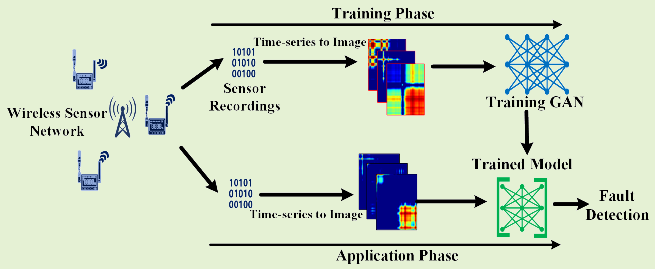
Wasserstein GAN-based Digital Twin Inspired Model for Early Drift Fault Detection in Wireless Sensor Networks

Md. Nazmul Hasan, Sana Ullah Jan, and Insoo Koo

***Abstract*—In this Internet of Things (IoT) era, the number of devices capable of sensing their surroundings is increasing day by day. Based on the data from these devices, numerous services and systems are now offered where critical decisions depend on the data collected by sensors. Therefore, error-free data are most desirable, but due to extreme operating environments, the possibility of faults occurring in sensors is high. So, detecting faults in data obtained by sensors is important. In this paper, a digital twin inspired detection approach is proposed, and its ability to detect a single type of fault in several sensor is analyzed. The digital equivalent of the sensor is developed using a Generative Adversarial Network (GAN). As GANs inherently performs well with images, Gramian Angular Field (GAF) encoding is used to convert timeseries data to image. The GAF encoding preserves the temporal relations of the timeseries data. The GAN is trained with the GAF images. The trained GAN model acts as the virtual representation of the sensor, and the discriminator network of the GAN model, once it has learned the pattern of normal data, is used as the fault detector. The performance of the virtual sensor is promising because it successfully generates data for normal conditions. The best fault detection accuracy achieved by the proposed model is 98.7%, which makes this GAN-based digital twin inspired approach a promising candidate for sensor fault detection.**



***Index Terms*—sensor faults, digital twin inspired model, Generative Adversarial Network (GAN), Gramian Angular Field (GAF), deep learning**

I. Introduction

THE pervasive deployment of Internet of Things (IoT) devices fuses the physical and digital worlds. This integration creates more automated and interactive surroundings. For instance, the inclusion of sensing and communication capabilities in real-world physical objects has transformed them into smart entities. Utilizing the wireless sensor network (WSN) and advanced communications protocols, these smart objects gather data from the vicinity and transfer them to central nodes or processing units [1]. Finally, by utilizing cloud-based services and data analytics algorithms, a myriad of IoT applications in diversified fields have been proposed. In the agricultural sector, IoT-based frameworks can be used for environment and soil condition monitoring on a large-scale farm [2], [3] for plant disease detection, pest control [4], and precise irrigation for efficient water management [5]. Through such approaches, farming can be possible in harsh areas through the efficient use of natural resources. In power systems, the IoT has a variety of applications, such as monitoring parameters in a smart grid network for reliable operations [6], for power consumption analysis and predictions in households [7], plus power theft detection and notification [8]. Proper integration of IoT- and cloud-based services can promote better living environments through automated lighting systems, intelligent transportation systems, smart parking management, as well as waste and power management [9]. In healthcare, the IoT shows great aspect in personalized patient care and monitoring [10]. Many innovative healthcare schemes have been proposed to monitor, control, and prevent Covid-19 outbreaks [11].

The recent research and implementation trends indicate that IoT-based services will continue to thrive in numerous fields. However, one critical issue that determines the quality of service from such IoT applications is the condition or quality of the data communicated by sensors. In real scenarios, there are heterogeneous physical parameters to be sensed, and thus, sensors might have to be placed in harsh and extreme conditions in either nature or industry [12]. In many cases, tiny low-cost sensors become defective due to aging, hardware malfunction, faulty installation, or from environmental impacts such as high temperatures or humid weather, which

“This work was supported by the 2023 Research Fund of the University of Ulsan.”

Md. Nazmul Hasan and Insoo Koo are with the Department of Electrical, Electronic and Computer Engineering, University of Ulsan, Ulsan 44610, South Korea (e-mail:hasan01@mail.ulsan.ac.kr; iskoo@ulsan.ac.kr).

Sana Ullah Jan is with School of Computing, Engineering and Built Environment, Edinburgh Napier University, UK (e-mail: s.jan@napier.ac.uk).

leads to generation of anomalous data by a sensor. Aryal et al. [13] defined anomaly in data as instances that do not conform to a normal pattern and that show significant deviation from a general trend. Such anomalous or faulty data can be caused by sensor network issues; or the sensor itself might be faulty due to internal and external factors [14]. In most of the IoT-based applications or services, decision and control instructions depend on data collected by sensors. Hence, if faulty data or sensors remain undetected, the system could become inaccurate and unreliable, and there could be serious consequences [15]. Primarily, sensor faults can be classified into two broad categories: incipient faults and abrupt faults [16]. Incipient faults develop slowly and detecting them is difficult in the initial stages of occurrences. However, the system may initially work well, but as time passes could suffer serious failures. On the other hand, abrupt faults are sudden phenomena that can last for a brief moment or that occur intermittently over time. This type of sensor fault is easily identifiable compared to the incipient fault. Common sensor faults and the possible contexts of an occurrence are listed in Table I

TABLE I: Common sensor faults and their causes

Type	Name of fault	Reasons of occurrence
Incipient	Bias fault	External interference, presence of bias voltage/current
	Drift fault	Calibration defect, change in sensor circuit parameters
	Gain fault	Aging, change in sensor circuit parameters
Abrupt	Spike fault	Defective hardware, communication defects, battery issues
	Stuck fault	Defective hardware, communications defects, clipping defects, battery issues
	Random fault	Sensor circuit failure, extreme condition (sensed variable exceeds sensor range)
	Short/open circuit fault	Poor circuit connection, hardware failure
	Noise fault	Battery issues, out-of-range communication, noise from circuit elements or external sources

Wu and Zaho broadly separated fault analysis techniques into three clusters: model-based, knowledge-based, and data-driven approaches [17]. In model-based methods, an analytical model is created utilizing the physical rules or specifications of the underlying practical system. Fault diagnosis is then performed by comparing real output from the system with output estimated by a model. Any inconsistency between these two responses is considered a possible fault. Xiong et al. proposed a model-based sensor fault scheme for lithium-ion battery packs from a joint estimation model of the battery pack using recursive least squares and unscented Kalman filters [18]. Those authors estimated the state of the charge in the battery pack, and then compared it with a reference to determine fault occurrences. Lyapunov theory-based modeling was also recently used by a few researchers for fault diagnosis. For example, in [19], detection and identification of intermittent, incipient, simultaneous, and abrupt faults in aircraft sensors was performed using Lyapunov function theory. The authors observed that this approach can result in faster fault detection, compared to an extended Kalman filter-based method. Multiple sensor fault analysis for internal combustion engines was investigated based on fuzzy Lyapunov modeling [20]. The proposed model has two sublayers to model and estimate a preprocessed sensor signal. In the initial stage, a fuzzy-assisted Gaussian-autoregressive-Laguerre approach models the signal, and later on, a fuzzy-Lyapunov-based computed ratio observer results in better estimation of the signal. Finally, the residual between the estimation and the original signal is used to determine the sensor fault. This approach has shown good fault classification performance when the signal change is minimal for different sensor faults. Observer-based modeling has also gained attention in recent times for sensor fault detection and isolation, especially in power systems [21] and in railway traction drives [22].

In many situations, the underlying systems are very complicated and are usually a combination of many subsystems. Therefore, constructing a mathematical model for them is tedious and difficult, and could provide inaccurate estimation of the output. For such scenarios, a knowledge-based sensor fault analysis approach is proposed. This type of method does not require a mathematical representation of the corresponding system. Instead, it uses historical data regarding faults, and domain expertise to determine and analyze sensor faults. Expert-system fault analysis for ship electronic power systems was proposed [23]. The expert system receives fault features through a human-computer interaction interface that loads some fuzzy-based rules, compares the features with a knowledge base, and finally, interprets the analysis for the user. A similar type of expert system approach was adopted by researchers for machinery fault analysis in the agricultural sector [24]. Fuzzy-rule-based expert systems are mostly used in sensor fault analysis [25].

