Mitigating Data Integrity Attacks in Building Automation Systems using Denoising Autoencoders

Caezarina Marie Calimbahin
Department of Computer Science
University of the Philippines
Quezon City, Philippines
cfcalimbahin@up.edu.ph

Susan Pancho-Festin
Department of Computer Science
University of the Philippines
Quezon City, Philippines
susan.pancho@up.edu.ph

Jhoanna Rhodette Pedrasa
Electrical and Electronics Engineering Institute
University of the Philippines
Quezon City, Philippines
jipedrasa@up.edu.ph

***Abstract*—Building automation systems (BAS) are a class of cyber-physical systems that aim to improve the efficiency of buildings through intelligent control. Typically, sensor measurements are used to compute for the optimal control action for equipment that minimizes the energy consumption while maintaining the comfort of occupants. Given the heavy reliance on sensor data, ensuring their validity is an essential concern in BAS. Several work have explored sensor validation, mostly for the purpose of automatic fault detection and diagnostics (AFDD). However, recent studies show BAS vulnerable to adversarial attacks - an area that has lacked consideration in the early design of BAS. In this work, we propose a sensor correction model, based on deep learning framework, for mitigating against data integrity attacks in BAS. We test our approach on real-world data obtained from a retrofitted air conditioning (AC) control testbed, with injected data simulating different attacker types and number of attacked sensors. We show that the model is capable of mitigating attacks by comparing with a baseline where no model is employed.**

***Index Terms*—sensors, data integrity attack, building automation system, cyber-physical system**

I. Introduction

Heating, ventilation, and air conditioning (HVAC) units account for almost half of the energy used in buildings [1], and may even exceed for buildings in tropical climate areas [2]. This level of consumption makes it a primary concern for efficiency improvements.

Building automation systems (BAS) are a class of cyber-physical systems (CPS) that aim to improve the efficiency of HVAC and other equipment through intelligent control. The primary functions include sensing of environmental factors and optimizing control strategies, that would minimize the energy consumption while maintaining the comfort of occupants [3]. Given the heavy reliance on sensor data, ensuring their validity is an essential concern in BAS.

The problem of sensor validation has been extensively studied in BAS, mostly for the purpose of automatic fault detection and diagnostics (AFDD) [4] [5]. AFDD methods employ redundancy to estimate a target sensor's value. Specifically, hardware redundancy refers to the use of additional components such as extra sensors performing the same measurement. A final read-out is obtained through a voting mechanism. Analytical redundancy, on the other hand, uses mathematical models based on physics-based process description or data-driven methods; The latter gaining attention due to the massive amount of data collected in modern buildings [6].

Recently, attacks in CPS [7] [8] have demonstrated malicious entities capable of injecting modified sensor or actuator signals, in order to manipulate an underlying physical process. In [9], vulnerability analysis of BAS show integrity attacks have adverse financial impact on the energy consumption of buildings and safety risk due to damaging of circuits. In [10], the capability of stealthy attacks to drive the operation of BAS to undesired levels is evaluated. Attacks targeting specific protocols, as well as solutions are also explored in [11] [12] [13] [14]. However, as a result, security measures vary from one system architecture to another.

Redundancy has also been considered against integrity attacks in BAS. The work in [15] uses existing hardware but with additional data transmission paths. Inconsistent received values are used for detection and voting mechanism carried out to mask over incorrect values. In [9], a two-step process consisted of a detection phase and an analytical model (called Virtual Sensor) that estimates corrected sensor data using physical system dynamics.

In this study, we propose analytical redundancy based on deep learning framework for sensor data correction. Specifically, we develop a denoising autoencoder (DAE) model leveraging on spatial correlation among sensors. DAE models have shown success in AFDD on various fields, including telemetry data [16], mechanical systems [17] [18] [19], among others. To our best knowledge, no other research has still been done with regards to the use of DAE for mitigating integrity attacks in BAS.

