Deep Learning for Estimating Sleeping Sensor's Values in Sustainable IoT Applications

Djamel Djenour
CSRC, Dep. Computer Science
University of the West of England
Bristol, UK
djamel.djenouri@uwe.ac.uk

Roufaida Laidi
CERIST Research Center and
Ecole Supérieure d'Informatique
Algiers, Algeria
ar_laidi@esi.dz

Youcef Djenouri
Dep. of Mathematics and Cybernetics
SINTEF Digital
Oslo, Norwa
youcef.djenouri@sintef.no

***Abstract*—The aim of this work is to develop a deep learning model that uses spatial correlation to enable turning turn off a subset of sensors while predicting their readings. This considerably saves the energy that would be consumed by those sensors both for sensing and communications (reporting the reading to the central station), which prolongs sensors' lifetime and opens sky for a plethora of Internet of Things (IoT) applications. Subject of this research, event-based sensing is more challenging than periodic sensing and is uncovered in the literature. We explore advanced learning approaches including Graph Convolutional Network (GCN) and Generative Adversarial Networks (GANs) and comb them in a novel way to derive a solution that uses both spatial correlation and the readings of the active sensors to accurately generate the missing readings from inactive sensors. The proposed solution is holistic and does not rely on any duty-cycling scheduling policy. A generic random pattern is used in this paper in which every sensor is duty-cycled randomly. The structure of the network is plugged into the GCN through a graph derived using the sensing range, as well as the euclidean distance between the sensors that determines the wights on the edges. Moreover, the accuracy of the GCN is enhanced by optmizing the weights of its deep neural network with a GANs and a game theory based model, which adversarially trains the GCN's generator by estimating the generator's performance and calculating the Wasserstein distance between the real and the generated data. The proposed solution is evaluated in comparison with the most relevant state-of-the-art approaches in terms of accuracy, energy consumption. The results show that the proposed solution provides high performance and is clearly superior to all the compared solutions in terms of reducing energy consumption and improving accuracy.**

***Keywords*—IoT, wireless sensor network, Deep Neural networks, adversarial training, graph convolutional networks, sensor energy saving.**

I. Introduction

Preserving batteries of low-power devices, such as wireless sensors, is a key solution for achieving sustainable Internet of Things (IoT). This will not only enable an abundance of applications that will benefit our society and economy but will significantly contribute towards green computing by reducing battery disposal and preserving the environment. However, today's real-world applications face a big challenge in achieving long term deployment without battery replacement. Many research efforts have been devoted to the design of power-management policies and protocols, e.g., duty-cycling scheduling [1], medium access protocols [2], routing protocols [3], optimal relay node placement [4], etc. These solutions contribute in prolonging batteries life-time but remain insufficient and are reaching their limits. Other trends focused on the energy harvesting from the environmental resources such as electromagnetic waves (wireless charging), solar, wind, etc., and proposed solutions to design appropriate hardware [5], as well as adaptive models and protocols, e.g., [6], [7]. All this helps prolonging the lifetime but remains insufficient given the instability and variability of the ambient resources. This problem is addressed herein by taking advantage of spatial correlation between sensors and exploring advanced machine learning methods to accurately estimate sensorial data readings when sensors are turned off. This way, the sensors can be kept off as long as possible while using the model to generate the missing data.

We focus on event-based applications that were uncovered in the literature from this perspective. Although optimal sensing coverage is often required for various applications, continuous monitoring of the entire field is not always needed for detecting events that occur during short periods and can be detected by more than one sensor. As the position of a sensor impacts its reading, several sensors may detect the same event when nodes' sensing fields overlap. Therefore, sensors can mutually cover missing detections of one another if they are correlated to the event. For example, in a commercial building application motion sensors in certain areas reporting the arrival of employees are generally sequentially related. By creating a model capable of learning data correlation, the sensors' values will be predicted and the energy that those sensors would consume for sensing and communications (transmission of their reading to central stations) will be saved. Two approaches may be considered to reach this target, 1) turning "off" all the sensors simultaneously then exploiting temporal correlations to predict future values from their last readings, 2) turning part of the network "on" and using spatial correlations to deduce the values of the sleeping sensors. We already explored the first approach in [8] and used a sequence model based on LSTM