In recent times, a wide range of networked sensor deployments with cloud connectivity have provided more access to sensor data, which really boosted the application of data-driven methods in sensor fault analysis. Data-driven approaches do not require mathematical models or a predefined knowledge base, as earlier methods did; rather, these methods can use raw data or processed data directly, and the algorithms can automatically identify the fault pattern with high accuracy. Among machine learning algorithms, the support vector machine (SVM) is widely used for sensor fault analysis. A classification task for a number of possible faults in temperature sensor data was performed using an SVM aided by a time domain feature [26]. Time-frequency-based empirical mode decomposition was used as a feature extraction method, and an SVM was used for sensor fault diagnosis in a gas turbine system [27]. In some recent work, various optimization techniques, such as α-Gray Wolf Optimization [28] and the Baum-Welch algorithm [29] were integrated with an SVM to make fault diagnosis more robust. Moreover, previously practiced model-based approaches are sometimes combined with data-driven methods in a hybrid model for fault analysis. Other popular machine learning algorithms like K-Nearest Neighbors (KNN), Random Forest (RF)-based classifiers, Gaussian Naive Bayes (GNB), and Multi-Layer Perceptron (MLP) have been utilized in sensor fault classification and analysis tasks [30]–[32]. With increased amounts of relevant data, various deep learning models now engage in sensor fault analysis. Although deep neural networks (DNNs), convolutional neural networks (CNNs), and other deep learning architectures have the ability to extract meaningful features from data, the majority of the literature has observed that additional methods are integrated before data preprocessing. Time-frequency-based preprocessing was used in conjugation with a CNN for sensor fault diagnosis in several studies. In [33], faulty and non-faulty sensor data from an aero-engine control system were first converted into scalograms, and then, different faults were classified from the scalograms by using a CNN model. A similar approach was proposed in [34] where the authors converted signals collected from a drone sensor into time-frequency images using short-time Fourier transform, and they then used the images for training a four-layer CNN classifier model. Another modified CNN architecture was investigated in [35] where faults in a hydrogen gas sensor were inspected. The CNN-based classifier model was modified by including a Morlet wavelet basis function as an activation function for two intermediate convolutional layers. Other deep learning models such as autoencoders and long short-term memory (LSTM)-based architectures were applied to analyzing faults in sensors, and in [36], a combined framework of a CNN and a convolutional autoencoder (CAE) was proposed for sensor fault detection and reconstruction. The authors used a CNN classifier to detect one of the four types of sensor faults, and in the subsequent stages, an individual CAE was used for each fault type to reconstruct a signal corresponding to the faulty signal. In the final stage, a post-processing method utilizes a threshold value to localize the faulty sensor, and it then replaces the faulty signal with the signal predicted from the CAE. Jan et al. demonstrated a sensor fault detection and analysis scheme in [37] where a stacked autoencoder (SAE) is used to extract a lower dimensional representation of the input signal to be used by an SVM classifier for fault detection, and later, a fuzzy DNN is used for further diagnosis of faults. Darvishi et at, in their recent work [38] proposed a modular architecture for detecting sensor bias fault. The modular architecture includes an estimator layer that estimates individual sensor measurements based upon the available sensors present in the physical system. The estimators are realized using NN-based blocks. Later, the estimated measurement is compared with the actual measurement to detect the presence of the fault. In the final layer there is a classifier utilizing the residual signal to provide a decision vector for identifying the faulty sensor. The same authors demonstrated an enhanced version of their modular sensor fault detection scheme in [39], where the structure is modified by adding a predictor layer in parallel to the estimator layer in the previous architecture [38]. A controller module has also been added to isolate any faulty sensor and prevent its data in the estimation process. In the later, experiment authors considered four types of faults. The newly developed approach demonstrated better fault detection as compared to the baseline models such as Auto-Encoder model and their previous model [38] when tested with real world data.

In very recent times, a new method for fault or anomaly detection has been applied by constructing the digital equivalent of a physical system [40]. This approach involves a digital representation of the underlying system, which is termed a Digital Twin (DT) that usually simulates normal operating behavior. Deciding on the occurrence of a fault is made when the residual between the signal obtained from the DT and the real measured signal exceeds a defined threshold. Construction of the DT is often based on the physical laws on which a particular component of a system operates. As in [41] a DT estimator of a photovoltaic energy conversion unit is developed. The design of the estimator for a single energy conversion unit involves a convoluted modeling steps. In [42] a DT based fault detection is proposed for bearing fault detection. In this case also authors combines signal approximation and signal estimation stages to create a DT. Both the approximation and estimation stages have several complex subsystems. It is evident that, this approaches gets severely complex, and is error prone when the system under consideration consists of a number of sub-systems in which responses depend on each other. However, using DNNs to generate DTs has recently been tapped by some researchers. A relatively new concept is the Generative Adversarial Network (GAN) approaches that seem promising in creating DTs. The capability of generating artificial samples very similar to the training examples provided is the key idea behind creating DTs using GANs. In [43], Xu et al. proposed an anomaly-detection framework based on a DT where the DT is created from a GAN architecture. The generator segment is composed of a graph convolutional network (GCN), a polling layer, and LSTM layers, and the discriminator segment is a multi-class classifier. A conditional-style GAN architecture was presented in [44] to create a DT for a machining model that generates time-series data representing a vibration signal. Booyse et al. demonstrated that DTs based on deep generative models can be used in fault detection and analysis [45].

A. Contribution of this Work

One aspect of DT is that it can represent the state of its corresponding physical system with bidirectional interaction with the physical entity. We take our inspiration from this fact that if we could develop a model that realizes only the normal operating condition of a physical sensor as a DT does, we could utilize it to detect sensor fault. Although our proposed model is not a complete DT but it is inspired from the ability of the DT to mimic the physical system. In our work, we construct a virtual twin of normal state of a physical sensor using the generative adversarial network architecture. The major contributions of this paper are as follows:

- We developed a virtual sensor motivated from DT concept using the Wasserstein generative adversarial network architecture. We demonstrated that with this GAN model the normal operating behaviour of a physical sensor can be realized.
- We implemented a fault detection method using the discriminator segment of the trained GAN model. We showed in our results the the fault detection method

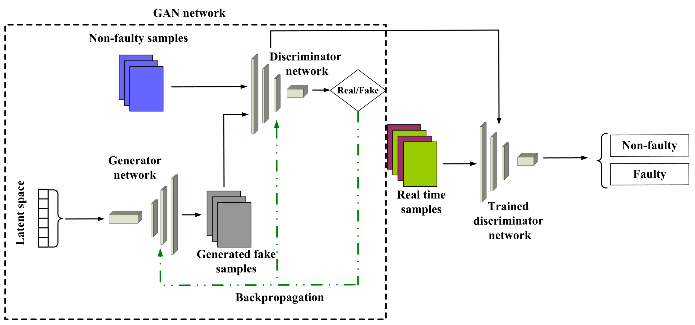


Fig. 1: Fault detection using DT inspired GAN model.

can produce consistent performance over three different datasets.

- Our proposed model proved that it is possible to detect fault with a good accuracy without using any data or samples corresponding to the fault instances. As the GAN model is trained with only the normal state data of a sensor which is easily available. The trained model then can differentiate a faulty instance as it appears for the first time with descent accuracy. We believe that this is significant because in real scenario it is quite difficult to collect a sufficient number of fault samples from sensors.

The rest of the paper is organized as follows: **Section II** provides a detailed description about the proposed fault detection approach. In **Section III** the the steps involved in dataset preparation is discussed. The performance of our proposed model is presented and explained in **Section IV**. Finally, concluding remarks are provided in **Section V**.

II. Proposed Sensor Fault Detection Approach

In this section, we briefly describe our proposed sensor fault detection and classification approach based on a generative model. In the first stage, a virtual representation of the normal state of the sensor is developed using a GAN model that will be able to generate synthetic data similar to the normal state of the sensor. Since the GAN output closely resembles the normal state of the sensor, it is considered a virtual representation of the sensor at its normal state. The capability of the digital representation will be used to detect the sensor fault. After detecting a fault, a CNN classifies the type of fault. A graphical representation of our approach is in Fig.1. The GAN-based model is trained only on the normal sensor signal and as soon as the model receives any abnormal or faulty pattern in the data it can distinguish it as fault with high accuracy. As the model does not require any historical fault data for training therefore it does not need to wait for several fault instances to occur. It can detect fault at the early instances of the fault occurrence. Therefore, it can be considered as early detection of fault.

After being proposed by Goodfellow et al. [46], the GAN became the most active area of research in deep learning and has been used mostly for image-related applications such as image synthesis, image-to-image translation, reconstruction of missing segments in an image, upscaling, plus video prediction and generation. The core of the GAN is composed of two competitive entities (the generator and the discriminator) trying to deceive one another. Both generator and discriminator are multi-layer convolutional neural networks or fully-connected neural networks. The generator network creates fake data, and the discriminator's task is to differentiate between fake and original instances. The ultimate goal of the generator network is to learn the distribution of the original data through the training process. In a properly trained GAN, the generator can produce realistic instances to such an extent that the discriminator will find it difficult to distinguish whether data are from the original dataset or are synthetic presented by the generator. The training procedure of the GAN makes it possible for the generator to achieve accuracy in producing realistic data. Because the GAN consists of two separate networks, they are trained in an alternative manner; the trainable parameters of the generator are kept constant while training the discriminator, and a similar approach is followed for the discriminator when the generator is trained. Since the discriminator is generally a binary classifier, it uses binary cross entropy as the loss function, which is defined in (1).

$$\mathbb{E}_{(X\sim P(X))}\left[\log D\left(X\right)\right]+\mathbb{E}_{(Z\sim P(Z))}\left[\log\left(1-D\left(G\left(Z\right)\right)\right)\right] \tag{1}$$

where Z is a random vector from the latent space, X is the original training data, $D(X)$ is the output from the discriminator for real training data, and $D(G(Z))$ is the output of the discriminator when the input comes from the generator, $G(Z)$. The discriminator's goal is to maximize this loss. On the other hand, the generator tries to generate data as similar as possible to the original data; in other words, the discriminator will label the output from generator $G(Z)$ as if it is an original instance of the data. Thus, the generator tries to minimize the following cost function:

$$\mathbb{E}_{(Z\sim P(Z))}[\log(1-D(G(Z)))] \tag{2}$$

Assuming θ_G and θ_D to be generator and discriminator parameters, respectively, the complete loss function of the GAN network mentioned in [46] is,

$$\underset{\theta_G}{Min}\underset{\theta_D}{Max}\left[\mathbb{E}_{(X\sim P(X))}\left[\log D\left(X\right)\right]+\mathbb{E}_{(Z\sim P(Z))}\left[\log\left(1-D\left(G\left(Z\right)\right)\right)\right]\right] \tag{3}$$

Upon successful training, a GAN can generate data from random numbers sampled from the latent space, Z, that have a very similar distribution to the original data used for training. Generative models inherently perform better with images, and our approach transforms the sensor signal into images using a Gramian Angular Field (GAF) encoding technique in the first stage. As a first attempt in this work, only the drift fault was considered, and thus, sensor-to-image encoding resulted in two classes of image: one corresponding to the normal sensor signal and the other corresponding to the drift fault signal. In the next stage, the GAN was trained using only the images obtained for the non-faulty signal. Once the GAN is successfully trained, it can generate images similar to those seen during the training period. During the training process, the discriminator is supposed to learn the features of real training images to successfully distinguish them from the images produced by the generator in the initial phase of the training. A trained discriminator is expected to distinguish an image that differs from the images it saw while training. Now, an image that corresponds to some type of fault signal is not the same as the image from the normal condition, and because the trained discriminator knows the pattern of a non-fault image, it will discriminate images from a drift fault. In this way, the GAN can be used for detecting faults in sensor signals. Therefore, in our proposed method a DT inspired model of a humidity sensor is realized based on the GAN that replicates the normal state of the sensor, and the digital version is in turn used to detect anomalies in the sensor signals.