The contributions of this work are the following:

- We employ DAE for sensor data correction. Furthermore, we propose a novel approach incorporating domain knowledge as corruption process during DAE training.
- Data correction method is run on the server and has the advantage of not relying on different architectures and protocols.
- We test our approach on real-world data obtained from a retrofitted testbed, with injected data simulating different attacker types and varying number of attacked sensors.

This work is supported by a Commission on Higher Education (CHED) Philippine California Advanced Research Institues (PCARI) grant under Project IIID54 - Resilient Cyber Physical Societal Scale Systems.

978-1-7281-1340-1/19/$31.00 ©2019 IEEE

- We show that the model is able to mitigate attacks by comparing with a baseline where no correction is employed.

The rest of the paper is organized as follows: Section II presents the background on autoencoders and implementation for sensor data correction. Section III describes the experiment set-up, including the system under consideration, attack model, and performance metrics. Section IV provides discussion of results. Sections V and VI present conclusions and future work.

II. Proposed Method

A. Denoising Autoencoder (DAE)

The traditional autoencoder (AE) [20] is a three-layer feed-forward neural network that attempts to copy its input to its output. Typically, it is composed of visible input and output layers with the same number of dimensions, and a hidden layer (also called bottleneck layer) with smaller number dimensions. The main concept behind AE is that by encoding data at (and decoding from) a lower dimension, the network is forced to learn the most salient features that would allow minimum reconstruction error at the output. Recently, the denoising autoencoder (DAE) [21] has been developed as a variant of the traditional autoencoder. The objective of DAE is to make the learned representations robust to partial corruption of the input, hence the term denoising.

B. DAE for Sensor Data Correction

Sensor attack mitigation can be modeled as a data reconstruction problem. For this, the DAE provides a natural framework. In this study, we consider spatial correlation among sensors, as follows: Let $\mathbf{s} = \{s_1, .., s_p\}$ be a measurement vector where p is the number of sensors. Further let $\tilde{\mathbf{s}}$ be a partially corrupted version of $\mathbf{s}$, mapped to a hidden representation $\mathbf{h}$ through a deterministic mapping as follows:

$$\mathbf{h} = f_\theta(\tilde{\mathbf{s}}) = \phi(\mathbf{W}\tilde{\mathbf{s}} + \mathbf{b}) \tag{1}$$

where $\theta = [\mathbf{W}, \mathbf{b}]$ are model parameters, and ϕ is a nonlinear activation function. The transformation can be regarded as an encoding step. After which, the hidden representation $\mathbf{h}$ is mapped back to a visible output layer $\hat{\mathbf{s}}$ through a similar transformation:

$$\hat{\mathbf{s}} = g_{\theta'}(\mathbf{h}) = \phi(\mathbf{W}'\mathbf{h} + \mathbf{b}') \tag{2}$$

This reverse transformation can be regarded as a decoding step. The parameters θ, $\theta^{'}$ of the model are tuned during training by minimizing a loss function, $L(\mathbf{s}, \hat{\mathbf{s}})$ representing the reconstruction error between the ground truth vector and the reconstructed output, such as the squared error, $\|\mathbf{s} - \hat{\mathbf{s}}\|^2$.

The common choice of corruption process are additive Gaussian noise, masking noise (forcing a fraction to 0), and salt-and-pepper noise (forcing to minimum and maximum possible values) [22]. In this work, we incorporate domain-based corruption process based on the known vulnerabilities. Specifically, we compare the performance of the model when corrupted with different attacker types [10].

III. Experiment Set-up

A. Dataset

The dataset used in the study was obtained from normal operation of a retrofitted BAS testbed for AC control. The system controls the indoor air temperature of the Computer Security Group (CSG) laboratory at the Department of Computer Science (DCS) building at the University of the Philippines. Figures 1 and 2 show the overview of the hardware set-up (components and interface) and the room placement layout. In real-time, a temperature sensor network (composed of S_1-S_4) transmits measurements to a server PC (not shown in the room layout). The server computes for the optimal control action and sends it to the actuator (A_1). Finally, the actuator executes the command by transmitting infrared codes to the AC unit. The server runs the learning-based model predictive control (LBMPC) algorithm from [23] which aims to keep the indoor air temperature within the range 20-24°C, in compliance with the Occupational Safety and Health Administration (OSHA) guidelines of thermal comfort.

