Catch You if Pay Attention: Temporal Sensor Attack Diagnosis Using Attention Mechanisms For Cyber-Physical Systems

Zifan Wang
Syracuse University
zwang345@syr.edu

Lin Zhang
University of Pennsylvania
cpsec@seas.upenn.edu

Qinru Qiu
Syracuse University
qiqiu@syr.edu

Fanxin Kong
University of Notre Dame
fkong@nd.edu

Abstract—**In Cyber-Physical Systems (CPS), sensor data integrity is crucial since acting on malicious sensor data can cause serious consequences, given the tight coupling between cyber components and physical systems. While extensive works focus on sensor attack detection, attack diagnosis that aims to find out when the attack starts has not been well studied yet. This temporal sensor attack diagnosis problem is equally important because many recovery methods rely on the accurate determination of trustworthy historical data. To address this problem, we propose a lightweight data-driven solution to achieve real-time sensor attack diagnosis. Our novel solution consists of five modules, with the attention and diagnosis ones as the core. The attention module not only helps accurately predict future sensor measurements but also computes statistical attention scores for the diagnosis module. Based on our unique observation that the score fluctuates sharply once an attack launches, the diagnosis module determines the onset of an attack through monitoring the fluctuation. Evaluated on high-dimensional high-fidelity simulators and a testbed, our solution demonstrates robust and accurate temporal diagnosis results while incurring millisecond-level computational overhead on Raspberry Pi.**

Index Terms—**cyber-physical systems, sensor attacks, real-time, detection, diagnosis**

I. INTRODUCTION

Cyber-Physical Systems (CPS) couple of computing and communication parts with sensing and actuation to interact with the physical world. CPS has been evolving from isolated control systems to being complex, heterogeneous, and connected to offer advanced functionalities. This evolution enables new applications such as autonomous vehicles, unmanned aerial vehicles and smart manufacturing that promise enormous benefits. Meanwhile, potential security vulnerabilities arise in modern CPS due to their open architectures [1]–[5].

Compared to conventional IT systems, challenges in CPS security are distinct in terms of not only consequences in case of security breaches but also attack surfaces [6]–[9]. Attacks on CPS might cause damage to the physical property and even endanger human lives, as results of plant explosion [10], [11], power cutoff [12] and car accidents [13]. Sensors act as an interface between the cyber and physical space, and thus their data integrity is critical to CPS security. Sensor data can be spoofed both in the cyber space such as software and network attacks [14] and in the physical space such as transduction attacks [7], [15]. Sensor spoofing by transduction attacks or physical tampering sensors may bypass conventional intrusion detection systems which mainly monitor computing and networking devices [15]–[17].

These security threats have motivated many research efforts on data integrity of sensor measurements [7], [8], [18]–[20]. One major focus is on determining whether sensor measurements are compromised, called sensor attack detection. Existing works can be categorized into two groups. The first group relies on domain knowledge of CPS, usually the mathematical model of the physical system such as a linear and non-linear system model [7], [15], [21]–[23]. The detection uses statistical methods such as χ^2 and kernel density estimation, to identify anomalies by tracking the difference between the observed and predicted sensor values. However, as CPS becomes more complex and attacks are more sophisticated (e.g., stealthy attacks [24]), these methods tend to require more domain knowledge and become insufficient to ensure the overall sensor data integrity in CPS [19].

To handle the high dimensionality in both spatial (i.e., large number of sensors) and temporal (i.e., long time series) aspects, the second group of works employ Deep-Learning (DL) based methods to detect sensor attacks in CPS [19]. Current studies have explored various neural network architectures such as Autoencoder (AE) [3], [25], Variational Autoencoder (VAE) [26], Convolutional Neural Network (CNN) [27], [28], Long Short-Term Memory (LSTM) [29]–[31], and Generative Adversarial Network (GAN) [32], [33], to detect sensor attacks in different CPS applications such as autonomous vehicles [3], [34], aerial systems [35], [36], smart grids [37], and industrial control systems [26], [38].