A. GAN Implementation

Although the concept of the GAN is exciting, and promises many novel applications, it is very difficult to train a GAN model. In some cases, loss of the generator and discriminator may exhibit an oscillating pattern, rather than converging to some value over the course of the training. Due to this oscillating loss problem, the model parameters do not stabilize, and the model does not provide good output. Mode collapse is another common challenge in GAN training, which occurs when the generator settles itself into generating a small number of similar outputs. In this case, the generator might make the discriminator believe the output is real output, but the generator is not able to generate a real data distribution, which is more diverse and complex, rather than the limited set it generates when mode collapse occurs. Additionally, the vanishing gradient problem arises when the discriminator is performing too perfectly, which causes the generator's gradient to diminish, which in turns prevents it from learning.

To provide stable GAN training, several approaches were proposed over the past few years. In our work, we utilized the Wasserstein GAN-Gradient Penalty (WGAN-GP) [47], which proved efficient in many complex GAN applications. The WGAN-GP is an intelligent modification of the original Wasserstein GAN (WGAN) first proposed in [48]. The WGAN is different from the standard GAN from several aspects; first, the WGAN uses a different loss function called the Wasserstein loss, which is derived from the Earth Mover's Distance metric, and can resolve the vanishing gradient problem. The discriminator and the generator loss function proposed in [48] are presented in equations (4) and (5).

$$\nabla_w \frac{1}{m} \sum_{i=1}^{m} \left[f\left(x^{(i)}\right) - f\left(G\left(z^{(i)}\right)\right) \right] \tag{4}$$

$$\nabla_\theta \frac{1}{m} \sum_{i=1}^{m} f\left(G\left(z^{(i)}\right)\right) \tag{5}$$

In the WGAN, generator parameters are updated after training the discriminator multiple times, which is not the case for a standard GAN, and while training the WGAN, the real data are labeled as 1 and the fake data are labeled as -1. Finally, unlike the standard GAN, there is no sigmoid activation used in the final layer of the WGAN discriminator. In the absence of sigmoid activation, the output of the WGAN discriminator ranges between $-\infty$ to ∞ rather than the usual 0 to 1 range. This large loss value is not desirable in a neural network, and to trace this unsettling loss, the authors in [48] introduced the concept of weight clipping the discriminator network to enforce the Lipschitz constraint. Although introducing weight clipping improved training stability and performance from the WGAN, the authors in [48] mentioned that this approach is not a good way, because the gradient value of the discriminator is changed and without accurate gradients, the generator weights cannot be updated effectively. This issue with the WGAN was addressed in [47] by introducing the gradient penalty into the discriminator loss function. In spite of clipping the weights of the discriminator, the WGAN-GP approach penalizes the discriminator if the gradient norm deviates from 1. The modified loss value extended with the gradient penalty is defined as follows:

$$\begin{gathered} L = \underbrace{\underset{\tilde{x}\sim\mathbb{P}_g}{\mathbb{E}}\left[D\left(\tilde{x}\right)\right] - \underset{\tilde{x}\sim\mathbb{P}_r}{\mathbb{E}}\left[D\left(x\right)\right] +}_{\text{Original critic loss}} \\ \underbrace{\lambda \underset{\hat{x}\sim\mathbb{P}_{\hat{x}}}{\mathbb{E}}\left[\left(\left\|\nabla_{\hat{x}} D\left(\hat{x}\right)\right\|_2 - 1\right)^2\right]}_{\text{Gradient penalty}} \end{gathered} \tag{6}$$

Here, $\hat{x}$ is sampled from $\tilde{x}$ and x with $t-$ uniformly sampled between 0 to 1. The inclusion of this penalty term improved the quality of the generated samples by the GAN and it showed more stability while converging as compared to the DCGAN [47].

Because the WGAN-GP proved superior to the traditional GAN and WGAN, we used this approach in our work to realize the GAN-based DT inspired model of the sensor. The generator and discriminator both have a DCGAN-like architecture, and each consists of five convolutional layers. We adopted the training approach described in the original paper. Table II describes the algorithm followed to implement the WGAN-GP model.

TABLE II: WGAN-GP implementation

```
Initialize: The gradient penalty coefficient $\lambda$, the number of critic, $n_{critic}$, batch size $m$, optimizer hyperparameters $(\alpha, \beta_1, \beta_2)$, Discriminator parameters $w_0$, generator parameters $\theta_0$
while $\theta$ has not converge do
    for $i$ = 1,...,$n_{critic}$ do
        for $j$=1,...,$m$ do
            Sample real data $x \sim \mathbb{P}_r$, latent variable $z \sim p(z)$,
            a random number $\epsilon \sim U[0,1]$.
            $\tilde{x} \leftarrow G_\theta(z)$
            $\hat{x} \leftarrow \epsilon x + (1-\epsilon)\tilde{x}$
            $L^{(i)} \leftarrow D_w(\tilde{x}) - D_w(x) + \lambda(\|\nabla_{\hat{x}} D_w(\hat{x})\|_2 - 1)^2$
        end for
        $w \leftarrow Adam\left(\nabla_w \frac{1}{m}\sum_{i=1}^{m} L^{(i)}, w, \alpha, \beta_1, \beta_2\right)$
    end for
    Sample a batch of latent variables $\left\{z^{(i)}\right\}_{i=1}^{m} \sim p(z)$
    $\theta \leftarrow Adam\left(\nabla_\theta \frac{1}{m}\sum_{i=1}^{m} -D_w(G_\theta(z)), \theta, \alpha, \beta_1, \beta_2\right)$
end while
```

The values of gradient penalty coefficient λ and learning rate α is set to 10 and 0.0001 as used in the original paper. The batch size m is selected as 32 and the number of critics n_{critic} is chosen as 3. The optimizer hyperparameters β_1, β_2 both set to 0.5

III. Data Preparation

For implementing machine learning or deep learning-based models, it is desirable to use actual data from real physi-

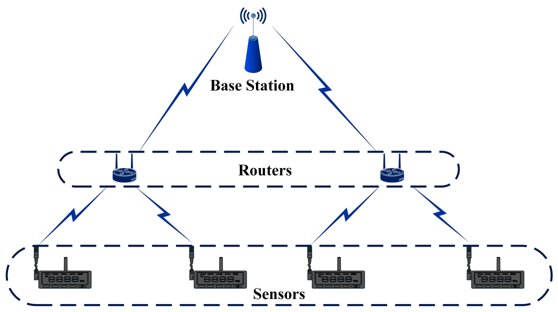


Fig. 2: WSN network architecture for data collection.

cal scenarios. For sensor fault analysis task, a dataset that consists of both non-faulty and fault-contaminated sensor measurements would be appropriate for model training and testing. However, there is no open dataset that provides sensor recordings affected by sensor faults. Therefore, to create such a dataset we synthetically injected faults into healthy sensor readings, this approach of artificial fault insertion is adopted by several researchers [32], [49]–[51].

The dataset we used for healthy sensor recordings is a publicly available wireless sensor network dataset that contains temperature and humidity measurements [52]. TinyOS-based motes collected the temperature and humidity readings from the vicinity of the sensors. The authors designed two network scenarios for collecting data (a single-hop network and a multi-hop network), and data were collected from both indoor and outdoor environments. In the single-hop network, two indoor sensors and two outdoor sensors measured temperature and humidity at the same time, transmitting readings to a base station. For the multi-hop network, the sensor nodes were placed so they covered a large area resulting in a large WSN. However, due to the limited transmission distances of individual sensor nodes, intermediate routers were used between the nodes and the base station, which then transmitted data to a base station as depicted in in Fig. 2

Like the single-hop arrangement, four sensors were used in the multi-hop network. To control data traffic, each sensor created a packet of 10 temperature and humidity readings to send to the base station. The data were collected for six hours at a sampling interval of five seconds. During the data collection process, an anomaly was induced in the signals by using a water kettle near one indoor sensor and near one outdoor sensor to artificially change the temperature and humidity readings. However, in our work we carefully discarded the anomalous instances. In this work, we considered the multi-hop scenario but took only a single sensor into account. In this case, few of the measurements made by the sensor could be subjected to faults, sometimes referred to as first-order anomalies [52]. We propose implementing the fault detection and analysis task at the intermediate router, as depicted in Fig. 3. This will help the base station to discard faulty observations and to refrain from making any decisions based on the faulty data.