(Long Short-term Memory) that learns temporal correlations along with a reinforcement learning model that decides when to activate/deactivate the sensors. Despite the promising results, this approach requires significant data and training time. The second approach is explored in this paper. This approach has already been used in the literature and some machine learning based approaches have been proposed. However, most of these solutions reduce energy consumption only for a part of the network while leaving it high for some sensors. They may save energy of less-informative sensors while putting all the network's operation burden on the most indicative ones. For example, Silvestri et al. [9] update the subset of active nodes only when their data do not describe the global network. They neither consider the current state of the nodes' batteries, nor balance the load between the sensors. We target the entire network with a dynamic approach that balances the load. Our approach operates both on sensor and data collector levels.

The following part of the paper is organized as follows. Sec.II presents the related work. Assumptions and problem formulation are given in Sec.III. The proposed solution is described in details in Sec.IV, including the GCN and the GAN models. Sec.V reports the experimental study and results. Finally Sec.VI draws conclusions and perspectives.

II. Related Work

Duty cycling is a technique[1] that schedules the nodes or their radios to continuously alternate between active and sleep modes (or on/off modes) while reducing the active periods to preserve energy. This technique is essential for low-power IoT devices but introduces latency in communication and may affect real-time applications. Furthermore, it is reaching its performance limit for applications with stringent energy constraints, which yields the need for innovative approaches [10]. One promising solution is to substitute the usage of radio and sensing modules by predicting sensory data using advanced machine learning methods. Dias et al. [11] showed that forecasting sensorial data can reduce transmissions without impacting the quality of the data. There are two classes of prediction schemes [11]; single prediction vs. dual prediction. In the first category, a central device (a data collector) holds the prediction model and generates sensorial values. In the second category, the central node (a cluster-head or the gateway) and sensors collaborate to make the prediction. The main drawback of the second category is the limited capacity of sensor nodes that may fail to hold the computation/memory resources greedy prediction models. The energy consumed in calculation may also exceed that saved through the model. Moreover, dual-based prediction tends to neglect the sensing energy that may be more important than transmission energy in some application scenarios. This motivates for the single prediction approach that is considered in this paper.

Silvestri et al. [9] explored probabilistic models and proposed an approach to infer sensor measurements using a Gaussian distribution. The approach is a heuristic that selects a set of monitoring sensors to remain awake. The authors suggested to periodically detect changes in the environment and run a new training phase if required to ensure that the density function remains inline with the data. Such Gaussian distributions based models are suitable for continuous and periodic monitoring. However, event occurrence does not always follow statistical probability distributions, which calls for novel solutions based on advanced knowledge of the domain [11]. Emekci et al. [12] proposed a linear transformation that enables active sensors to predict data of sleeping sensors. They combined temporal, spatial, and spatiotemporal correlations among sensor readings. However, their solution is based on static scheduling of nodes activity– similarly to all previous solutions. This may have dramatic consequence on the network's lifetime and long-term data precision. Our solution overcomes this shortcoming by dynamically selecting the working nodes using a generic random policy [13] that balances the network's load and distributes it amongst all the sensors.

Similar to our approach, Yoon et al., [14] explored the use of GANs for missing data imputation at the data collector level, but it does not learn from structural information included in the data. Seller and Sornin [15] considered data representation as a graph structure and proposed a method to automatically generate the graph by measuring the distance between data features. However, the generated graphs do not reflect the real structure of sensor networks, which are naturally organized as graphs using the sensor's communication and sensing ranges. Our work exploits nodes sensing coverage to build the graph structure.