In spite of these extensive efforts on attack detection, sensor attack diagnosis has been inadequately addressed so far. Note that attack diagnosis differs from attack detection: the former aims to not only identify "who": which sensors are under attack, but also find out "when": when the attack starts. Diagnosis is equally important because many attack recovery works rely on historical sensor data to estimate the current system state [39]–[46]. Using corrupted historical data will lead to a failed recovery which drives a system to unsafe states. The diagnosis needs to tell not only which sensors are trustworthy but also until when their data are trustworthy. In short, after the attack is detected, attack diagnosis is activated to identify the trustworthy data that attack recovery can use.

Although some existing detection works may be extendable to identify the "who", but none can do the "when" [7], [8]. For example, we may set a threshold for the difference between the predicted value and observed measurement of each individual sensor, and then recognize a sensor as corrupted if the difference is greater than its threshold [15], [23], [47], [48]. However, attack detection is not attack diagnosis, and it is unable to determine when the attack starts. Because using the detection time, i.e. the moment when an attack is detected, as the attack start time, is far from accurate. The reason is that a detector may take some time, i.e. the detection delay, to find out an attack after it starts.

This paper is right focused on diagnosing the "when", i.e., the temporal sensor attack diagnosis problem. However, solving this problem is challenging because attacks can deviate a system in an accumulative and unpredictable way. For example, stealthy attacks usually attempt to hide themselves from attack detectors by making small and gradual modifications to sensor measurements such as at a scale of sensing noise [24], [49], [50]. Thus, it might have taken a considerable amount of time when such an attack is detected. Tracing back the difference between the predicted value and observed measurement will not work for deciding when the attack starts, because at that time, the difference is small and has not become greater than the threshold yet. Therefore, we need a diagnosis method that is sensitive (and robust, as always) to attacks.

To address the challenge, we propose a new real-time diagnosis system for CPS sensor attacks. The system consists of multiple data-driven modules that are trained offline and run online, providing temporal sensor attack diagnosis. Our diagnosis solution explores novel uses of recently-proposed attention mechanisms [51], [52], and the foundation is our observation that the attention mechanism is sensitive to input changes. Our proposed system's main advantages are fourfold: 1) it diagnoses sensor attacks with zero knowledge of the corresponding CPS and is not confined to certain types of attacks; 2) it uses historical data and does not depend on sensor redundancy; 3) it can easily take in most existing data-driven models as an enhancement to diagnose sensor attacks; 4) it is light-weight and only requires millisecond-level computing time in edge devices such as Raspberry Pi. To be specific, the contribution of this work is summarized as follows:

• We bridge the significant research gap in temporal sensor attack diagnosis for CPS by proposing a real-time and robust diagnosis solution based on attention mechanisms. By leveraging the sensitivity of the attention mechanism to input changes, it can accurately diagnose the starting time of sensor attacks, reducing the potential damage caused by malicious attackers.

• We implement our proposed diagnosis system and conduct extensive validation using high-fidelity quadrotor simulators from Ardupilot [53] and multiple numerical simulators on Raspberry Pi. The experimental results demonstrate that our solution can robustly and accurately diagnose the attack start time while requiring minimal computational overhead. This validates the effectiveness and practicality of our proposed solution for real-world CPS applications.

The rest of this paper is organized as follows. Section II discusses related work. Section III presents preliminaries. Section IV describes the system overview. Section V details the design for each system module. Section VI evaluates the proposed solution. Section VII concludes the paper.

II. RELATED WORKS AND THEIR LIMITATIONS

Few works exist on addressing the temporal sensor attack diagnosis problem in CPS, to the best of our knowledge. Thus, we focus on discussing sensor attack detection works and their limitations on temporal diagnosis. Note that the diagnosis is activated after the detection of an attack, and thus our method can work with most existing detection works.