The steps of creating additional samples from the reading of a sensor is listed in Table III. As the first step for creating the dataset, the sensor signal is segmented to create a number

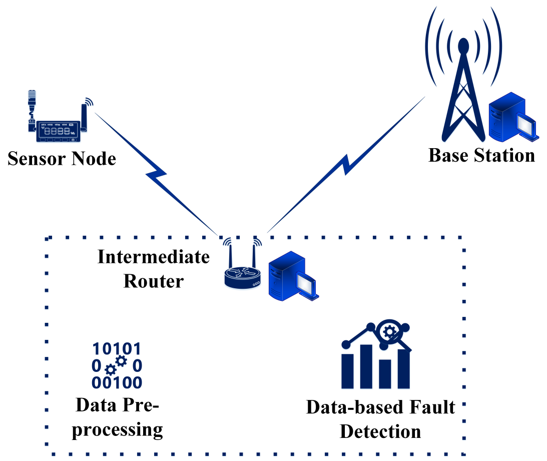


Fig. 3: Overall system structure.

TABLE III: Additional sample creation

```
N: length of original data stream, D
w_n: window length
i: sample index
set i = 0
while $i \times w_n \leq N$ do:
    Obtain the $i^{th}$ segment of no fault signal,
    $S_i = D[i \times w_n : i \times w_n + w_n]$
    $i = i + 1$
end while
Increase the number of samples to $M$
for $j = 0 : M$ do:
    Choose a random instance from $S_i$
    Add zero mean, low variance random signal with $S_i$ to create
    additional samples:
    $S_j = S_i + \mathcal{H}_n(0, \mu)$
end for
```

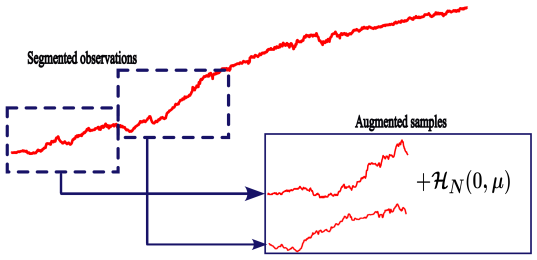


Fig. 4: Additional sample generation.

of samples. The samples created through this segmentation process is not enough for deep learning models. Therefore, more additional samples are created by adding zero-mean and low-variance signals with the segmented sensor signal instances. The concept is depicted in Fig. 4

A. Synthetic Fault Injection

After generating samples through the segmentation process mentioned above, sensor faults were artificially injected into the original samples. Here, synthetic fault injection process for common sensor faults are discussed briefly.

1) Drift fault: The drift fault is one of the common faults in sensors, where the sensor measurement changes in a linear fashion over time. This deviation in measurements could occur due to external factors or changes in circuit parameters. Such faulty signals can be synthetically generated by adding a linearly increasing bias signal to a healthy sensor signal. A sensor signal of length N that is affected by a drift fault, $S_{drift}(N)$, can be mathematically expressed with (7), where $S_{healthy}(N)$ is the uncontaminated signal, $H_N(0,\mu)$ denotes a zero-mean noise signal having a variance of μ, and β_N is the drift parameter that increases linearly over time:

$$S_{drift}(N) = S_{healthy}(N) + \mathcal{H}_N\ (0,\mu) + \beta_N \qquad (7)$$

Here, β_N is the drift that increases linearly with time and that can be realized from $\beta = \alpha \times N$ where α represents the amount of drift, and N is the index of the data point.

2) Spike fault: In this case, periodic high-amplitude spikes are observed in the sensor signal. This fault could occur due to defects in the sensor hardware or battery, from an external environmental phenomenon, or from defects in communication links. Mathematically, this fault can be defined as

$$S^N_{spike} = S^N_{healthy} + \delta_N,$$
$$\delta_N = \begin{cases} v, & N = i \times \tau,\ i = (1, 2,\ \ldots\ldots), \tau = constant \\ 0, & otherwise \end{cases} \qquad (8)$$

3) Bias fault: The sensor signal is said to be contaminated with a bias fault when the sensor output shifts to a value higher than a normal value. This fault is generated synthetically by adding a constant bias term to the normal readings of the sensor, defined as follows:

$$S^N_{bias} = S^N_{healthy} + v, \qquad v = constant \qquad (9)$$

4) Stuck fault: If the sensor readings show zero variation for a considerably larger number of samples than expected, it is assumed that the sensor data are erratic, called a stuck fault. This could happen from clipping of signals, due to hardware, or from battery malfunctions:

$$S^N_{stuck} = v, \qquad v = constant \qquad (10)$$

This fault is generated synthetically by fixing a healthy sensor signal to some offset value after some random period of time.

5) Precision fault: This fault is present when the variance increases above that of the normal scenario. To obtain precision fault samples, a random signal with a variance larger than a healthy signal is added to the original measurements:

$$S^N_{precision} = S^N_{healthy} + \ddot{\mathcal{N}}\ (\mu = 0,\ \sigma^2 = 0,\ \sigma^2 \gg \sigma^2_{healthy} \qquad (11)$$

A visual representation of common sensor faults is presented in Fig. 5

Some observations following the data generation process as discussed are presented in Fig. 6(a). In Fig. 6(b), the state of the sensor data after a synthetic drift fault injection is displayed. It has already been stated that only the drift fault was considered in this work, and therefore, we can see from the figure that the amplitude of the drift-injected signal increased over time. The fault initiation starts from the middle of the measurement, and at the beginning the severity of the fault is negligible; however, the drift increases over time, and at the end of the measurement window, the difference between a no-fault sensor signal and a drift-injected signal is at a maximum. The drift parameter was set to 0.05 for this experiment. In the later stage, these faulty and no-fault data were converted into images using the GAF encoding technique.

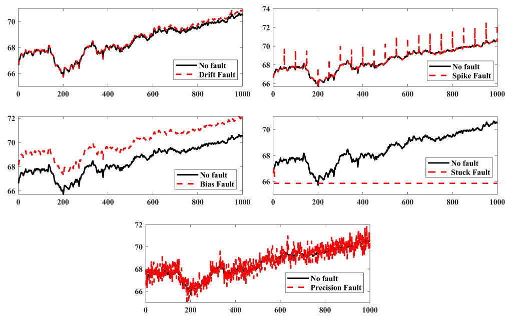
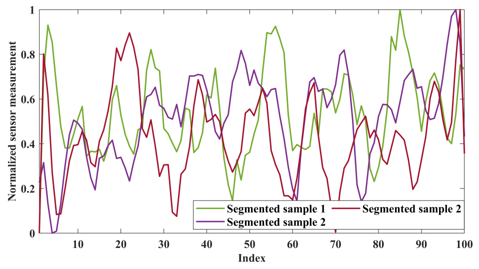
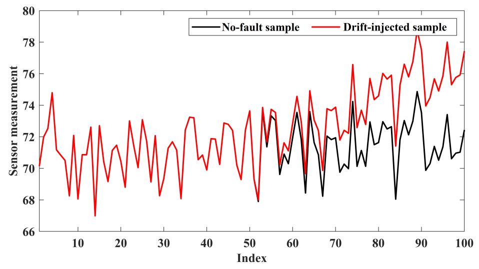


Fig. 5: Common sensor faults.



(a) Segmented samples.



(b) Synthetic drift fault.

Fig. 6: Data Preparation.

B. Gramian Angular Field Imaging for Sensor Data

The GAF image encoding technique can be segmented into three steps: rescaling the time series observation, converting the rescaled time series into polar coordinates, and finally, constructing a Gramian-like matrix using the angle values from the polar coordinates. The steps are discussed below with detailed explanations.

To begin the encoding process, a time series, $X_n = \{x_1, x_2, x_3, \ldots\ldots\ldots x_n\}$, consisting of n samples or observations is rescaled using the equation (12):

$$\widetilde{x_i} = 2\frac{x_i - \min(X_n)}{\max(X_n) - \min(X_n)} - 1 \qquad (12)$$

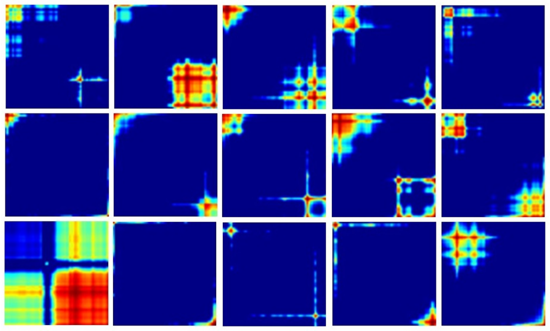


Fig. 7: Non-faulty signal instances converted to GAF images.

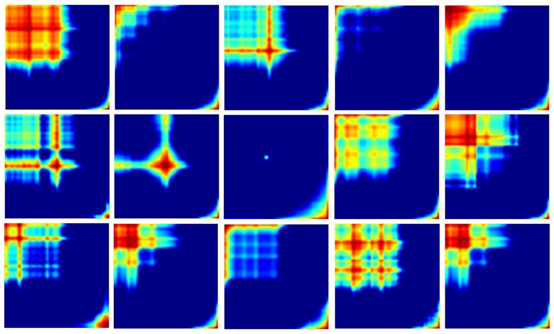


Fig. 8: Drift-fault signals converted to GAF images.