III. Assumption, Formulation and Solution Overview

We consider a general scenario that consists of a network of sensor devices that monitor an area of interest and report their reading to a centralized data collector. The latter is generally implemented in a more powerful device such as a base-station or a cloud central station. We propose the system operates in epochs and in each one only a part of the network monitors the field while the remaing sensors go to sleep mode to save energy. All the sensors are supposed to have the same sensing ranges and a distributed random sensor scheduler to be used, such as LEACH [13], to dynamically schedule the sensors between active and sleeping. Let $X \in \{0, 1, *\}^N$ denote the vector representing the sensors' readings, where N is the number of sensors. The values 1 and 0 correspond to the presence and absence of an event, and $*$ represents a missing value due to the sensor being in the sleep mode. Let us denote by, A, the adjacency matrix representing the mutual coverage between sensors. A is built during the preprocessing phase that will be explained later. To express the missing data in X, we define a binary mask vector, $M \in \{0, 1\}^N$, where m_i indicates the presence or absence of a reading in x_i, i.e., $M_i = 0$ if $x_i = *$. The generator (described in Sec.IV-B) uses X, M, and the spatial relationships between the sensors reflected by

A to complete the missing values in X and produce the vector $\widehat{Y} \in \{0,1\}^N$ for the generated sensor readings that results from the learning of a mapping function of A, M, X. Deep learning methods are used for learning this mapping as described in the next section. The final output $Y \in \{0,1\}^N$ of inputted missing values is then calculated using M, where values from X are copied for entries i in which $M_i = 1$, and from $\widehat{Y}$ for the other entries.

IV. Solution Description

A. Preprocessing: Sensor Graph Construction

An undirected weighted graph is used to represent the sensor network, $(\mathcal{V}, \mathcal{E})$, where the set of vertices, $\mathcal{V}$, represents the N sensors and the set of edges, defined in $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$, represents mutual coverage links. A unit-disk sensing range based model is used to construct $\mathcal{E}$, such that $(i,j) \in \mathcal{E}$ iff the sensor j is within the sensing range of i, i.e., i can be covered by j. The graph is undirected since the sensors have the same sensing ranges. We denote by A the adjacency matrix that associates weights to the edges. A probabilistic sensing model is used to build A, which captures the effect of the environment and the distance on the sensing area. It relies on the "exponential decay with distance" model commonly used in the literature [8]. The weight of an existing edge, A_{ij}, is proportional to the probability P_{ij} that j covers i. It is given by: $A_{ij} = P_{ij} = e^{-\beta d_{ij}}$, where d_{ij} is the euclidean distance between v_i and v_j, $\beta \in [0,1]$ is the sensing capacity decay factor. It describes how fast the sensing decays with distance and depends on the sensor and the environment. Notice $A_{ij} = 0$ if there is no edge between i and j, and $A_{ii} = 1\ \forall i$.

B. GCN Generator

The GCN generator network generates the missing data by capturing the spatial dependencies between sensors. The output of the generator is defined as a function $f_g(X, M, A)$, where X, M, and A are respectively the features vector, the mask vector, and the adjacency matrix as defined in Sec.III. The GCN is based on the Kipf et al. [16] multi-layer model, in which each hidden layer is expressed as $H^{(l)} = f(H^{(l-1)}, A)$, where f is the propagation rule and $H^0 = X$. Following this propagation rule model, we define $Z = f_g(X, M, A) \in (0,1)^{N \times d}$, the output of a two-layer GCN as defined in Eq. 1.