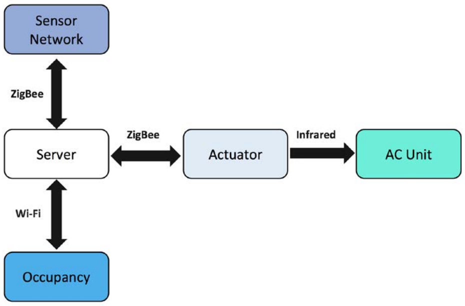

Fig. 1. Hardware components and interface

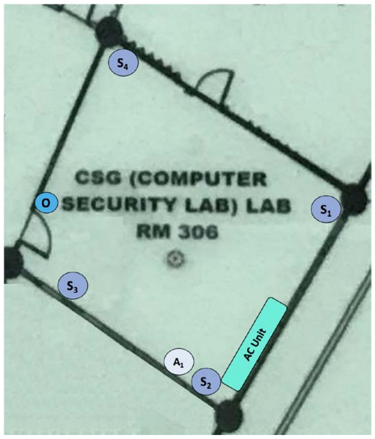

Fig. 2. Room Layout

Collected data is composed of 35 days, from October 11 - November 20, 2017 (excluding Sundays), from 4 temperature sensors with 1-minute sampling time. We denote this as ground truth for which we compare the reconstruction results of the denoising autoencoder (DAE) model against data integrity attacks.

B. Attack Model

To test our approach, we use the model of targeted attacks from [10]. These are the ones where adversaries tailor their strategy with the aim of disrupting the physical system under control. Two types of attackers were also defined in [10], namely a powerful attacker who tends to be stealthy by injecting the minimum values needed for a successful attack, and a naive attacker who injects random values.

TABLE I
ATTACK SCENARIOS

End-goal	Type	Number of attacked sensors
Under-cool	Powerful	0
		1
		2
		3
	Naive	0
		1
		2
		3

C. Performance Metrics

We measure the performance of the model based on accuracy, and sensitivity metrics [24].

1) Accuracy: Measured based on the root mean squared error (RMSE) between the model predictions, $\hat{\mathbf{s}}$, and the ground truth sensor values, $\mathbf{s}$. The RMSE for sensor i is given by:

$$S^{i}_{RMSE} = \sqrt{\frac{1}{N}\sum_{k=1}^{N}\left(\hat{s}_{ki} - s_{ki}\right)^{2}} \tag{3}$$

where N is the total number of test samples, s_{ki} is the ground truth measurement of sensor i at time k, and $\hat{s}_{ki}$ is the model estimate of sensor i at time k. The average RMSE for all sensors is:

$$S_{RMSE} = \frac{1}{p}\sum_{i=1}^{p} S^{i}_{RMSE} \tag{4}$$

where p is the total number of sensors. Since accuracy is a measure of error, a low value is desired.

2) Auto sensitivity (Robustness): Measure of the model's ability to make correct sensor predictions when the respective sensor value is incorrect due to some corruption. As such, this metric is computed for sensors i that are under attack, using the following equation:

$$S^{i}_{Auto} = \frac{1}{N}\sum_{k=1}^{N}\left|\frac{\hat{s}^{attack}_{ki} - \hat{s}_{ki}}{\tilde{s}^{attack}_{ki} - \tilde{s}_{ki}}\right| \tag{5}$$

where i and k are the sensor and sample index, respectively. $\tilde{s}_{ki}$ is the original sensor value without attack, $\hat{s}_{ki}$ is the model estimate against $\tilde{s}_{ki}$, $\tilde{s}^{attack}_{ki}$ is the attacked sensor value, and $\hat{s}^{attack}_{ki}$ is the model estimate against $\tilde{s}^{attack}_{ki}$.