This will rescale X_n to the range -1 to 1. This rescaling is necessary because in the next step $\widetilde{x}_i$ is remapped to a polar coordinate system with corresponding values for the angle and radius. The angle and the radius for each $\widetilde{x}_i$ are defined as

$$\begin{cases} \phi_i = \arccos(\widetilde{x}_i) \\ r_i = \frac{t_i}{n} \end{cases} \tag{13}$$

The value of $\widetilde{x}_i$ is within the range -1 to 1, and thus, the value of ϕ_i lies between angles 0 and π. Here, r_i represents the corresponding radius of $\widetilde{x}_i$ in the polar coordinate system, which is computed by dividing the timestamp by the length of the time series, n. According to the authors of [53], this transformation is bijective and preserves the temporal relations. The values of angle ϕ_i are utilized to create a Gramian-like matrix as defined in (14):

$$GAF = \begin{pmatrix} \cos(\phi_1+\phi_1) & \cos(\phi_1+\phi_2) & \cdots & \cos(\phi_1+\phi_n) \\ \cos(\phi_2+\phi_1) & \cos(\phi_2+\phi_2) & \cdots & \cos(\phi_2+\phi_n) \\ \vdots & \vdots & \ddots & \vdots \\ \cos(\phi_n+\phi_1) & \cos(\phi_n+\phi_2) & \cdots & \cos(\phi_n+\phi_n) \end{pmatrix} \tag{14}$$

$$GAF = \widetilde{X}' \cdot \widetilde{X} - \sqrt{I - \widetilde{X}^2}' \cdot \sqrt{I - \widetilde{X}^2} \tag{15}$$

In (15) $\widetilde{X}'$ represents the transpose of the rescaled time series, and I corresponds to a unit row vector of length n. Some instances of GAF-encoded images of non-faulty sensor signals and drift-fault-injected signals are shown in Fig. 7 and Fig. 8, respectively.

IV. Results and discussion

In this section, performance of the digital sensor and fault detection are discussed. First, the performance of the GAN as a digital representation of the sensor is discussed, and then, the fault detection capability is explored. Commonly used performance metrics for classification problem as: accuracy, F1-score, precision, recall, and confusion matrix are calculated to interpret the fault detection performance.

A confusion matrix is a tabular representation generally used to evaluate the performance in a classification task. The matrix enlists the predicted labels by the classifier and the actual labels of the samples. In a binary classification scenario, there are four entries in the confusion matrix as:

True Positive (TP): Instances of positive class that are correctly predicted (predicted as positive class)

True Negative (TN): Instances of negative class that are correctly predicted (predicted as negative class)

False Positive (FP): Instances of the negative class that are incorrectly predicted (predicted as positive)

False Negative (FN): Instances of the positive class that are incorrectly predicted (predicted as negative)

Confusion matrix is useful because it provides a visual representation of the models performance and other common performance metrices such as precision, recall, F1 score, and accuracy can be derived from the entries of the matrix using the following expressions:

$$Accuracy = \frac{TN + TP}{TN + FP + TP + FN} \tag{16}$$

$$Precision = \frac{TP}{FP + TP} \tag{17}$$

$$Recall = \frac{TP}{FN + TP} \tag{18}$$

$$F1 - score = \frac{TP}{TP + \frac{1}{2}(FP + FN)} \tag{19}$$

A. The WGAN-GP as a Virtual Sensor

As mentioned in the proposed methodology segment, the sensor signal is converted into an image for training, and therefore, the performance of the DT inspired GAN model will be assessed by observing the similarity between the images generated by the GAN and the corresponding images seen during training. Summaries of the generator and discriminator segments of the GAN are presented in Table IV and Table V, respectively.

The model was trained for 200 epochs, and the losses for both the generator and the discriminator networks are presented in Fig. 9.

From the loss plots in Fig. 9, we can see that for both cases the loss converges to values around 20. The generator took a lot of time to converge, the loss decreased with a constant slope until 400 training steps and then stabilized. The discriminator loss converged earlier than the discriminator at around the 100^{th} step. For both plots, although there are local variances in the loss values, the global pattern did not vary by

TABLE IV: Summary of the generator segment

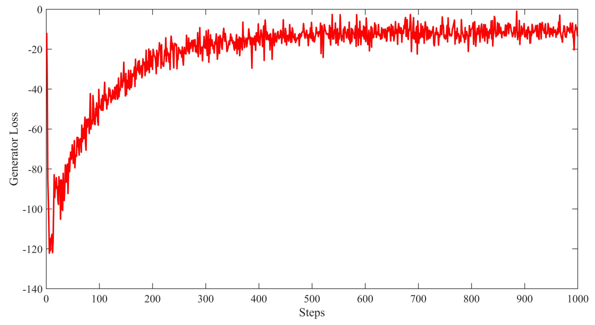
Layer(type)	Output shape	Parameters
Input layer	[(None,100)]	0
Dense layer	(None, 8192)	819200
Reshape	(None, 4, 4, 512)	0
Conv 2D Transpose layer	[(None,4,4,512)]	4194304
Batch Normalization layer	[(None,4,4,512)]	2048
Leaky ReLU	(None,4,4,512)	0
Conv 2D Transpose layer	(None, 8, 8, 256)	2097152
Batch Normalization layer	(None, 8, 8, 256)	1024
Leaky ReLU	(None, 8, 8, 256)	0
Conv 2D Transpose layer	(None, 16, 16, 128)	524288
Batch Normalization layer	(None, 16, 16, 128)	512
Leaky ReLU	(None, 16, 16, 128)	0
Conv 2D Transpose layer	(None, 32, 32, 64)	131072
Batch Normalization layer	(None, 32, 32, 64)	256
Leaky ReLU	(None, 32, 32, 64)	0
Conv 2D Transpose layer	(None, 64, 64, 3)	3072
Total params:	7,772,928	
Trainable params:	7,771,008	
Non-trainable params:	1,920	

TABLE V: Summary of the discriminator segment

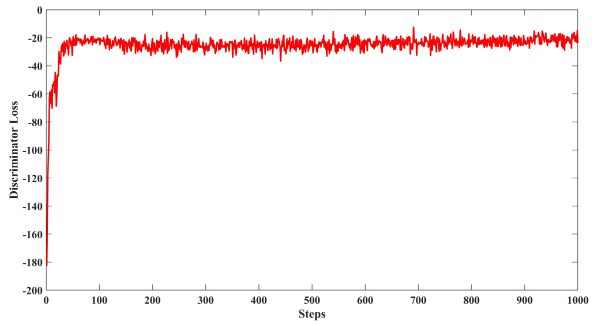
Layer(type)	Output shape	Parameters
Input Layer	[(None, 64, 64, 3)]	0
Conv 2D layer	(None, 32, 32, 64)	3072
Leaky ReLU	(None, 32, 32, 64)	0
Conv 2D layer	(None, 16, 16, 128)	131072
Leaky ReLU	(None, 16, 16, 128)	0
Conv 2D layer	(None, 8, 8, 256)	524288
Leaky ReLU	(None, 8, 8, 256)	0
Conv 2D layer	(None, 4, 4, 512)	2097152
Leaky ReLU	(None, 4, 4, 512)	0
Conv 2D layer	(None, 4, 4, 1)	8192
Flatten	(None, 16)	0
Dense	(None, 1)	17
Total params:	2,763,793	
Trainable params:	2,763,793	
Non-trainable params:	0	

much. The plots are shown for 1000 iterations to observe that if there are any overfitting issues. This is a good indication that the GAN model was trained well and can generate plausible images from random vectors that are similar to the training set. Fig. 10 depicts some sample images generated by the GAN model after completion of the training.

The images shown in Fig. 10 closely resemble the images of the training set created through GAF encoding of the sensor signals. Since the GAN model is capable of generating images similar to normal data from the sensor, we conclude that the GAN model can be considered a digital representation of the sensor in its normal state.



(a) Generator loss.



(b) Discriminator loss.

Fig. 9: Generator and discriminator losses of the GAN during training.

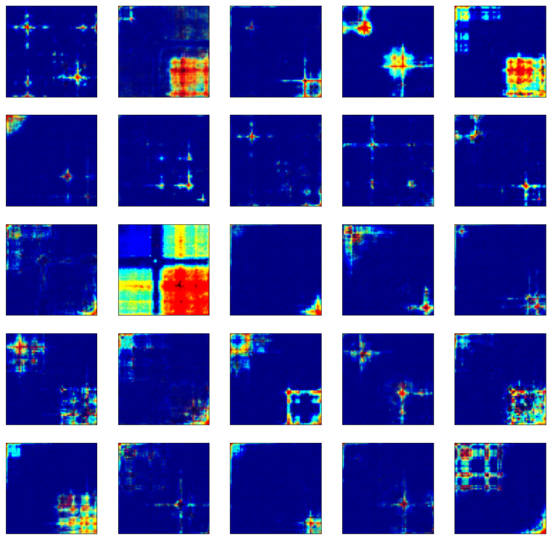


Fig. 10: Sample images generated from the GAN model.