$$Z = f_g(X, M, A) = \sigma(\widehat{A} ReLU(\widehat{A}(X \odot M)W^0)W^1), \quad (1)$$

where: $\widehat{A} = D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$, D is a matrix with elements defined as, $D_{ii} = \sum_{j \in \mathcal{N}(i)} A_{ij}$, which are calculated in the prepossessing phase, $\odot$ denotes the Hadamard product. W^0 and W^1 are trainable vectors: $W^0 \in \mathbb{R}^H$ is the first weight vector (connecting the input layer to hidden layer), $W^1 \in \mathbb{R}^H$ is weight vector connecting the hidden layer to output layer, while H the size of hidden unit. ReLU(.) and $\sigma(.)$ are REctified Linear Unit and sigmoid activation layer, respectively. The ReLU activation function is shown in practice to be efficient against the vanishing gradient problem. Moreover, we use a sigmoid function in the output layer of the GCN model since we consider two classes, i.e, the presence or absence of an event, and we include an intermediate dropout layer between the two GCN layers. Finally, notice that the identity matrix was not added to A when calculating $\widehat{A}$ as the graph already contains self-loops (Sec. IV-A).

The generator uses the information from the labels and its capacity to learn from structural data to predict the readings of the remaining sensors. The generated vector, Z, is compared to the values of active sensors in X, entries for which $m_i = 1$.

Given that the task is a binary classification, the binary cross-entropy loss is used to express the loss function. The loss function for the GCN network is defined by Eq. 2.

$$\mathcal{L}_g = -\sum_{i=0}^{N} x_i m_i \log z_i + (1 - x_i m_i) \log(1 - z_i). \quad (2)$$

In the deployment phase, Z is rounded to the binary vector $\widehat{Y}$ (values above 0.5 rounded to 1 and to 0 otherwise). However, the generator is combined with the critic during the training phase when the critic challenges the generator for the purpose of producing synthetic data samples closer to the real distribution. The critic network and the critic's training process are described in the next section.

C. GAN Critic Network

The GAN critic network adversarially challenges the GCN generator to help improving its outcome during the training process, in which the critic enhances the capability to differentiate the generated sensor readings from the real data while the generator aims to increase its capacity to produce sensor readings that are similar to the real data . We use the W-GAN approach of [17] based the *Wasserstein* distance (also called *Earth-mover* distance) between the real data distribution, p_r, and the generated data distribution, p_g.

The goal of the critic is to learn a function, say c, that allows the calculation of the *Wasserstein*.

The critic learns c during training and guarantees it is a continuous *1-Lipschitz* function that is continuously differentiable with a gradient's norm no more than1 everywhere, i.e, it satisfies $|c(x) - c(y)| \leq |x - y|, \forall x, \forall y$.

Gulrajani et al. [18] proposed a penalty on the gradient's norm to enforce this constrains when defining the *Wasserstein* distance. This is given in Eq. 3, where $\mathcal{L}_c{}'$ is the *Wasserstein* distance (loss) with gradient penalty, x and $\tilde{x}$ are sampled from p_r and p_g, respectively, λ is the weight (its common value is 10 [18]) and ∇ denotes the gradient of the function.

$$\mathcal{L}_c{}' = \mathop{\mathbb{E}}_{\tilde{x} \sim p_g}[c(\tilde{x})] - \mathop{\mathbb{E}}_{x \sim p_r}[c(x)] + \underbrace{\lambda \mathop{\mathbb{E}}_{\hat{x} \sim p_{\hat{x}}}[(\|\nabla_{\hat{x}} c(\hat{x})\| - 1)^2]}_{\text{gradient's penalty}}, \quad (3)$$

The gradient's norm is calculated for random samples $\hat{x} \sim p_{\hat{x}}$. That is, $p_{\hat{x}}$ is sampled uniformly along straight lines between pairs of points sampled from p_g and p_r, i.e., $\hat{x} = \epsilon x + (1-\epsilon)\tilde{x}$, $\epsilon \sim U[0,1]$.