3) Cross sensitivity (Spillover): Measures the influence of a corrupted sensor input on the prediction of other sensors. The cross sensitivity of sensor j with respect to an attacked sensor i is computed as follows:

$$S^{ji}_{Cross} = \frac{1}{N}\sum_{k=1}^{N}\left|\frac{\hat{s}^{attack}_{kj} - \hat{s}_{kj}}{\tilde{s}^{attack}_{ki} - \tilde{s}_{ki}}\right| \tag{6}$$

where k is the sample index, $\tilde{s}_{ki}$ is the original value of sensor i without any attack, $\hat{s}_{kj}$ is the model estimate of sensor j against $\tilde{s}_{ki}$, $\tilde{s}^{attack}_{ki}$ is the value of attacked sensor i, and $\hat{s}^{attack}_{kj}$ is the model estimate of sensor j against $\tilde{s}^{attack}_{ki}$. This metric is computed for all sensors that are not attacked. The average cross sensitivity of an unattacked sensor j over all attacked sensors i is given by:

$$S^{j}_{Cross} = \frac{1}{|M|}\sum_{i\in M} S^{ji}_{Cross} \tag{7}$$

where M represents the set of attacked sensor indices. In both sensitivity metrics, smaller values (minimum of 0) are preferred, which denotes that the model outputs are not significantly affected by the corrupted inputs.

IV. RESULTS

We employed the K-fold cross validation method for training and testing. In particular, data was divided into 7 groups with 5 random days each. We initially held-out Group 1 as test data, and the remaining groups for training and validation. From the said held-out test data, we simulated the different attack scenarios from Table I. The process is repeated until each group has been used as test data. The overall model performance is computed by averaging the evaluation scores among all groups. The choice of K = 7 for this study allows for equal partition among the data such that at each fold, approximately 14% (5 days at held-out group) are used for testing, and the remaining groups for training and validation with 69% and 17%, respectively.

A. Training

1) Experiment on corruption process: Table II shows the results in terms of average root mean squared error (S_{RMSE}). Each column represents a test scenario (attacker type - number of attacked sensors) when trained with different corruption types. Highlighted in bold are the best performance on each test.

In the test case where the number of attacked sensors = 0, training with powerful corruption yields better performance. On all the remaining tests, specifically where there are attacked sensors, training with naive corruption resulted in better performance. The random injections of a naive attacker on more training samples allows the model to be more robust to attacks. Further on in this study, we select the naive attack as corruption process during training. We also show that we can still improve its performance when the number of attacked sensors = 0 by optimizing the hyperparameters of the model.

TABLE II

S_{RMSE} COMPARISON WHEN TRAINED ON DIFFERENT CORRUPTION PROCESS

	Powerful				Naive			
	0	1	2	3	0	1	2	3
Corruption process (train)								
Powerful	**0.2135**	0.9073	0.8075	0.8320	**0.2135**	1.3202	2.0128	2.5475
Naive	0.3736	**0.5030**	**0.5501**	**0.5525**	0.3736	**0.6852**	**1.1572**	**1.7562**

2) Hyperparameter optimization: Grid search method was conducted to determine the set of hyperparameters of the model. Specifically, we used:

- Architecture: 4-3-4 (nodes at each layer; 4 nodes at input and output represent the number of sensors)
- Activation: Rectified Linear Unit
- Corruption rate $v = 0.1$ (probability of corrupted inputs during training)

B. Test

We used the held-out unseen test set at each fold to simulate different attack scenarios. Attack simulation is performed as follows: From the five days data of the test set, m sensors are randomly selected and corrupted per day, where m is the number of attacked sensors, and the corruption is based on the model for powerful or naive attacker. The overall performance at each test scenario is computed by averaging the scores from all folds. In order to determine the advantages and disadvantages of the reconstruction model, we consider in this study a baseline where no model is employed i.e., if sensor data are sent directly to the controller.