A. DL Models in Detection

Deep learning prediction models typically employ structures such as fully-connected networks (FCN), recurrent neural networks (RNN), AE, and CNN. FCN is a fast yet general model that learns entangled sensor relations, classifies inputs to be attacked or not [54], [55], reconstructs states to distinguish attacked inputs, or predicts successive measurements and keeps track of residuals from actual values. AE (regarded as a special type of FCN), RNN, and CNN are more competitive in reconstruction and prediction. AEs are used to learn the correlations among sensors and raise alarms if new measurements deviate from reconstructions [3], [25], [56], and VAEs as an advanced AE are used for more generative reconstructions [57]. LSTM, capturing long and short term relationships, is a representative of RNNs that can generate sensor predictions using a long sequence of historical data. Thus attacks are detected if the residual between predictions and measurements exceeds the threshold [29], [30], [58]. CNN, naturally good at image recognition, is also widely used for attack detection [28], [59]. Other structures, such as GAN [32], can also be applied for attack detection, but usually as supporting roles.

The attention mechanism [52] gains a reputation for general improvements on multiple tasks. Therefore, some works combine it with FCN, CNN [27], and RNN [30] for better attack detection. However, these attention-based models do not fully exploit the advantages of the attention mechanism to diagnose attacks and only treated it as an enhancement in learning. Our model is also attention-based, but we extend the vanilla self-attention mechanism and use the extension to fulfill the gap of attack diagnosis, along with other novel enhancements.

B. Limitations of Detection

Extensive studies exist in attack detection, setting goals to be high accuracy, few false alarms, shorter inference time, etc. However, most works do not derive from system perspectives nor provide attack diagnosis. Typical works [5], [26], [28], [36] usually choose publicly available datasets, slice their data series into samples, evaluate models on the test set that mixed with less positive (attacked) and more negative (normal) samples, and analyze the results in terms of accuracy, F1 score, etc. Some work [28] augment the attacked samples by flipping, rotating, or manually modifying data to prevent the well-known problem of accuracy cheating, which occurs when the model achieves high accuracy on an unbalanced dataset.

Following their workflows leads to three main drawbacks: 1) publicly available datasets contain too few attacked samples for training and testing; 2) the augmented attacked samples violate the system running pattern and do not contribute to model performances in the real world because those samples will never occur; and 3) heavily focusing on optimizing the conventional metrics constrains their models from practical use. The third drawback can be attributed to the fact that their workflows do not prioritize early detection and attack diagnosis. Usually, if a model raises an alarm for an attack sample, regardless of the relative timing of the attack within the sample, the test is considered a true positive. This approach does not provide sufficient information on when an attack occurs and how severe it is, making it difficult to take appropriate countermeasures. Put differently, detecting an attack sample where the majority of the sample is already under attack signifies a late detection, which can result in severe consequences for the system. Because untimely defense is just as damaging as no defense [8], [23], [60], [61].

We acknowledge that attack forensics, which aims to identify the cause and extent of attacks through logs and other records, may seem deceptively similar to sensor attack diagnosis in CPS. However, most attack forensics system runs offline and are incapable of identifying attacks in real-time while the proposed system runs online, providing accurate temporal attack diagnosis. That is, our real-time diagnosis system can be nicely integrated with the online recovery works (e.g. [40], [41], [44], [46]) to achieve a holistic defense system. In addition, existing approaches that utilize redundancies are compatible with and benefit our proposed method.

In summary, the lack of solutions for accurate temporal diagnosis of sensor attacks in CPS motivates this work. Our approach is able to address this significant research gap and provide a practical solution for real-world CPS applications.

III. PRELIMINARIES

This section presents the scope of this paper, the system model, and the threat model.

A. Scope of Work

As mentioned above, extensive studies exist in the literature on detecting sensor attacks. However, those methods cannot be used for diagnosis because they only alert when the system is under attack, but do not suggest how long an attack has affected the system. Note that the detection delay is not fixed, rather varies on attack magnitudes. The expected detection delay for large-magnitude attacks is not applicable for diagnosis because it becomes long under attacks such as stealthy attacks. Thus need arises for attack diagnosis.