While the critic is trained to minimize $\mathcal{L}_c{}'$, the generator is trained to minimize $-\mathop{\mathbb{E}}_{\tilde{x}\sim p_g}[c(\tilde{x})]$. Eq.(4) defines the global loss of the generator, $\mathcal{L}_g{}'$, which combines the GCN loss (Eq.(2)) with the adversarial loss. The hyperparameter ζ is included to balance between the two losses.

$$\mathcal{L}_g{}' = \mathcal{L}_g - \zeta \mathop{\mathbb{E}}_{\tilde{x}\sim p_g}[c(\tilde{x})] \tag{4}$$

Similarly to [18], [17], the critic is iteratively trained. The practically, $\mathop{\mathbb{E}}_{x\sim p_r}[c(x)]$ and $\mathop{\mathbb{E}}_{\tilde{x}\sim p_g}[c(\tilde{x})]$ are computed by sampling a random mini-batche, x, from real sensor reading. x and $\tilde{x}$ are fed to the critic network and the expectations are approximated with critic's output averages. We used the *Adam* optimization algorithm [19] for the gradient optimization of the two networks, which is more robust against the vanishing gradient problem.

V. Performance Evaluation and Experimental Results

Without loss of generality, we evaluate the performance of the proposed solution (we call herein AG-SensPred) in a smart building environment that serves as a use case for an application scenario. This is a typical IoT application with great potential to take advantage from the advanced machine learning techniques [20]. The *MERL* [21] real dataset was used in this experimental study. It was collected in an indoor setting and includes over 50 million of motion sensor events spanning over two years with milliseconds granularity. The models have been implemented using Pytorch[22] deep learning framework and GDL (Deep GraphLibrary)[23] for the generator's implementation on two GPU environments[1]. 80% of data was used for training and the remaining 20% for tests. In the following, the proposed solution is compared with the most relevant solutions of the literature, 1) JGD [9], 2) GCN [16], 3) GAIN [14], 4) GINN[15]. JGD uses Gaussian distribution, standard heuristics for the prediction of sleeping sensors' values. GCN uses a semi-supervised classification with GCN but without adversarial training function to infer sensor readings. GAIN uses GAN for data imputation but without graph structure, contrary to AG-SensPred. GINN shares many features with AG-SensPred, notably of the exploration of adversarially-trained GCN for data imputation. However, its context is different and the graph generation in GINN is not tailored to sensor networks.

A. Accuracy

We use the *F-score* as the accuracy metric that captures the recall as well as the precision [24]. Precision may be defined as the probability that an event is relevant given that it is predicted by the system, while recall is the probability of correctly predicting events. We varied the percentage of the active sensors that represents the percentage of the data used from the datasets as real data. Fig.1 shows the superiority of AG-SensPred with respect to this metric over all the compared solutions. Compared to GIIN, the results confirm that AG-SensPred's formulation for the weighted graph is effective for generating missing sensor data since it considers sensor locations and the sensing range. On the other hand, being superior to GCN confirms the usefulness of including adversarial training. The fact that the best three solutions with respect to this metric are those including adversarial training also confirms this. The performance of AG-SensPred is very close to GAIN when the percentage of active sensors is below 20% and above 70%. This is since both networks learn equally when a high or a low amount of data is available. However, our solution clearly shows better performance in the interval 20% to 70%. The results also show that almost all solutions (except GCN) have a similar and low F-score at 10%. This can be explained by the difficulty of training the models when a very small amount of data is used for learning. In a more realistic scenario, the active sensors (implicitly missing data) are typically in the middle range. For these values, AG-SensPred is clearly outperforming all solutions including GAIN. JGD has less varied performance. This confirms that event-based sensing data does not follow traditional probability distribution methods like Gaussian distributions.

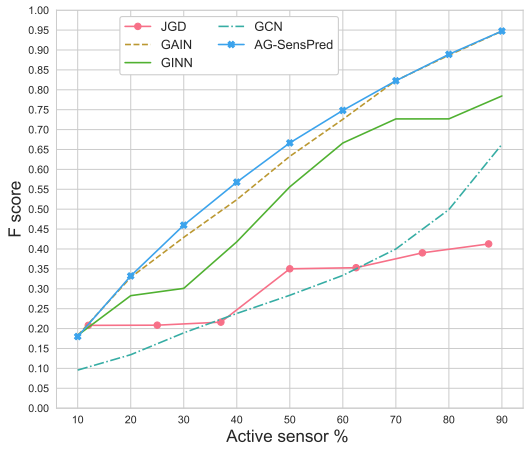

Fig. 1. F-score

B. Energy Conservation

We measured the energy in *watt second* consumed on average by one sensor. We used a hardware manufacturer datasheet of a wireless motion detector as a reference, which is powered with a *CR2032* coin cell battery of $3V$) and 240 mAh. It consumes 1.57 mA in active mode for 56.66 ms, 3.45 μA in standby mode, and 2.16 μA when shutdown [25]. Fig. 2 shows