1) Powerful Attacks: Figure 3 shows RMSE comparison between DAE model and baseline. When the number of attacked sensors = 0, a perfect RMSE of 0.0000 is observed for the baseline. This is because there is no corruption injected on all sensors. This presents one limitation of dimensionality reduction models, such as autoencoders, which is the inherent error at the reconstructed output due to the compression of input data. Moreover, the DAE model was able to reconstruct clean data with an average RMSE of 0.2622°C. In all the remaining tests, with number of attacked sensors equal to 1, 2, and 3, the DAE performed better against the baseline as shown in lower computed RMSE among all sensors.

Tables III and IV shows DAE model auto sensitivity and cross sensitivity, respectively. Auto sensitivity quantifies the effect of a corrupt sensor on its reconstruction output, while cross sensitivity quantifies the effect of a corrupt sensor on the reconstructed output of other sensors.

TABLE III

AUTO SENSITIVITY FOR POWERFUL ATTACKS

	Number of attacked sensors			
	0	1	2	3
s_1	-	0.1087	0.1921	0.4377
s_2	-	0.1179	0.2954	0.4731
s_3	-	0.2208	0.3400	0.6997
s_4	-	0.0342	0.4078	0.7572
Average	-	0.1184	0.3139	0.5919

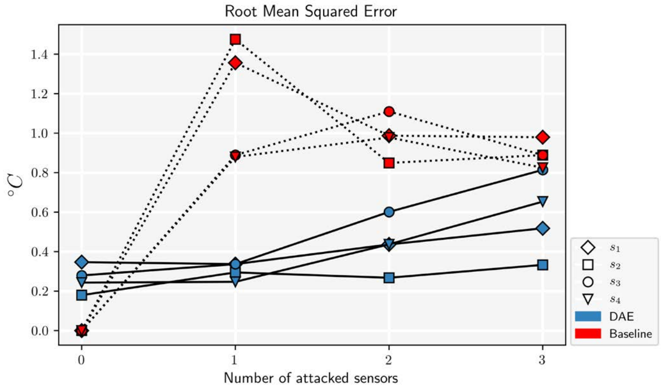

Fig. 3. Root Mean Square Error (RMSE) comparison of baseline and DAE model against powerful attacks with varying number of attacked sensors

TABLE IV

CROSS SENSITIVITY FOR POWERFUL ATTACKS

	Number of attacked sensors			
	0	1	2	3
s_1	-	0.0680	0.2374	0.4487
s_2	-	0.0716	0.1230	0.2399
s_3	-	0.0911	0.2460	0.5005
s_4	-	0.1102	0.1990	0.3656
Average	-	0.0853	0.2014	0.3909

The maximum observed sensitivity of 0.7572 shows that the model was still able to mitigate (with sensitivity < 1), even on the case when the number of attacked sensors = 3. We note that a sensitivity measure of 1 means that the model prediction follows the corrupted signal with zero residuals. The degradation of performance as the number of attacked sensors increases is expected, and is mainly due to the model's reliance on the clean inputs to reconstruct data at the the output. Moreover, in all cases, the model was able to mitigate attacks in all tests.

To illustrate the resulting signal reconstruction of the model, Figures 4 and 5 show simulation for powerful attack when the number of attacked sensors = 1 and 3, respectively. The black line represents ground truth. The red dotted line represents the attack signal. The blue line represents DAE model-estimate. In the case where the number of attacked sensors is 1, we can visually observe that the model-estimate closely follows the ground truth signals, which is desired. In contrast, when the number of attacked sensors is 3, with now more than half of the sensors corrupted, we can visually observe that the model-estimate closely follows the attack signal.