B. Sensor fault detection

The discriminator of the GAN will perform sensor fault detection because it learned the pattern of a non-faulty signal in the training stage. Therefore, when any image is not similar to the non-faulty signal, the discriminator output will be different. To observe the response of a trained discriminator, a total of 2000 images, half of them corresponding to non-faulty signals and half corresponding to a drift fault signal, were given as input to the discriminator. The response of the discriminator is presented in Fig. 11. The different responses of the discriminator to faulty and non-faulty samples are evident. The model was trained several times, and in each case, a similar trend was observed. In Fig. 11, the best response distribution is where a minimum overlap between classes occurred. In some experiments, two classes overlap to some extent, but with a careful choice for the decision threshold it is possible to achieve better accuracy. For this best case scenario, the discriminator output was more negative for faulty signal instances; for non-faulty samples, the discriminator output fell into the range -33 to -25. However, there were about 30 faulty

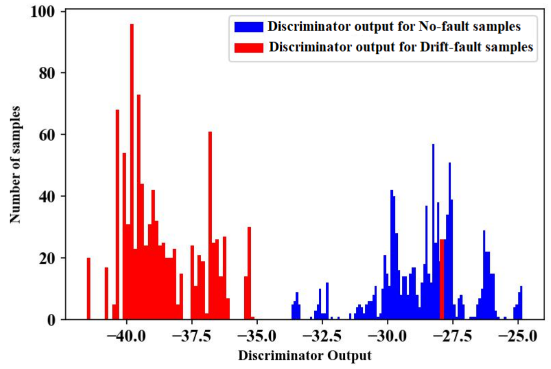


Fig. 11: Discriminator response for non-fault and drift fault signal.

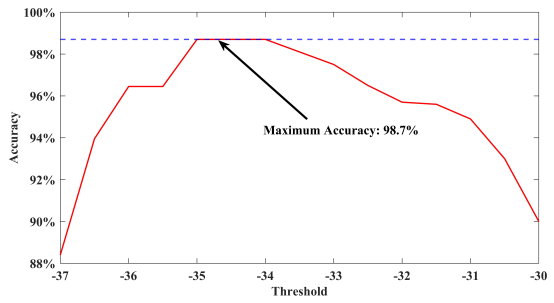
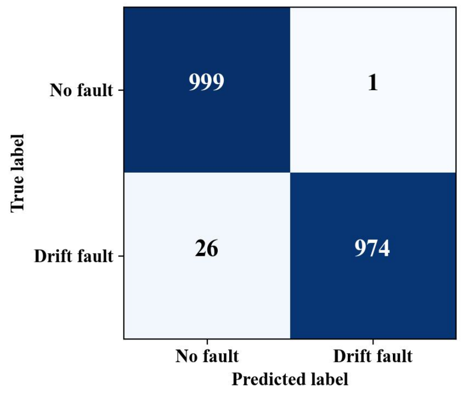
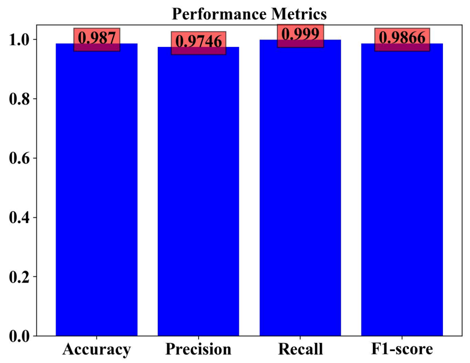


Fig. 12: Detection accuracy for various threshold values.



(a) Confusion matrix.



(b) Performance metrics.

Fig. 13: Summary performance.

instances for which the discriminator output was around -28, which is within the range of non-faulty instances. Now, a threshold for discriminator output is required for deciding fault occurrences. A graph of fault detection accuracy with respect to different discriminator threshold values is plotted in Fig. 12. The graph shows that detection accuracy decreases if the threshold is less than -35 or greater than -34. The maximum detection accuracy was attained when the decision threshold was between -34 to -35 (98.7%). Thus, if the threshold is set within the range -34 to -35 , the discriminator can detect faults with high accuracy. Common performance metrics are depicted in Fig. 13. The confusion matrix shows that, in the current scenario, the fault detection system can identify almost all no-fault instances, but out of 1000 drift fault samples, it failed to detect 26 of them. The precision, recall, and F1-score values are also very consistent.

Our proposed approach involves training a GAN model with only a single class of image for non-faulty signals. A similar idea of using a single class for abnormality detection was developed in [54], where the ResNet18 architecture was used to compute features. Later, the computed features were used to train a machine learning model. In the initial stage, this approach only used normal road surface images as input to the ResNet18 architecture. However, the final fully connected layer was omitted, which allowed use of the values obtained from the final pooling layer as features. Following a similar process, features for images that represented road cracks were also extracted. In the training stage, the extracted features from normal images were used to train a support vector machine. Because the classifier was trained on normal images, in the final stage, the classifier was provided with features for both normal and abnormal images. Because the classifier was trained on only one class, it was expected to provide different values for images it did not use during the training stage. For classification of crack and non-crack images, a threshold was selected and (based on the threshold) decisions were made by comparing the classifier value with the threshold. We used a GAF-image dataset of no-fault and drift-fault images to observe sensor fault detection performance. The ResNet18-SVM-based model was trained with 2000 no-fault images, and later, 1200 images were used for testing with the one class. The performance is summarized in the confusion matrix presented in Fig. 14.

TABLE VI: Comparison between the two approaches

Model	Accuracy	Precision	Recall	F1-Score
ResNet18-SVM [54]	0.945	0.921	0.973	0.947
GAN-based model (Proposed)	0.987	0.975	0.999	0.987

If we observe the confusion matrixes in Fig. 13 and Fig. 14, we can see that the number of misclassified samples was a little higher in the ResNet18-SVM-based model. Table VI

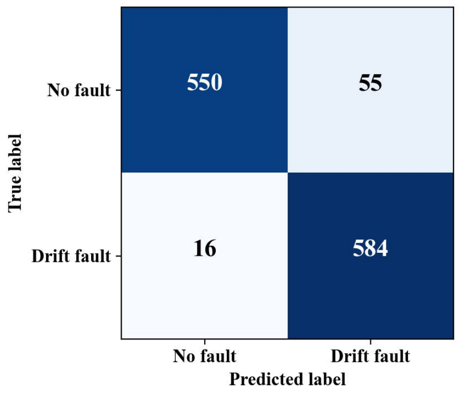


Fig. 14: Confusion matrix for the ResNet18-SVM classifier.

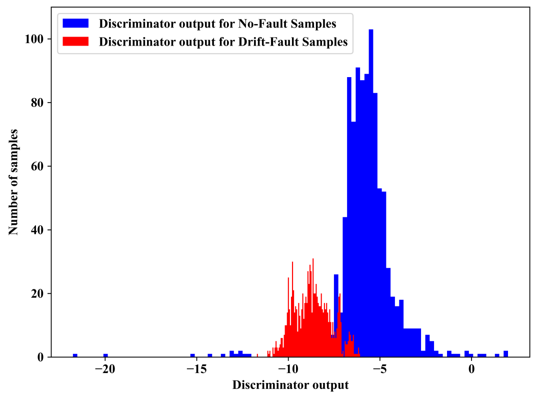


Fig. 15: Discriminator response for ETDataset temperature sensor signal.

provides a comparison of the common classification metrics for the two models, and the numbers indicate that the accuracy of the GAN-based fault classifier was 4.2% higher than the ResNet18-SVM approach.

To support the robustness of the proposed approach, we checked the fault detection performance by the model for two other datasets. The datasets considered are the Electricity Transformer Dataset(ETDataset) [55] and the TON_IoT dataset from University of New South Wales (UNSW) used in [56]. The ETDataset consists the oil temperature of a transformer which is significant to indicate the transformer condition and this signal is considered here. On the other hand the TON_IoT dataset contains a collection of sensor signal and we considered a temperature sensor signal collected by a Modbus sensor deployed for environment monitoring. We injected fault following the similar approach as discussed in data preparation section and encoded the timeseries data into images. The same GAN architecture was trained and then implemented for fault detectin on a set of 2000 images. The discriminator responses for the two datasets are provided in Fig. 15 and Fig. 16. From the discriminator responses it is evident that the discriminator response are largely different for the no-fault and fault class samples. For the discriminator response corresponding to the ETDataset maximum accuracy of 92.7% is attained for a threshold of -7.1 and in case of the discriminator response for TON_IoT dataset the maximum accuracy of 96.7%is obtained for a threshold value of 15.5. The corresponding confusion matrices are presented in Fig. 17 and Fig. 18. Other performance parameters are listed in Table VII. The parameters in Table VII indicates that the proposed model delivers consistent performance across different datasets.

TABLE VII: Performance of the proposed method on different datasets

Dataset	Accuracy	Precision	Recall	F1-Score
WSN Dataset	0.987	0.975	0.999	0.987
ETDataset	0.927	0.937	0.915	0.926
TON_IoT Dataset	0.967	0.936	0.915	0.926

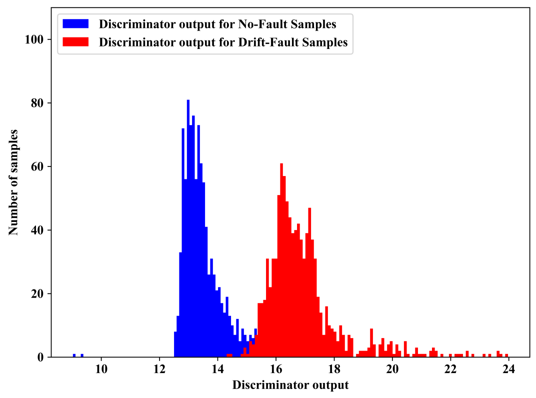


Fig. 16: Discriminator response for TON_IoT temperature sensor signal .