This work thus focuses on the temporal diagnosis. The proposed method decides the attack starting point after detection. It is not a replacement for attack detection; instead, it works with most existing data-driven attack detection models. Further, combing detection and diagnosis to identify trustworthy data makes recovery from attacks possible. Addressing how to improve detection performance or how to recover from attacks is out of the scope of this work. Interested readers may refer to [3], [23], [41], [44], [46], [48], [62].

B. System Model

We consider a CPS where a physical system is supervised by a controller to follow reference states. The controller executes periodically. At each control step (or time step), the controller first reads sensor measurements (e.g. velocity, pressure, etc.), and then uses a control policy to compute control demands/signals (e.g, throttle, etc.) that are applied to the actuators to drive the physical system.

System states are modeled as a N-dimensional multivariate time series $\boldsymbol{X}$. We use subscript i and superscript t to denote i^{th} dimension and time step t, and use $\boldsymbol{x}$ and $\boldsymbol{X}$ with double superscripts t, w to denote i^{th} univariate time series and the entire multivariate time series with a window size of w:

$$\begin{aligned}
&\boldsymbol{x}_i = \{x_i^1, .., x_i^t\}^T \in \mathbb{R}^{N\times 1}, \boldsymbol{x}^t = \{x_1^t, ..., x_N^t\} \in \mathbb{R}^{1\times N},\\
&\boldsymbol{x}_i^{t,w} = \{x_i^{t-w}, ..., x_i^{t-1}\}^T \in \mathbb{R}^{w\times 1},\\
&\boldsymbol{X}^{t,w} = \{\boldsymbol{x}^{t-w}, ..., \boldsymbol{x}^{t-1}\}^T \in \mathbb{R}^{w\times N}.
\end{aligned}$$

Specially, we consider control demands and sensor readings separately in this work, and $\boldsymbol{C}$ and $\boldsymbol{S}$ are used for representations: $\boldsymbol{x}^t = \{\boldsymbol{c}^t, \boldsymbol{s}^t\} = \{c_1^t, ..., c_{N_c}^t, s_1^t, ..., s_{N_s}^t\}$, where $N = N_c + N_s$, and N_c and N_s are the numbers of control channels and sensors. Similarly, $\boldsymbol{c}^{t,w}$ and $\boldsymbol{s}^{t,w}$ are the i^{th} control channel and i^{th} sensor univariate time series with a window size of w, while $\boldsymbol{C}^{t,w}$ and $\boldsymbol{S}^{t,w}$ denote the entire control channels and sensors multivariate time series.

C. Threat Model

We consider a malicious attacker who can launch sensor attacks that alter measurements received by the controller. Thus, the controller may produce misleading commands to deviate the physical system from the reference state. Typical attacks include transduction attacks, attacking state estimator or code, and attacking sensor-controller communication (e.g. I/O driver in-between or injecting malicious packets). Other code or components such as the control code or controller, controller-actuator communication, and actuators are intact. The attacker can compromise the integrity of sensor data: $\hat{\boldsymbol{S}}^{\tau,\hat{w}} = \boldsymbol{S}^{\tau,\hat{w}} \pm \boldsymbol{V}^{\tau,\hat{w}}$, where τ and $\hat{w}$ denote the attack end and duration, and $\hat{\boldsymbol{S}}^{\tau,\hat{w}}$ and $\boldsymbol{V}^{\tau,\hat{w}}$ are the compromised value and attack magnitude from time step $\tau - \hat{w}$ to $\tau - 1$. This threat model is widely used in CPS and security papers [8], [14]–[17], [19], [23], [50], [63]. Defense usually focuses on one attack vector (such as sensor attacks in this paper), because one single defense for all kinds of attacks is generally impractical. Consequently, this paper is focused on sensor attacks, with the presumption that the remaining components are unaffected.