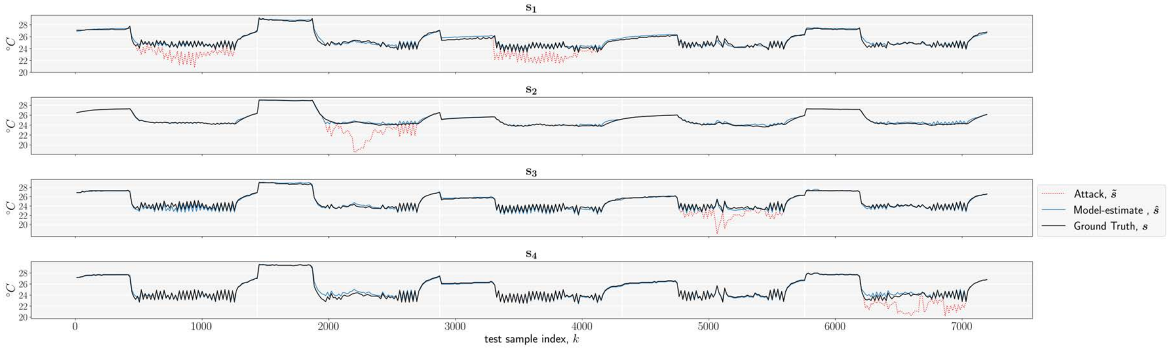

Fig. 4. Sensors 1-4 (top to bottom) where the number of attacked sensors = 1

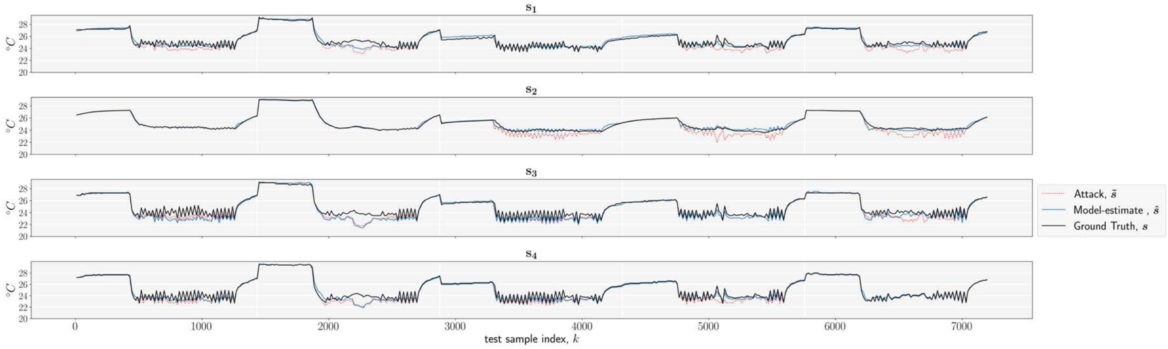

Fig. 5. Sensors 1-4 (top to bottom) where the number of attacked sensors = 3

2) Naive Attacks: As compared to a powerful attacker, a naive one randomly selects values to be injected at each time step. The higher injected attack values (compared to powerful attacks) are shown in baseline RMSE reaching up to more than 3°C on all sensors when the number of attacked sensors = 3. Furthermore, results show that DAE model was able to perform better than the baseline at all tests except when the number of attacked sensors = 0, with baseline having perfect RMSE of 0.0000. Figure 6 show corresponding RMSE performance.

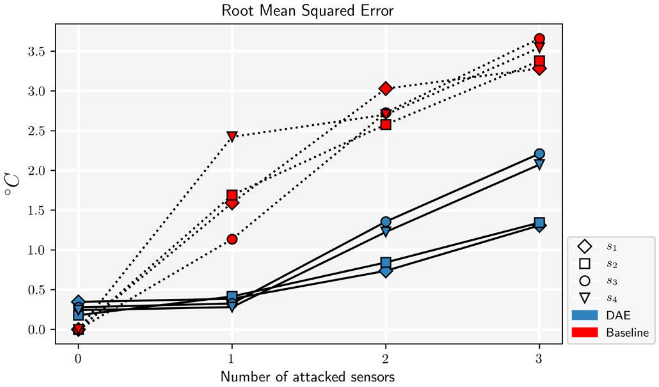

Fig. 6. Root Mean Square Error (RMSE) comparison of baseline and DAE model against naive attacks with varying number of attacked sensors

The sensitivity results are presented in Tables V and VI. Up to the point where the number of attacked sensors = 2, the model was able to mitigate attacks with sensitivity less than 1. However, at the worst case, when the number of attacked sensors = 3, sensitivity reaches more than 1, which means reconstructed output is amplified more than the injected signal. Moreover, results for sensitivity follow the trend for powerful attacks, with degradation in performance as the number of attacked sensors increases.

