SECOE: Alleviating Sensors Failure in Machine Learning-Coupled IoT Systems

Yousef AlShehri
School of Computing
University of Georgia
Athens, United States
yousefAlshehri@uga.edu

Lakshmish Ramaswamy
School of Computing
University of Georgia
Athens, United States
laksmr@uga.edu

Abstract—Internet of Things (IoT) domains are characterized by continuous streams of data originating from diverse, geographically distributed sensors. Sensor/network failures that result in data stream interruptions is a major challenge in applying ML techniques to IoT domains. Unfortunately, the performance of many ML applications quickly degrades when faced with data incompleteness. With the aim of building robust IoT-coupled ML applications, this paper proposes SECOE — a unique, proactive approach for alleviating potentially simultaneous sensor failures. The fundamental idea behind SECOE is to create a carefully chosen ensemble of ML models in which each model is trained assuming a set of failed sensors. SECOE includes a novel technique to minimize the number of models in the ensemble by harnessing the correlations among sensors. We demonstrate the efficacy of the SECOE approach through a series of experiments involving two distinct datasets.

Keywords—Sensor Failure, Robust Machine Learning, Machine Learning Ensemble.

I. INTRODUCTION

The Internet of Things (IoT) paradigm is transforming many domains of human endeavor, such as agriculture, healthcare, physical infrastructure management, manufacturing, and transportation [1]. IoT domains are characterized by high-velocity data streams originating from diverse devices. This provides the opportunity to build innovative Big Data applications by constructing models that harness historical and current data, thereby gaining valuable insights that facilitate better decision-making. In this context, there is significant research interest in applying machine learning (ML) techniques for IoT domains. Indeed, many ML applications have already been built for IoT data.

However, IoT environments possess several unique characteristics that pose significant challenges for building effective ML-based applications. One such challenge is the possibility of sudden and unexpected interruption of data streams due to IoT-sensor or communication failures resulting in inference-time data incompleteness. In face of such failures, ML applications are required to make inferences based on incomplete/partial data. Unfortunately, many ML algorithms fail when faced with inference-time data incompleteness in the sense that their performance degrades drastically even when a small fraction of sensors fail.

Our research aims to design ML-based techniques that are robust against sensor/network failure-induced inference-time data incompleteness. Several previous researchers have worked on overcoming missing data of sensors [2, 3, 4, 5, 6]. Most of the existing techniques focus on effective imputation techniques to fill the missing values of failed sensors. These imputation-based strategies, however, are not effective when faced with simultaneous failures of multiple sensors because they often suffer from the out-of-distribution (OOD) problem, where the new record produced by applying imputation deviates highly from the model’s training data distribution.

Towards enhancing the robustness of ML-coupled IoT systems, this paper presents SECOE — a novel approach to proactively alleviate sensor failure-induced inference-time data incompleteness. SECOE is fundamentally distinct from imputation-based approaches. The basic idea of SECOE is to create an ensemble [7] of ML models in which each model is specifically trained with an incomplete dataset representing a set of failed sensors. In other words, specific sensor feeds are omitted from training sets in anticipation of potential failures of corresponding sensors during inference time.

In designing SECOE, this paper makes the following unique technical contributions. First, we present the architecture of SECOE that includes novel techniques for ensemble creation and inference-time optimization, thereby demonstrating how the ensemble techniques can be harnessed for mitigating sensor failures in IoT systems. Second, to improve training and inference efficiency, we present an algorithm that minimizes the sub-models in the ensemble. Our technique forms sensor groups based on inherent correlations among sensors in an IoT system. Furthermore, we present, Random-Selection, an alternate simple strategy for ensemble creation that randomly omits a set of sensors when training sub-models. Third, we study the advantages and limitations of SECOE through experimentation of our system. We use two distinct datasets for our experimental study. Our study shows that SECOE is highly effective in overcoming sensor failure-induced data incompleteness.

The rest of the paper is organized as follows. The implementation details of SECOE are described in Section II, followed by Section III, in which we demonstrate the conducted experiments and results. Finally, we conclude in Section IV.
II. SENSOR CORRELATION-BASED ENSEMBLE (SECOE)

A. Overview

The main objective of the presented approach in this paper is to reduce the impact of sensor failures on the performance of IoT ML-based applications during inference time. Our method leverages an ensemble of a few sub-models each trained on a distinct subset of sensors formed based on the correlation of sensors in the IoT system. We use the correlation between sensors to minimize the number of models and build sub-models that perform similarly to the base model, which is a model trained using the entire IoT sensors data of the system.