Note that our proposed method is not confined to specific kinds of attacks. Below just lists three example types of attacks (modified from [64]) used for the evaluation. A typical attack is bias attack, where the attacker modifies measurements by adding or subtracting fixed values, causing a stable drift: $v_i^t = v_i^{\tau-\hat{w}}$, where $t \in (\tau - \hat{w}, \tau)$. The second type is stealthy attack that changes sensor readings little by little starting from a negligible magnitude: $v_i^t = v_i^{t-1} + v_i^{\tau-\hat{w}-1}$, where $t \in [\tau - \hat{w}, \tau)$ and $v_i^{\tau-\hat{w}-1} = 0$. The last is random attack that changes sensor readings of two successive time steps randomly, representing sudden and unpredictable attacks: $v_i^t = v_i^{t-1} + U(0, \nu)$, where $t \in (\tau - \hat{w}, \tau)$ and $U(0, \nu)$ denotes a uniform distribution. ν is scale, which is smaller than $v_i^{\tau-\hat{w}}$.

IV. SYSTEM DESIGN OVERVIEW

This section presents the overview of our real-time sensor attack diagnosis system. Fig. 1 depicts the system design.

At each time step, the diagnosis system takes in the sensor readings, detects if there is an attack, and diagnoses an attack if it is detected. Our system has five modules working in sequence: preprocessing module, attention module, prediction module, detection module, and diagnosis module. There are two phases: offline and online. For the offline phase, we collect normal data from an application such as the ArduCopter drone in our evaluation, train the corresponding modules, and tune the parameters in all the modules. All module settings are fixed for a specific application once it starts to run. For the online phase, we implement all the modules in the control loop, which run online at every time step for real-time diagnosis. As mentioned, our diagnosis system requires little domain knowledge and is not confined to specific attack types.

The first task in the offline phase is to train the attention module and prediction module, which we jointly name as sensor reading predictor. The predictor is trained using collected normal or benign data from a CPS such that it accurately predicts sensor measurements when the CPS is running normally. After an attack is launched, the prediction deviates from readings, which is the basis for attack detection and diagnosis. Also, though the other three modules are trained in the offline phase, their parameters are determined using offline data or results before the online phase. Specifically, the maxima and minima of the training data are recorded for the preprocessing module, and the overall validation performance of the predictor is preserved for the detection module. The following briefly introduces each module.

The preprocessing module normalizes the system inputs. The reason is that different sensors and control channels have various value ranges, and overlarge values will dominate others in neural networks if no normalization. Hence, the module records the maxima and minima of the training data for all input dimensions and uses them for normalization.

The attention module is one of the two core modules in this system that exploits the attention mechanism to learn underlying relationships and potential attention transfer strategies. At every control step, it takes the scaled data as input and generates two outputs: latent representation and statistical attention score. The latent representation is the attended input that is embedded with the system running logic and correlations, which will be a better starting point for sensor prediction. On the other hand, the statistical attention score is the refined information of attention weights (or attention matrix) that will be used for attack diagnosis in the diagnosis module.

The prediction module takes as inputs of the latent representations and predicts the next time step's sensor measurements. Note that it does not predict control signals as we assume actuators remain intact. The prediction module can be instantiated as various forecasting neural networks, and most state-of-the-art can be adapted here. We do not claim novelty of this module. The prediction module accurately predicts sensor readings due to careful neural network layer design.

The detection module uses both the predicted and actual sensor readings to compute and monitor the residuals, or differences, between them. By analyzing these residuals, the detection module can detect any deviations from the expected sensor behavior and raise an alarm if they exceed a certain threshold, which typically indicates an attack.

Another core module of our proposed solution is the diagnosis module, which provides temporal attack diagnosis. Timing diagnosis is based on our observation that input changes tend to trigger fluctuations in attention weights. Thus, this module monitors the statistical attention score, records all possible attack onsets, and determines the most likely onset after an attack is detected. Considerable fluctuations in the attention score typically characterize possible onsets of attacks.

V. ATTACK DIAGNOSIS SYSTEM

As noted, our diagnosis system consists of five modules. This section first presents the design details of the two core modules, and then briefly describes the other three supporting modules. We only claim the novelty of the two core modules for temporal diagnosis, and design the supporting modules based on standard approaches. Actually, supporting modules are not strictly confined as presented, and other applicable models can be adopted instead. The modules' design and corresponding notations are shown in Fig. 2.

