial correlation among energy consumption of all individual homes. In our experiment, WoDS utilizes the spatial correlations of all 726 homes, WDS utilize their correlation to select the homes that have strong correlation with the aggregated demand. For WDS + FB, the prediction error is then used for adjust the weight for the homes that are used for demand prediction. From Figure 4(a), WDS and WDS+FB have much lower (38.9% and 47.2%) MAPE . Because MAPE is calculated with the average error percentage, the difference of MAPE between WDS and WDS+FB are relative small. However, in Figure 4(b), we can find the performance of WDS+FB is much better (24.9% lower RMSE) than WDS.

Dynamic Feedback. To find out why dynamic feedback can provide better demand prediction accuracy, we show a 24 hour prediction results of ground truth, WDS and WDS+FB in Figure 5. From hour 2 to hour 3, the aggregated demand increases significantly from $508kW$ to $1368kW$.



(a) CDF By MAPE (b) CDF By RMSE

Figure 4: Aggregated Demand Prediction

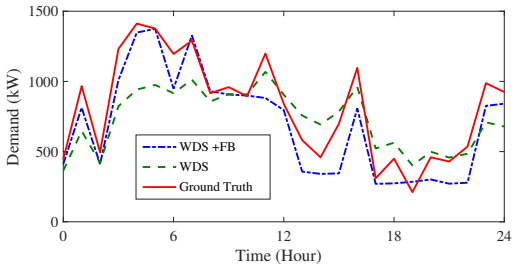
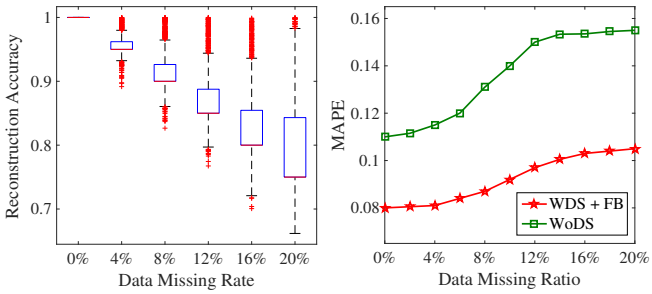


Figure 5: Benefit with Dynamic Feedback



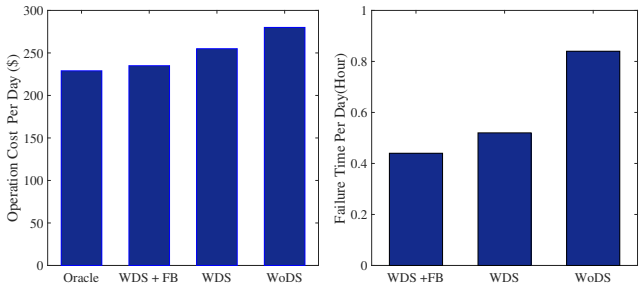
(a) Data Reconstruct Accuracy (b) Prediction with Missing Data

Figure 6: Impact of Missing Data

For WDS, since it does not have the dynamic feedback for real-time prediction accuracy, it can not follow the sudden increase of the demand. While for WDS+FB, though the prediction accuracy is only 76% due to the sudden increase at hour 3, the prediction accuracy is 94.5% for hour 4 with dynamic feedback. The rest part of the figure show the similar phenomenon. To predict sudden high increase or decrease, WDS+FB can achieve much better prediction accuracy than WDS.

Missing Data. In these sets of simulations, we simulated the accurate data to generate missing data with different missing rate from 4% to 20%. The results of reconstruction accuracy are shown in Figure 6(a). With the increase of data missing rate, reconstruction accuracy decreases slowly. Even with 20% data missing rate, the average of reconstruction accuracy is above 80%. Thus, our design is robust to high data missing rate. With the reconstructed data, we evaluate the performance of our design with different data missing rate. The results are shown in Figure 6(b). The X-axis is the percentage of missing data; and Y-axis is the MAPE value of prediction results. We can find that the prediction accuracy decreases with higher missing data rate.

Impact on Generation Control. With different demand prediction accuracy, we also evaluate how different prediction accuracy will impact on the performance of generation control. Because in our simulation, the local generator will be the sole energy source for the smart grid, if the generation is higher than demand, then the extra energy will be wasted, which introduce high operation cost. If the generation is lower than demand, then some of the homes will be out of power for that period. In this simulation, we apply simple generation control algorithm to generate the same amount



(a) Operation Cost (b) Failure Time

Figure 7: Impact on Generation Control

of the energy that are predicted to be consumed in next hour. The operation cost and failure time per day are shown in Figure 7(a) and 7(b). For operation cost, because WoDS has lowest prediction accuracy, thus the operation cost will be highest while WDS+FB with highest prediction accuracy has the lowest operation cost. For average failure time, since ground truth will always fulfill the energy demand, there would be no failure time. Because WDS+FB can predict demand more accurately, the failure time can be reduce by 47% compared to WoDS. Therefore, with our data management middleware, the generation scheduler can minimize the operation cost and failure time.