Fig. 1 shows an overview of the training architecture of our method. Initially, our method groups the correlated sensors together, producing different groups of correlated sensors. Then, after computing the minimum number of sub-models (MinM), for instance, four sub-models, our method forms the features for each sub-model by selecting a distinct ~50% of sensors from each correlated group. The sub-model features created will then be used to build the sub-models along with the base model. The sensors included in each model’s features correspond to ~50% of the total sensors of the IoT system. Our method guarantees that for each sensor x in the IoT system, there exists at least one suitable sub-model trained using a subset of sensors that includes some correlated sensors to sensor x, excluding sensor x. Thus, during inference-time, if a sensor x fails, then the prediction accuracy of the IoT application is maintained via an ensemble of the suitable sub-models or at least one of the sub-models.

The following subsections describe the architecture of our approach in more detail.

B. Forming Correlated Groups

The correlation between sensors is the key factor of our approach. We have only considered the positive Pearson correlation measurement from 0 to 1 scale when forming the groups of correlated sensors. We have categorized the coefficient interpretations into four categories, which are strong, moderate, weak, and very weak correlation. The coefficient correlation value ranges are 0.76-1.0, 0.50-0.75, 0.25-0.50, 0.10-0.25 for the above-mentioned categories, respectively. Each sensor is grouped with its highly correlated sensor. Note that a correlated group Gi could contain correlated sensors with different coefficient interpretations.

C. Selecting Sensors and Building the Sub-models

We implemented an algorithm called Sensors Selection for Sub-Models (SSMs). Selecting sensors for each sub-model is based on SSMs. Each sub-model’s features can have a different 50% of the sensors from each correlated group Gi. If the size of a Gi is an odd number, then we select sensors for each sub-model equal to the nearest integer to 50% of the total sensors in such a Gi (i.e., No. of sensors in Gi is 7, then sensors that would be selected for each sub-model from such a Gi is 4). For reliability, a sensor must be included in at least one sub-model and excluded from another sub-model.

As a result, when up to almost 50% of sensors from each correlated group Gi fail, our proactive solution still provides a sub-model, trained on the remaining 50% of sensors (free-of-failure), that could make a real-time prediction with high accuracy using the correlated sensors to the ones that have failed. Furthermore, in case of the failure of many sensors concurrently, leading to the necessity of imputing the missing values of such sensors, the number of imputations is minimized by at least ~50% less than the base model.

SSMs algorithm requires two arguments: lists of the correlated groups CG and the size of the largest correlated group L. First, SSMs, using (1), finds the minimum number of the sub-models (MinM). Equation (1) finds MinM in the case of the percentage of sensors that would be selected from each Gi is set to 50%. MinM satisfies these two conditions: (a) Each sensor from each Gi is at least included in one sub-model, and (b) Each sensor is at least excluded from one sub-model.

\[
\text{MinM} = \begin{cases} \text{round}(50\%*L)+1, & \text{if } L \text{ is even.} \\ \text{round}(50\%*L), & \text{otherwise.} \end{cases}
\]  

(1)

The above two conditions guarantee that if any sensor x from the IoT system fails, there is a sub-model, trained without sensor x, to make a real-time prediction.

Second, SSMs, for each sub-model, selects the leftmost sensors from each Gi, equals to ~50% of sensors in Gi, then it shifts each Gi to the left by 1. This step ensures that sensors that will be picked for the next sub-model differ from those selected for the previous sub-model. As illustrated in Fig. 2, if we have a correlated group Gi as [H, B, D, A, G], SSMs will select the leftmost three sensors ([H, B, D]) from Gi for the first sub-model, then shift Gi to the left by 1. Next, for the second sub-model, SSMs will choose the leftmost three sensors ([B, D, A])

![Fig. 1 Architecture of the training phase.](image)

![Fig. 2 Example of selection of sensors from two correlated groups.](image)
from $G_t$ after being shifted. This procedure is repeated for the following sub-models. Finally, after iterating over every $G_t$, SSMs outputs the model features $MF_j$ for each sub-model. The resulted features for each sub-model is approximately 50% of the entire IoT sensors.

**D. Accuracy Optimization**

To optimize the performance of the IoT ML-based system during the presence of sensor failures, whenever records are received from sensors for prediction, they would be fed to the most suitable sub-model, the model that is trained on the subset of sensors that contains the least matched number of the failed sensors. The final prediction is their majority voting (ensemble) if there are several suitable sub-models. However, in some cases in which only two suitable models exist, the model with the highest training accuracy will be chosen for prediction. Note that the base model handles the prediction in the case of no failure of sensors. See Fig. 3.