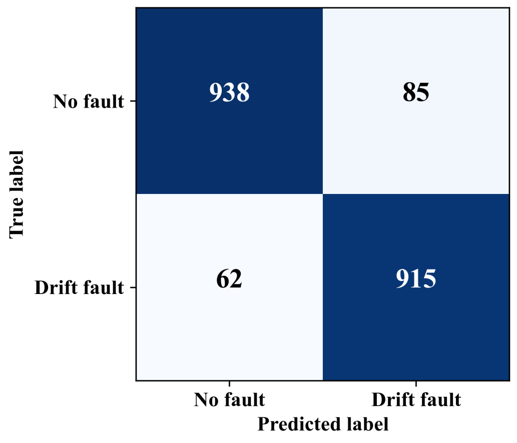


Fig. 17: Confusion matrix for ETDataset temperature sensor signal.

Our proposed approach differs from most of the existing works on sensor fault detection in the aspect that, it uses only single class i.e., the normal state data from the sensor while the major existing works considered both the fault and

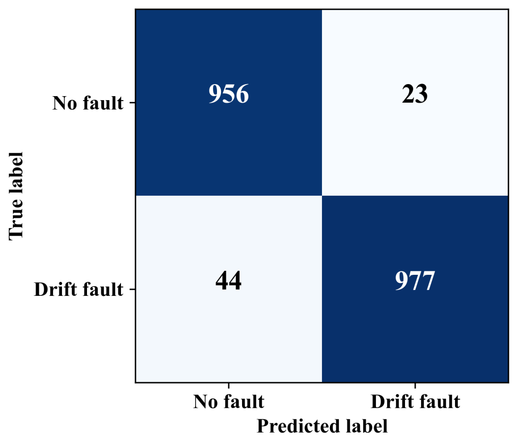


Fig. 18: Confusion matrix for TON_IoT temperature sensor signal.

normal state data. However, for the shake of comparison we consider two previous works on same dataset. In [32] authors investigated extremely randomized tree for several fault classification. In their work they used F1-score to compare individual fault detection performance. They achieved an F1-score of 98 for drift fault while using the same dataset. In our work we also obtained an F1-score of 99. The same work also implements MLP, DT, SVM, and RF while reporting F1-score values of 61, 99.2, 99.4 and 99 respectively. Our proposed approach shows higher value of F1-score than the MLP and is comparable with the other models. Similar performance is also reported in [49] for the same parameter where the F1-score is found as 98 for drift fault which is again marginally smaller than our proposed model. However, it must be considered that we achieve the similar performance while using only a single class samples i.e., the normal state data from the sensor.

V. Conclusion

In this work, a different data-driven fault detection approach inspired from the concept of DT for a humidity sensor was presented and its performance discussed. Fault detection was performed using a virtual representation of the physical sensor that was created by the newest type of deep neural network, famously known as the Generative Adversarial Network. Specifically, a Deep Convolutional Generative Adversarial Network architecture was used to create the virtual sensor. Considering the fact that major research work used the GAN architecture for image data extensively, in our work, we also converted the times series sensor data to images using Gramian Angular Field encoding. Prior to image encoding, a fault dataset was created by injecting synthetic faults into normal sensor signals. The GAN model was trained with images created from non-faulty sensor signals. After successful training, we observed that the model is capable of generating samples from a random noise vector, and thus, a GAN-based functional digital representation of the sensor was created. In the fault detection stage, the discriminator network of the GAN model was utilized by selecting the decision threshold for the discriminator. In a few repeated experiments, we observed a minimum fault detection accuracy of 92.55% and a maximum of 98.7%. Comparison with similar approaches that involve a ResNet architecture with an SVM showed that the GAN-based approach could provide better classification of our GAF-image dataset of sensor faults. The performance of the proposed method is found to be consistent across three different datasets which is an indication of robustness of the model. The overall performance assessment indicates that this digital-twin inspired GAN-based approach can be considered a new method for sensor fault detection even only the normal condition data is used during the training process. Although only one type of sensor fault was considered in our work, the efficacy of this approach is promising. We plan to investigate the performance of this method with more fault types and greater complexities.

References

[1] C. Sobin, "A survey on architecture, protocols and challenges in iot," *Wireless Personal Communications*, vol. 112, no. 3, pp. 1383–1429, 2020.
[2] M. A. Ahmed, J. L. Gallardo, M. D. Zuniga, M. A. Pedraza, G. Carvajal, N. Jara, and R. Carvajal, "Lora based iot platform for remote monitoring of large-scale agriculture farms in chile," *Sensors*, vol. 22, no. 8, p. 2824, 2022.
[3] H. Bates, M. Pierce, and A. Benter, "Real-time environmental monitoring for aquaculture using a lorawan-based iot sensor network," *Sensors*, vol. 21, no. 23, p. 7963, 2021.
[4] J. Mendes, E. Peres, F. Neves dos Santos, N. Silva, R. Silva, J. J. Sousa, I. Cortez, and R. Morais, "Vineinspector: The vineyard assistant," *Agriculture*, vol. 12, no. 5, p. 730, 2022.
[5] M. Forcén-Muñoz, N. Pavón-Pulido, J. A. López-Riquelme, A. Temnani-Rajjaf, P. Berríos, R. Morais, and A. Pérez-Pastor, "Irriman platform: Enhancing farming sustainability through cloud computing techniques for irrigation management," *Sensors*, vol. 22, no. 1, p. 228, 2021.
[6] F. Khan, M. A. B. Siddiqui, A. U. Rehman, J. Khan, M. T. S. A. Asad, and A. Asad, "Iot based power monitoring system for smart grid applications," in *2020 International Conference on Engineering and Emerging Technologies (ICEET)*, pp. 1–5, 2020.
[7] A. Shashank, R. Vincent, A. K. Sivaraman, A. Balasundaram, M. Rajesh, and S. Ashokkumar, "Power analysis of household appliances using iot," in *2021 International Conference on System, Computation, Automation and Networking (ICSCAN)*, pp. 1–5, 2021.
[8] K. Kumaran *et al.*, "Power theft detection and alert system using iot," *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, vol. 12, no. 10, pp. 1135–1139, 2021.
[9] T. Alam, "Cloud-based iot applications and their roles in smart cities," *Smart Cities*, vol. 4, no. 3, pp. 1196–1219, 2021.
[10] T. Wu, F. Wu, C. Qiu, J.-M. Redouté, and M. R. Yuce, "A rigid-flex wearable health monitoring sensor patch for iot-connected healthcare applications," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 6932–6945, 2020.
[11] Y. Dong and Y.-D. Yao, "Iot platform for covid-19 prevention and control: A survey," *IEEE Access*, vol. 9, pp. 49929–49941, 2021.
[12] A. I. Sunny, A. Zhao, L. Li, and S. K. Sakiliba, "Low-cost iot-based sensor system: A case study on harsh environmental monitoring," *Sensors*, vol. 21, no. 1, p. 214, 2020.
[13] S. Aryal, A. A. Baniya, and K. Santosh, "Improved histogram-based anomaly detector with the extended principal component features," *arXiv preprint arXiv:1909.12702*, 2019.
[14] N. Bayar, S. Darmoul, S. Hajri-Gabouj, and H. Pierreval, "Fault detection, diagnosis and recovery using artificial immune systems: A review," *Engineering Applications of Artificial Intelligence*, vol. 46, pp. 43–57, 2015.
[15] R.-X. Guo, K. Guo, and J.-K. Dong, "Fault diagnosis for sensors in a class of nonlinear systems," *IMA Journal of Mathematical Control and Information*, vol. 35, no. 2, pp. 375–391, 2018.
[16] D. Li, Y. Wang, J. Wang, C. Wang, and Y. Duan, "Recent advances in sensor fault diagnosis: A review," *Sensors and Actuators A: Physical*, vol. 309, p. 111990, 2020.

[17] H. Wu and J. Zhao, "Deep convolutional neural network model based chemical process fault diagnosis," *Computers & chemical engineering*, vol. 115, pp. 185–197, 2018.

[18] R. Xiong, Q. Yu, W. Shen, C. Lin, and F. Sun, "A sensor fault diagnosis method for a lithium-ion battery pack in electric vehicles," *IEEE Transactions on Power Electronics*, vol. 34, no. 10, pp. 9709–9718, 2019.

[19] M. Taimoor and L. Aijun, "Lyapunov theory based adaptive neural observers design for aircraft sensors fault detection and isolation," *Journal of Intelligent & Robotic Systems*, vol. 98, no. 2, pp. 311–323, 2020.

[20] S. TayebiHaghighi and I. Koo, "Sensor fault diagnosis using a machine fuzzy lyapunov-based computed ratio algorithm," *Sensors*, vol. 22, no. 8, p. 2974, 2022.

[21] H. H. Alhelou, "Fault detection and isolation in power systems using unknown input observer," *Advanced condition monitoring and fault diagnosis of electric machines*, pp. 38–58, 2019.

[22] F. Garramiola, J. Del Olmo, J. Poza, P. Madina, and G. Almandoz, "Integral sensor fault detection and isolation for railway traction drive," *Sensors*, vol. 18, no. 5, p. 1543, 2018.

[23] C. Liang, S. Wang, R. Chen, S. Zhao, and Y. Wu, "Research on ship electronic power system fault diagnosis based on expert system," in *IOP Conference Series: Materials Science and Engineering*, vol. 738, p. 012017, IOP Publishing, 2020.