V. Related Work

Our work is related to the areas of sensor data collection and management:

Sensor Data Collection. With the growing deployed sensor networks, more and more data needs to be collected from the sensors [1, 6, 11]. Many works have focused on developing new technologies for data collection in sensors [7, 10]. In the meantime, since there are normally limited energy storage in sensors, many works are aim to reduce the power consumption of individual sensors by utilizing proper data management techniques and minimizing the transmission time of wireless communication [12, 15]. Our research seeks to expand upon the existing works and we are able to detect and reconstruct unreliable sensor data at software level in real-time based on data correlation mining.

Sensor Data Management. With the rapid increasing collected sensor data, it is important to select proper sensor data for the specific application [4, 5, 9]. Algorithms are proposed to find the best balance for a particular application between data latency, sensor energy use, and data accuracy [2, 14]. Many frameworks for managing the growing number of IoT devices are proposed to make the collected data more accessible for everyday use [8]. Different from existing data management in different applications, our middleware are generic and can work with different types of applications.

VI. Conclusion

Different from many approaches that are data-driven and try to use as much available data as possible, in this paper, we try to employ an application-driven approach that select data only if it is relevant to the application. To achieve this goal, we propose a data management middleware to learn the correlations between time series data from sensors without prior knowledge. To generalize the correlation models for different applications, we utilize the adaptive feedback from each application to reconstruct unreliable incoming sensor data in real-time. Then, the learned correlations and reconstructed data are used to select sensor data that are relevant to the application. The evaluation results show that our middleware can achieve better application performance with data reconstruction, selection and dynamic feedback.

VII. Acknowledgement

This project is supported by NSF grant CNS-1503590 and UMBC COEIT Strategic Plan Implementation Grant.

References

[1] Y. Chen, D. Lymberopoulos, J. Liu, and B. Priyantha. Indoor localization using fm signals. *IEEE Transactions on Mobile Computing*, 12(8):1502–1517, 2013.

[2] D. Chu, N. D. Lane, T. T.-T. Lai, C. Pang, X. Meng, Q. Guo, F. Li, and F. Zhao. Balancing energy, latency and accuracy for mobile sensor data classification. In *SenSys*, 2011.

[3] W. Fan and A. Bifet. Mining big data: current status, and forecast to the future. *ACM sIGKDD Explorations Newsletter*, 14(2):1–5, 2013.

[4] P. Giridhar, M. T. Amin, T. Abdelzaher, L. Kaplan, J. George, and R. Ganti. Clarisense: Clarifying sensor anomalies using social network feeds. In *PERCOM*, 2014.

[5] Z. Huang, H. Luo, D. Skoda, T. Zhu, and Y. Gu. E-sketch: Gathering large-scale energy consumption data based on consumption patterns. In *IEEE International Conference on Big Data*, 2014.

[6] Z. Huang, T. Zhu, H. Lu, and W. Gao. Accurate power quality monitoring in microgrids. In *IPSN*, 2016.

[7] W. Kleiminger, C. Beckel, and S. Santini. Household occupancy monitoring using electricity meters. In *UbiComp*, 2015.

[8] C.-J. M. Liang, B. F. Karlsson, N. D. Lane, F. Zhao, J. Zhang, Z. Pan, Z. Li, and Y. Yu. Sift: Building an internet of safe things. In *IPSN*, 2015.

[9] L. Nachman, R. Kling, R. Adler, J. Huang, and V. Hummel. The intel reg; mote platform: a bluetooth-based sensor network for industrial monitoring. In *IPSN*, 2005.

[10] S. Nawaz, C. Efstratiou, and C. Mascolo. Smart sensing systems for the daily drive. *IEEE Pervasive Computing*, 15(1):39–43, 2016.

[11] M. Philipose, J. R. Smith, B. Jiang, A. Mamishev, S. Roy, and K. Sundara-Rajan. Battery-free wireless identification and sensing. *IEEE Pervasive Computing*, 4(1):37–45, 2005.

[12] K. Romer and B. C. Renner. Aggregating sensor data from overlapping multi-hop network neighborhoods: Push or pull? In *INSS*, 2008.

[13] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[14] L. Su, S. Hu, S. Li, F. Liang, J. Gao, T. F. Abdelzaher, and J. Han. Quality of information based data selection and transmission in wireless sensor networks. In *RTSS*, 2012.

[15] N. Trigoni, Y. Yao, A. Demers, J. Gehrke, and R. Rajaraman. Wave scheduling and routing in sensor networks. *ACM Trans. Sen. Netw.*, 3(1), 2007.