**III. EVALUATION**

We evaluate our method on two datasets from the UCI repository [9]: (1) Dry-Beans and (2) Wall-Following Robot Navigation. Dry-Beans contains more than 13K samples forming 16 dry beans’ geometric features to classify the bean into one of 7 species. We slightly preprocessed the Dry-Beans dataset. More specifically, we rescaled the features of the training dataset by subtracting the mean and dividing all values by the standard deviation of the training samples. The second dataset, Wall-Following Robot Navigation, comes with different versions having 2, 4, and 24 attributes corresponding to numerical ultrasound sensors’ readings collected from sensors embedded on a SCITOS G5 robot navigating in a room following the wall in a clockwise direction. We used the version that contains 24 attributes. This version has more than 5K instances, through which the movement of the robot is predicted into one of 4 classes: move-forward, slight-right-turn, sharp-right turn, or slight-left-turn. In our experiments, we examine SECOE when all models are Multi-Layer Perceptron (MLPs) [8] trained on 85% of the dataset size and tested on the remaining 15%. To show the effectiveness of our SECOE, we compare SECOE with two other approaches. The first one is the base model. The second approach is Random-Selection, an approach we have created similar to ours. Instead of using correlation of sensors, Random-Selection randomly selects 50% of sensors from the entire sensors of the IoT system for each sub-model. We evaluate each approach based on its classification accuracy of the testing set. For simplicity, in all experiments, we utilize the Mean Imputation technique to replace the missing sensors’ values with the Mean values of their records across the training set. All models are built using the default parameters defined by the “Scikit-Learn” python library [10]. A detailed experimental study with additional ML algorithms can be found in [11].

**A. Performance of Models**

In this subsection, we illustrate the performance of all models when there are no sensor failures. The $MinM$ that SECOE produced is four sub-models for both datasets. Fig. 4 demonstrates the classification accuracy of the sub-models of both SECOE and Random-Selection and the base model on the two datasets: Dry-Beans and Wall-Following Robot Navigation. Although they are built using ~50% of the sensors from the dataset, most of the sub-models by SECOE are comparable to the base model in terms of test accuracy. In contrast, some Random-Selection sub-models perform poorly; more specifically, sub-model-1 and sub-model-3 on Wall-Following Robot Navigation dataset.

Choosing from the dataset a subset of sensors that contains the important sensors, which are the sensors that have a high impact on the model classification accuracy, is essential to make
the ML model performs well. Since Random-Selection randomly selects sensors for each sub-model, some important sensors or their correlated ones were ignored (not included in any sub-model), producing weak-performed models. SECOE offers well-performed sub-models built using the important sensors or ~50% of its correlated ones.

B. Static Sensor Failure Evaluation

To show the importance of SECOE, we created several test cases in which specific sensors were statically chosen as the simulated failed sensors. More specifically, a different number of sensors were chosen from each correlated group, equating to a maximum of 50% of the sensors in each group. Each test case corresponds to a different percentage of simultaneous sensor failures, from 5% up to 50%. In this experiment, for each test case, there are suitable sub-models or at least one suitable sub-model, free of sensors’ failure, trained on the correlated sensors to the failed sensors. Fig. 5 demonstrates the results of the test case scenarios for each dataset. As can be seen in Fig. 5, the test accuracy of the base model drops sharply when the percentage of failed sensors is 30% or more. At the highest percentage of sensor failures, SECOE maintains the accuracy at 90.84% and 84.98%, which are 77.49% and 94.42% more than what has been achieved by the base model on the Dry-Beans and Wall-Following Robot Navigation datasets, respectively. Such a dramatic drop in the base model’s accuracy is due to the OOD problem caused by imputing many missing values of the failed sensors. Substantially, SECOE minimizes the percentage of imputation to 0% compared to the base model. These results confirm the intuition behind SECOE: by using a few sub-models built carefully using correlation of sensors, we can achieve high classification accuracy even though up to 50% of sensors fail without the necessity of imputing the missing values of these sensors.

C. Random Sensor Failure Evaluation

To precisely examine our approach, we have mimicked real-world scenarios by simulating several test cases where sensors failed concurrently at random. We compare our SECOE with the Random-Selection and base model approaches under seven different fractions of failed sensors, ranging from 5% to 60%. Fig. 6 shows, for all datasets, the average test accuracy of 10 iterations per test case. The results indicate that SECOE outperforms the base model and Random-Selection. When less than 30% of sensors fail on both datasets, the base model exhibits performance close to our SECOE. However, its performance decreases severely when the failure rate of sensors is above 30% on both datasets. More specifically, when 60% of sensors fail, the accuracy of the base model reaches 56.83%, and 51.79% on the Dry-Beans and Wall-Following Robot Navigation datasets, respectively. These results are 13.10% and 9.57% less than what our SECOE has obtained on the two datasets, respectively. In SECOE, since each sub-model is trained on a distinct ~50% of the entire IoT sensors from the dataset, it minimizes the number of imputations of the missing sensors’ values by at least ~50% less than the base model.

Fig. 5 Test accuracy with a 95% confidence interval of SECOE versus the base model during the failure of sensors on each dataset.