A. Attention Module

The attention module is a novel designed neural network that is based on the multi-head attention mechanism (MHA) [52]. We present the vanilla attention mechanism, our novel designs upon it, and the statistical method for attention score.

1) Vanilla MHA: At every time step, the MHA projects inputs $\boldsymbol{X}^{t,w}$ to query $\boldsymbol{Q}^{t,w}$, key $\boldsymbol{K}^{t,w}$, and value $\boldsymbol{V}^{t,w}$ by different fully-connected layers with weights $\boldsymbol{W}_q$, $\boldsymbol{W}_k$, and $\boldsymbol{W}_v$ while maintaining the same shapes. Then it uses dot product to compute attention weights $\boldsymbol{\mathcal{A}}^t$ between $\boldsymbol{Q}^{t,w}$ and $\boldsymbol{K}^{t,w}$, which denotes the attention paid from $\boldsymbol{Q}^{t,w}$ to $\boldsymbol{K}^{t,w}$ at individual time steps. The attention weights are used to reweight $\boldsymbol{V}^{t,w}$ and compute latent representation $\boldsymbol{O}^{t,w}$. Formally, we have:

$$\boldsymbol{Q}^{t,w} = \boldsymbol{X}^{t,w}\boldsymbol{W}_q \ , \boldsymbol{K}^{t,w} = \{\boldsymbol{X}^{t,w}\boldsymbol{W}_k\}^T \ , \boldsymbol{V}^{t,w} = \boldsymbol{X}^{t,w}\boldsymbol{W}_v \ ,$$
$$\boldsymbol{\mathcal{A}}^t = softmax(\boldsymbol{Q}^{t,w}\boldsymbol{K}^{t,w}) \ , \boldsymbol{O}^{t,w} = \boldsymbol{\mathcal{A}}^t\boldsymbol{V}^{t,w}\boldsymbol{W}_o \ ,$$

where $\boldsymbol{Q}^{t,w}$, $\boldsymbol{V}^{t,w}$, and $\boldsymbol{O}^{t,w} \in \mathbb{R}^{w\times N}$. $\boldsymbol{K}^{t,w} \in \mathbb{R}^{N\times w}$. $\boldsymbol{\mathcal{A}}^t \in \mathbb{R}^{w\times w}$. $\boldsymbol{W}_q$, $\boldsymbol{W}_k$, $\boldsymbol{W}_v$, and $\boldsymbol{W}_o \in \mathbb{R}^{N\times N}$.

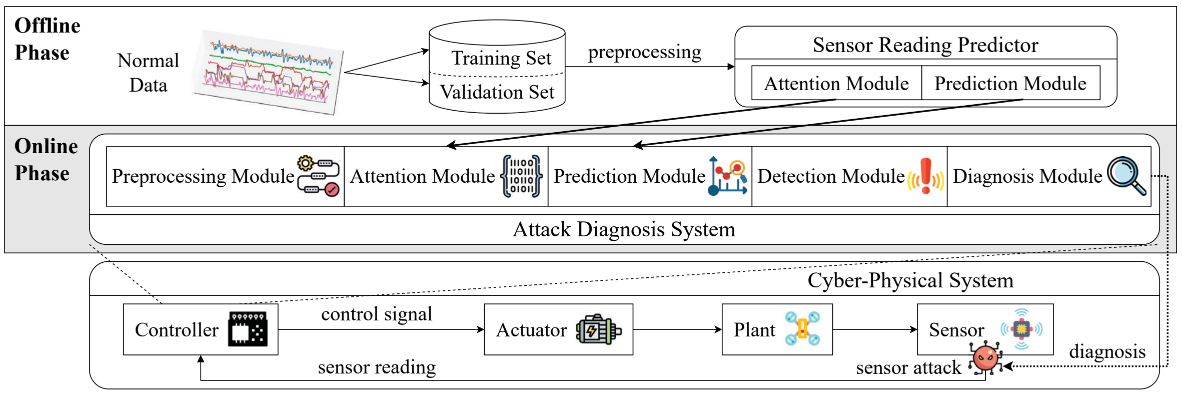

Fig. 1: **Sensor Attack Diagnosis System Overview.** A sensor reading predictor is trained using collected normal data during the offline phase. It is employed in our attack diagnosis system during the online phase and provides attack diagnosis for CPS sensor attacks along with other modules. The proposed system works in the control loop and reads measurements from sensors.