TABLE V
AUTO SENSITIVITY FOR NAIVE ATTACKS

	Number of attacked sensors			
	0	1	2	3
s_1	-	0.0543	0.3184	1.1158
s_2	-	0.2556	0.5194	0.7735
s_3	-	0.0819	0.5803	1.4990
s_4	-	0.0597	0.6675	0.8303
Average	-	0.1094	0.5252	1.0546

TABLE VI
CROSS SENSITIVITY FOR NAIVE ATTACKS

	Number of attacked sensors			
	0	1	2	3
s_1	-	0.0548	0.3456	1.0880
s_2	-	0.0344	0.1823	0.2992
s_3	-	0.0668	0.4187	1.1697
s_4	-	0.0579	0.2262	0.5010
Average	-	0.0535	0.2907	0.7652

V. Conclusions

This work presents the application of denoising autoencoders (DAE) for sensor data correction against data integrity attacks in Building Automation Systems (BAS). Ground truth dataset was obtained from real-world operation of a retrofitted air conditioning (AC) control testbed.

Two experiments for optimization of the model were carried out, specifically (1) to determine the corruption process during training and (2) for hyperparameter selection. From the experiments, the best performing model was selected.

Accuracy of model predictions was measured in terms of root mean squared error (RMSE), and compared with a baseline where no model is employed. DAE-specific metrics, such as auto sensitivity (robustness) and cross sensitivity (spillover), were also computed to quantify the effect of corrupt inputs on the model outputs.

Results show that the model was able to mitigate both types of attacks, as computed with lower RMSE compared with the baseline. However, a tradeoff of minimal reconstruction error when the number of attacked sensors is 0 is observed. This is an inherent limitation of dimensionality reduction models, such as autoencoders, due to compression of data. Results also show performance degradation in all metrics as the number of attacked sensors increases. This is expected and can be attributed to the model's reliance on the clean inputs to reconstruct data at the output.

VI. Future Work

The study evaluated denoising autoencoder (DAE) models relying on spatial correlation among sensors. This was implemented through a fixed number of neural computing units at the input and output, representing measurement vector at each time step. In our future work, we explore other model variations, such as including temporal correlation, and other dimensionality reduction techniques such as principal component analysis (PCA).

We tested our model on single time-step attacks, simulating attacked sensors at each sample independently. This is a limitation of a fixed ground truth dataset of a closed loop system, in which an attack at a certain time step changes the ground truth at the succeeding time step, wherein the said change is unavailable in the existing data. Testing against consecutive attacks can be done through real-time deployment.

References

[1] L. Pérez-Lombard, J. Ortiz, and C. Pout, "A review on buildings energy consumption information," *Energy and buildings*, vol. 40, no. 3, pp. 394–398, 2008.

[2] K. Chua, S. Chou, W. Yang, and J. Yan, "Achieving better energy-efficient air conditioning–a review of technologies and strategies," *Applied Energy*, vol. 104, pp. 87–104, 2013.

[3] W. Kastner, G. Neugschwandtner, S. Soucek, and H. M. Newman, "Communication systems for building automation and control," *Proceedings of the IEEE*, vol. 93, no. 6, pp. 1178–1203, 2005.

[4] W. Kim and S. Katipamula, "A review of fault detection and diagnostics methods for building systems," *Science and Technology for the Built Environment*, vol. 24, no. 1, pp. 3–21, 2018.

[5] S. Katipamula and M. R. Brambley, "Methods for fault detection, diagnostics, and prognostics for building systemsa review, part i," *Hvac&R Research*, vol. 11, no. 1, pp. 3–25, 2005.