Fig. 6 Mean test Accuracy of SECOE, Random-Selection, and Base model during random failure of sensors on each dataset.
Therefore, when more than 30% of the sensors in the system fail, the ensemble of sub-models by SECOE obtains better classification accuracy than the base model. On the other hand, due to its random selection of sensors, Random-Selection gets the lowest average accuracy in every simulated test case on the Dry-Beans dataset. However, on the Wall-Following Robot Navigation dataset, Random-Selection has similar performance to the base model when the percentage of failed sensors is 30% or above.

From analyzing these results, we can draw the conclusion that using the correlation of sensors to build an ensemble of sub-models alleviates the impact of concurrent sensor failures. To further enhance the performance of SECOE, we suggest that future research should investigate the performance of SECOE utilizing alternative imputation algorithms that take sensor “features” correlation into account when calculating the missing value of a sensor.

IV. CONCLUSION

In this paper, to make IoT-coupled ML systems robust against simultaneous sensor failures, we introduced SECOE, an ensemble of sub-models, each constructed utilizing distinct subsets of sensors obtained through a method that employs sensor correlation. In addition, we created a Random-Selection technique for comparison purposes. A series of empirical studies carried out have proven the intuition behind SECOE. At various percentages of simulated concurrent missing sensor data streams, SECOE enabled the IoT ML-based system to produce accurate real-time predictions without having to fill in the missing sensor readings. When 40-50% of sensors simultaneously failed, SECOE produced classification accuracies that were substantially better than the base model by 77.49% and 94.42% on the two datasets, respectively. In other scenarios in which SECOE was examined when sensors were failing randomly and imputation of their values was required, SECOE achieved a noticeably higher classification accuracy than the base model and Random-Selection. Furthermore, SECOE reduced the imputation of missing sensor data streams by at least 50% less than the base model.

ACKNOWLEDGMENT

This research has been partially funded by grants from the National Science Foundation (NSF) and the US Department of Agriculture's National Institute of Food and Agriculture (USDA NIFA), and gifts from Accenture Research Labs. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding agencies or the companies mentioned above.

REFERENCES

[1] L. Atzori, A. Iera, and G. Morabito, “The internet of things: a survey,” Computer Networks: The Int. J. of Comput. and Telecommun. Netw., vol. 54, no. 15, pp. 2787–2805, Oct. 2010.
[2] B. Fekade, T. Maksymyuk, M. Kyrk and M. Jo, "Probabilistic Recovery of Incomplete Sensor Data in IoT," in IEEE Internet of Things J., vol. 5, no. 4, pp. 2282-2292, Aug. 2018, doi: 10.1109/JIOT.2017.2730360.
[3] T. Wang, H. Ke, A. Jolfaei, S. Wen, M. S. Haghighi and S. Huang, "Missing Value Filling Based on the Collaboration of Cloud and Edge in Artificial Intelligence of Things," in IEEE Transactions on Industrial Informatics, vol. 18, no. 8, pp. 5394-5402, Aug. 2022, doi: 10.1109/TII.2021.3126110.
[4] E. B. Ilyas, M. Fischer, T. Iggena and R. Tönjes, "Virtual sensor creation to replace faulty sensors using automated machine learning techniques," 2020 Global Internet of Things Summit (GloTS), 2020, pp. 1-6, doi: 10.1109/GIOTS49054.2020.9119681.
[5] F. -K. Tsai, C. -C. Chen, T. -F. Chen and T. -J. Lin, “Sensor abnormal detection and recovery using machine learning for IoT sensing systems,” IEEE 6th Int. Conf. on Ind. Eng. and Appl. (ICIEIA), 2019, pp. 501-505, doi: 10.1109/IEA.2019.8715215.
[6] J. Fonollosa, A. Vergara and R. Huerta, “Sensor failure mitigation based on multiple kernels,” SENSORS, 2012 IEEE, 2012, pp. 1-4, doi: 10.1109/ICSENS.2012.6411124.
[7] O. Sagi and L. Rokach, Ensemble learning: A survey, Data Mining and Knowledge Discovery, vol. 8, no. 4, pp. 1–18, 2018.
[8] Aurélien Géron, Hands-on Machine Learning with Scikit-Learn and TensorFlow, 2nd ed. Sebastopol, CA: O’Reilly Media, Inc., 2019, p. 286.
[9] D. Due and C. Graff, UCI Machine Learning Repository, Irvine, CA: University of California, School of Information and Computer Science, 2017. Accessed on: Oct 15, 2021. [Online]. Available: http://archive.ics.uci.edu/ml.
[10] F. Pedregosa et al., “Scikit-learn: machine learning in python,” The J. of Machine Learning Research, vol 12, pp. 2825-2830, Jan. 2011.
[11] Y. AlShehri and L. Ramaswamy, “SECOE: Alleviating sensors failure in machine learning-coupled IoT systems,” Oct. 2022, arXiv:2210.02144. [Online]. Available: https://arxiv.org/abs/2210.02144.