[1]Nvidia GeForce RTX 2080 Ti and Nvidia Quadro RTX 6000.

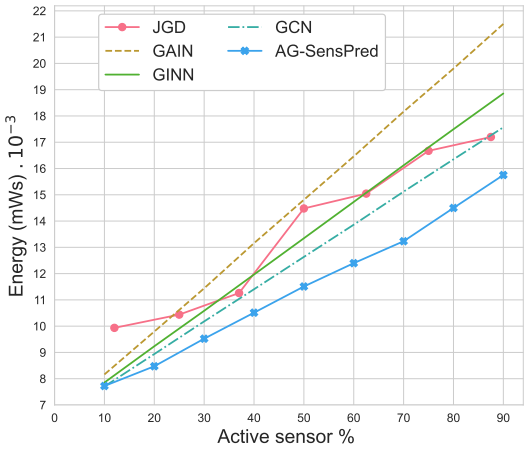

Fig. 2. Average consumed energy

a clear superiority of the AG-SensPred compared to all the other solutions and confirms it provides the best energy-saving. Moreover, the inevitable increase of the consumed energy with the percentage of active sensors is smooth for AG-SensPred.

VI. Conclusion and Future Work

Predicting sensor reading to preserve energy in event-based IoT applications has been dealt with in this paper and a new approach has been proposed. It saves the energy consumed for sensing and communications by deactivating part of the sensors and using the readings from the active sensors along with the spatial correlation of the network to generate the missing data. The proposed approach is holistic and does not rely on any duty-cycle policy. Instead, the sensors sleep and wake up randomly while an adversarially trained GCN generates the missing data. The proposed solution has been evaluated using a real dataset and compared against four state-of-the-art relevant solutions. The results showed that the proposed solution offers the best accuracy, as well as energy-saving. Future directions to this work include the quest for optimization techniques that increase spatiotemporal learning, e..g, by adding attention layers to extract both local and global relevant features from spatiotemporal data.

References

[1] R. C. Carrano, D. G. Passos, L. C. S. Magalhães, and C. V. N. de Albuquerque, "Survey and taxonomy of duty cycling mechanisms in wireless sensor networks," *IEEE Commun. Surv. Tutorials*, vol. 16, no. 1, pp. 181–194, 2014. [Online]. Available: https://doi.org/10.1109/SURV.2013.052213.00116

[2] M. Doudou, D. Djenouri, J. M. Barceló-Ordinas, and N. Badache, "Delay-efficient MAC protocol with traffic differentiation and run-time parameter adaptation for energy-constrained wireless sensor networks," *Wirel. Networks*, vol. 22, no. 2, pp. 467–490, 2016.

[3] D. Djenouri and I. Balasingham, "Traffic-differentiation-based modular qos localized routing for wireless sensor networks," *IEEE Trans. Mob. Comput.*, vol. 10, no. 6, pp. 797–809, 2011. [Online]. Available: https://doi.org/10.1109/TMC.2010.212

[4] D. Djenouri and M. Bagaa, "Energy-aware constrained relay node deployment for sustainable wireless sensor networks," *IEEE Trans. Sustain. Comput.*, vol. 2, no. 1, pp. 30–42, 2017.

[5] R. V. Prasad, S. Devasenapathy, V. S. Rao, and J. Vazifehdan, "Reincarnation in the ambiance: Devices and networks with energy harvesting," *IEEE Commun. Surv. Tutorials*, vol. 16, no. 1, pp. 195–213, 2014.