[6] M. Manic, K. Amarasinghe, J. J. Rodriguez-Andina, and C. Rieger, "Intelligent buildings of the future: Cyberaware, deep learning powered, and human interacting," *IEEE Industrial Electronics Magazine*, vol. 10, no. 4, pp. 32–49, 2016.

[7] N. Falliere, L. O. Murchu, and E. Chien, "W32. stuxnet dossier," *White paper, Symantec Corp., Security Response*, vol. 5, no. 6, p. 29, 2011.

[8] J. Slay and M. Miller, "Lessons learned from the maroochy water breach," in *International Conference on Critical Infrastructure Protection*. Springer, 2007, pp. 73–82.

[9] K. Paridari, A. El-Din Mady, S. La Porta, R. Chabukswar, J. Blanco, A. Teixeira, H. Sandberg, and M. Boubekeur, "Cyber-physical-security framework for building energy management system," in *Proceedings of the 7th International Conference on Cyber-Physical Systems*. IEEE Press, 2016, p. 18.

[10] C. M. Calimbahin, S. Festin, and J. R. Pedrasa, "Sensor Data Correction in Building Automation Systems using Deep Learning," Master's thesis, University of the Philippines, Diliman, Quezon City, Philippines, 2019.

[11] P. Ciholas, A. Lennie, P. Sadigova, and J. Such, "The security of smart buildings: a systematic literature review," *arXiv preprint arXiv:1901.05837*, 2019.

[12] W. Granzer, W. Kastner, G. Neugschwandtner, and F. Praus, "Security in networked building automation systems," in *2006 IEEE International Workshop on Factory Communication Systems*. IEEE, 2006, pp. 283–292.

[13] N. Vidgren, K. Haataja, J. L. Patino-Andres, J. J. Ramirez-Sanchis, and P. Toivanen, "Security threats in zigbee-enabled systems: Vulnerability evaluation, practical experiments, countermeasures, and lessons learned," in *2013 46th Hawaii International Conference on System Sciences*. IEEE, 2013, pp. 5132–5138.

[14] I. I. Bezukladnikov and E. L. Kon, "Method to counter the threat of covert channels in lonworks-based industrial control systems," in *2015 9th International Conference on Application of Information and Communication Technologies (AICT)*. IEEE, 2015, pp. 173–177.

[15] Z. Zheng and A. Reddy, "Towards improving data validity of cyber-physical systems through path redundancy," in *Proceedings of the 3rd ACM Workshop on Cyber-Physical System Security*. ACM, 2017, pp. 91–102.

[16] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders with nonlinear dimensionality reduction," in *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*. ACM, 2014, p. 4.

[17] R. Thirukovalluru, S. Dixit, R. K. Sevakula, N. K. Verma, and A. Salour, "Generating feature sets for fault diagnosis using denoising stacked auto-encoder," in *2016 IEEE International Conference on Prognostics and Health Management (ICPHM)*. IEEE, 2016, pp. 1–7.

[18] W. Sun, S. Shao, R. Zhao, R. Yan, X. Zhang, and X. Chen, "A sparse auto-encoder-based deep neural network approach for induction motor faults classification," *Measurement*, vol. 89, pp. 171–178, 2016.

[19] G. Jiang, P. Xie, H. He, and J. Yan, "Wind turbine fault detection using a denoising autoencoder with temporal information," *IEEE/ASME Transactions on Mechatronics*, vol. 23, no. 1, pp. 89–100, 2018.

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

[21] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 1096–1103.

[22] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *Journal of machine learning research*, vol. 11, no. Dec, pp. 3371–3408, 2010.

[23] A. Aswani, N. Master, J. Taneja, D. Culler, and C. Tomlin, "Reducing transient and steady state electricity consumption in hvac using learning-based model-predictive control," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 240–253, 2012.

[24] J. W. Hines and D. R. Garvey, "Development and application of fault detectability performance metrics for instrument calibration verification and anomaly detection," *Journal of Pattern Recognition Research*, vol. 1, no. 1, pp. 2–15, 2006.