[24] Z. Dong, J. Zhao, J. Duan, M. Wang, and H. Wang, "Research on agricultural machinery fault diagnosis system based on expert system," in *2018 2nd IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, pp. 2057–2060, IEEE, 2018.

[25] M. Geetha and J. Jerome, "Fuzzy expert system based sensor and actuator fault diagnosis for continuous stirred tank reactor," in *2013 International Conference on Fuzzy Theory and Its Applications (iFUZZY)*, pp. 251–257, IEEE, 2013.

[26] S. U. Jan, Y.-D. Lee, J. Shin, and I. Koo, "Sensor fault classification based on support vector machine and statistical time-domain features," *IEEE Access*, vol. 5, pp. 8682–8690, 2017.

[27] D. Peng, S. Yun, D. Yin, B. Shen, C. Xu, and H. Zhang, "A sensor fault diagnosis method for gas turbine control system based on emd and svm," in *2021 Power System and Green Energy Conference (PSGEC)*, pp. 682–686, IEEE, 2021.

[28] X. Cheng, D. Wang, C. Xu, and J. Li, "Sensor fault diagnosis method based on-grey wolf optimization-support vector machine," *Computational Intelligence and Neuroscience*, vol. 2021, 2021.

[29] A. Anandhalekshmi, V. S. Rao, and G. Kanagachidambaresan, "Hybrid approach of baum-welch algorithm and svm for sensor fault diagnosis in healthcare monitoring system," *Journal of Intelligent & Fuzzy Systems*, no. Preprint, pp. 1–10, 2022.

[30] A. Naimi, J. Deng, S. Shimjith, and A. J. Arul, "Fault detection and isolation of a pressurized water reactor based on neural network and k-nearest neighbor," *IEEE Access*, vol. 10, pp. 17113–17121, 2022.

[31] A. M. Abed, S. A. Gitaffa, and A. H. Issa, "Quadratic support vector machine and k-nearest neighbor based robust sensor fault detection and isolation," *Engineering and Technology Journal*, vol. 39, no. 5, pp. 859–869, 2021.

[32] U. Saeed, S. U. Jan, Y.-D. Lee, and I. Koo, "Fault diagnosis based on extremely randomized trees in wireless sensor networks," *Reliability engineering & system safety*, vol. 205, p. 107284, 2021.

[33] L. Gou, H. Li, H. Zheng, H. Li, and X. Pei, "Aeroengine control system sensor fault diagnosis based on cwt and cnn," *Mathematical Problems in Engineering*, vol. 2020, 2020.

[34] J. Huang, M. Li, Y. Zhang, L. Mu, Z. Ao, and H. Gong, "Fault detection and classification for sensor faults of uav by deep learning and time-frequency analysis," in *2021 40th Chinese Control Conference (CCC)*, pp. 4420–4424, IEEE, 2021.

[35] T. Zhao, H. Zhang, X. Zhang, Y. Sun, L. Dou, and S. Wu, "Multi-fault identification of iron oxide gas sensor based on cnn-wavelelet-based network," in *2021 19th International Conference on Optical Communications and Networks (ICOCN)*, pp. 1–4, IEEE, 2021.

[36] D. Jana, J. Patil, S. Herkal, S. Nagarajaiah, and L. Duenas-Osorio, "Cnn and convolutional autoencoder (cae) based real-time sensor fault detection, localization, and correction," *Mechanical Systems and Signal Processing*, vol. 169, p. 108723, 2022.

[37] S. U. Jan, Y. D. Lee, and I. S. Koo, "A distributed sensor-fault detection and diagnosis framework using machine learning," *Information Sciences*, vol. 547, pp. 777–796, 2021.

[38] H. Darvishi, D. Ciuonzo, and P. S. Rossi, "Exploring a modular architecture for sensor validation in digital twins," in *2022 IEEE Sensors*, pp. 1–4, IEEE, 2022.

[39] H. Darvishi, D. Ciuonzo, and P. S. Rossi, "A machine-learning architecture for sensor fault detection, isolation and accommodation in digital twins," *IEEE Sensors Journal*, 2022.

[40] T. N. Nguyen, R. Ponciroli, P. Bruck, T. C. Esselman, J. A. Rigatti, and R. B. Vilim, "A digital twin approach to system-level fault detection and diagnosis for improved equipment health monitoring," *Annals of Nuclear Energy*, vol. 170, p. 109002, 2022.

[41] P. Jain, J. Poon, J. P. Singh, C. Spanos, S. R. Sanders, and S. K. Panda, "A digital twin approach for fault diagnosis in distributed photovoltaic systems," *IEEE Transactions on Power Electronics*, vol. 35, no. 1, pp. 940–956, 2019.

[42] F. Piltan and J.-M. Kim, "Bearing anomaly recognition using an intelligent digital twin integrated with machine learning," *Applied Sciences*, vol. 11, no. 10, p. 4602, 2021.

[43] Q. Xu, S. Ali, and T. Yue, "Digital twin-based anomaly detection in cyber-physical systems," in *2021 14th IEEE Conference on Software Testing, Verification and Validation (ICST)*, pp. 205–216, IEEE, 2021.

[44] E. Zotov, A. Tiwari, and V. Kadirkamanathan, "Conditional style-gan modelling and analysis for a machining digital twin," *Integrated Computer-Aided Engineering*, no. Preprint, pp. 1–17, 2021.

[45] W. Booyse, D. N. Wilke, and S. Heyns, "Deep digital twins for detection, diagnostics and prognostics," *Mechanical Systems and Signal Processing*, vol. 140, p. 106612, 2020.

[46] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[47] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," *Advances in neural information processing systems*, vol. 30, 2017.

[48] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International conference on machine learning*, pp. 214–223, PMLR, 2017.

[49] U. Saeed, Y.-D. Lee, S. U. Jan, and I. Koo, "Cafd: context-aware fault diagnostic scheme towards sensor faults utilizing machine learning," *Sensors*, vol. 21, no. 2, p. 617, 2021.

[50] S. Safavi, M. A. Safavi, H. Hamid, and S. Fallah, "Multi-sensor fault detection, identification, isolation and health forecasting for autonomous vehicles," *Sensors*, vol. 21, no. 7, p. 2547, 2021.

[51] H. Darvishi, D. Ciuonzo, E. R. Eide, and P. S. Rossi, "A data-driven architecture for sensor validation based on neural networks," in *2020 IEEE SENSORS*, pp. 1–4, IEEE, 2020.

[52] S. Suthaharan, M. Alzahrani, S. Rajasegarar, C. Leckie, and M. Palaniswami, "Labelled data collection for anomaly detection in wireless sensor networks," in *2010 sixth international conference on intelligent sensors, sensor networks and information processing*, pp. 269–274, IEEE, 2010.

[53] Z. Wang and T. Oates, "Encoding time series as images for visual inspection and classification using tiled convolutional neural networks," in *Workshops at the twenty-ninth AAAI conference on artificial intelligence*, 2015.

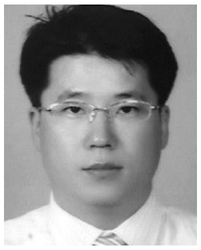
[54] K. Ithakura, "Crack detection using one-class svm." https://github.com/KentaItakura/Crack-detection-using-one-class-SVM, 2021.

[55] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *The Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Virtual Conference*, vol. 35, pp. 11106–11115, AAAI Press, 2021.

[56] N. Moustafa, "A new distributed architecture for evaluating ai-based security systems at the edge: network ton_iot datasets. sustain. cities soc. 72, 102994 (2021)," 2021.

Md. Nazmul Hasan received his BS and MS degree in electronics and communication engineering from Khulna University, Khulna, Bangladesh in 2012 and 2016. He worked as a Data Communication Engineer in Huawei Technologies Bangladesh Ltd. in 2013. He is currently pursuing his PhD in electrical and computer engineering with University of Ulsan, Ulsan, South Korea. His research interests include machine learning, deep learning, and sensor fault analysis.

Sana Ullah Jan is an experienced researcher with more than 8 years of cutting-edge research and teaching experience in prestigious institutes including the University of the West of Scotland, the University of Ulsan (South Korea) and the University of Lahore (Pakistan). He is currently enrolled as Lecturer/Assistant Professor in Edinburgh Napier University, UK since September 2021. He was previously enrolled as Postdoctoral Research Fellow at the Center of Affective and Human Computing for Smart Environment at the school of computing, engineering and physical sciences, University of the West of Scotland since September 2020 to August 2021. He has (co)authored more than 20 papers in international journals and peer-reviewed international conference proceedings. His research area is closely related to the Artificial Intelligence or Machine Learning-based cyber security and privacy in the Internet-of-Things, Cyber Physical Systems and eHealth, and Sensor Fault Detection and Diagnosis.



Insoo Koo received the B.E. degree from Kon-Kuk University, Seoul, South Korea, in 1996, and the M.Sc. and Ph.D. degrees from the Gwangju Institute of Science and Technology (GIST), Gwangju, Korea, in 1998 and 2002, respectively.,From 2002 to 2004, he was a Research Professor at the Ultrafast Fiber-Optic Networks Research Center, GIST. In 2003, he was a Visiting Scholar at the Royal Institute of Science and Technology, Stockholm, Sweden. In 2005, he joined the University of Ulsan, Ulsan, South Korea, where he is currently a Full Professor. His current research interests include spectrum sensing issues for CRNs, channel and power allocation for cognitive radios (CRs) and military networks, SWIPT MIMO issues for CRs, MAC and routing protocol design for UW-ASNs, and relay selection issues in CCRNs.