[6] H. Azarhava and J. M. Niya, "Energy efficient resource allocation in wireless energy harvesting sensor networks," *IEEE Wireless Communinactions Letters*, vol. 9, no. 7, pp. 1000–1003, 2020.

[7] M. Khiati and D. Djenouri, "Adaptive learning-enforced broadcast policy for solar energy harvesting wireless sensor networks," *Comput. Networks*, vol. 143, pp. 263–274, 2018.

[8] R. Laidi, D. Djenouri, and I. Balasingham, "On predicting sensor readings with sequence modeling and reinforcement learning for energy-efficient IoT applications," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–12, 2022.

[9] S. Silvestri, R. Urgaonkar, M. Zafer, and B. J. Ko, "A framework for the inference of sensing measurements based on correlation," *ACM Transactions on Sensor Networks*, vol. 15, no. 1, pp. 4:1–4:28, Dec 2018.

[10] H. C. D. V. Bui and C. V. Nguyen, "Asymmetrical pwm scheme eliminating duty cycle limitation in input-parallel output-series DC–DC converter," *IEEE Transactions on Power Electronics*, vol. 37, no. 3, pp. 485–2490, March 2022.

[11] D. G. Martins, B. Boris, and O. Simon, "A survey about prediction-based data reduction in wireless sensor networks," *ACM Computing Surveys*, vol. 49, no. 3, pp. 58:1–58:35, Nov. 2016.

[12] F. Emekci, S. E. Tuna, D. Agrawal, and A. E. Abbadi, "Binocular: A system monitoring framework," in *Proceeedings of the 1st International Workshop on Data Management for Sensor Networks: In Conjunction with VLDB 2004*, ser. DMSN '04, 2004, pp. 5–9.

[13] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Trans. Wirel. Commun.*, vol. 1, no. 4, pp. 660–670, 2002.

[14] J. Yoon, J. Jordon, and M. van der Schaar, "GAIN: Missing data imputation using generative adversarial nets," in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 80, 2018, pp. 5689–5698.

[15] I. Spinelli, S. Scardapane, and A. Uncini, "Missing data imputation with adversarially-trained graph convolutional networks," *Neural Networks*, vol. 129, pp. 249–260, 2020.

[16] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations (ICLR)*, 2017.

[17] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *arXiv preprint arXiv:1701.07875*, 2017.

[18] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Advances in neural information processing systems*, 2017, pp. 5767–5777.

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[20] D. Djenouri, R. Laidi, Y. Djenouri, and I. Balasingham, "Machine learning for smart building applications: Review and taxonomy," *ACM Comput. Surv.*, vol. 52, no. 2, pp. 24:1–24:36, 2019.

[21] C. Wren, Y. Ivanov, D. Leigh, and J. Westhues, "The merl motion detector dataset," in *Workshop on Massive Datasets (MD)*, Nov. 2007, pp. 10–14. [Online]. Available: https://www.merl.com/publications/TR2007-069

[22] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," in *NIPS Autodiff Workshop*, 2017.

[23] M. Wang, D. Zheng, Z. Ye, Q. Gan, M. Li, X. Song, J. Zhou, C. Ma, L. Yu, Y. Gai, T. Xiao, T. He, G. Karypis, J. Li, and Z. Zhang, "Deep graph library: A graph-centric, highly-performant package for graph neural networks," *arXiv preprint arXiv:1909.01315*, 2019.

[24] C. Goutte and E. Gaussier, "A probabilistic interpretation of precision, recall and f-score, with implication for evaluation," in *Advances in Information Retrieval*, D. E. Losada and J. M. Fernández-Luna, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 345–359.

[25] "Ultra-low-power wireless pir motion detector for cost-optimized systems reference design," https://www.ti.com/lit/ug/tiducu5/tiducu5.pdf, accessed: 2019-08-06.