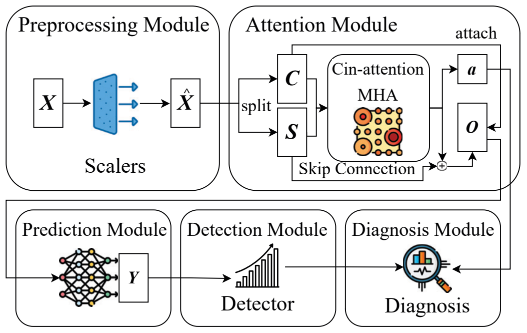

Fig. 2: **Module Design.** The core attention module splits input $\boldsymbol{X}$ into control signal $\boldsymbol{C}$ and sensor measurement $\boldsymbol{S}$, and uses cin-attention MHA, skip-connection, and split and attach mechanism to generate latent representation $\boldsymbol{O}$. Statistical attention score $\boldsymbol{a}$ is also recorded for temporal diagnosis.

The MHA allows various attention strategies for different sets of dimensions by dividing the inputs into distinct groups. If multiple heads are enabled, the $\boldsymbol{Q}^{t,w}$, $\boldsymbol{K}^{t,w}$, and $\boldsymbol{V}^{t,w}$ are split into H (the number of attention heads) subsets. Different heads compute the $\boldsymbol{\mathcal{A}}$ and $\boldsymbol{O}$ individually. The final $\boldsymbol{\mathcal{A}}$ are averaged from all the heads while $\boldsymbol{O}$ is the concatenation.

Rationale of using MHA. MHA is known to greatly extend neural networks' generalization ability and thus enables them to make precise and accurate predictions [65]–[68]. Moreover, the fundamental reasons that we apply MHA for attack diagnosis are 1) it interprets what the neural network focuses [69]–[71], and 2) it out-stands input fluctuations. The first reason is the basis of determining attack onsets. The second one can be partly understood in two ways: 1) matrix multiplication magnifies the changes, and 2) the well-trained attention network learns to focus more on the fluctuations.

2) Cin-attention MHA: The vanilla attention mechanism uses the same inputs $\boldsymbol{X}^{t,w}$ for $\boldsymbol{Q}^{t,w}$, $\boldsymbol{K}^{t,w}$, and $\boldsymbol{V}^{t,w}$, which is called self-attention. Our novel design is to use different inputs for $\boldsymbol{Q}^{t,w}$, $\boldsymbol{K}^{t,w}$, and $\boldsymbol{V}^{t,w}$, and we name it cin-attention. The cin-attention projects control signals $\boldsymbol{C}^{t,w}$ to $\boldsymbol{Q}^{t,w}$ using a full-connected layer with weights $\boldsymbol{W}_q \in \mathbb{R}^{N_c \times N_s}$, and projects sensor measurements $\boldsymbol{S}^{t,w}$ to $\boldsymbol{K}^{t,w}$ and $\boldsymbol{V}^{t,w}$ by full-connected layers with weights $\boldsymbol{W}_k \in \mathbb{R}^{N_s \times N_s}$ and $\boldsymbol{W}_v \in \mathbb{R}^{N_s \times N_s}$. Accordingly, $\boldsymbol{O}^{t,w} \in \mathbb{R}^{w \times N_s}$ and $\boldsymbol{W}_v \in \mathbb{R}^{N_s \times N_s}$. The proposed cin-attention offers several advantages over self-attention:

- Firstly, it reduces computation overheads by shrinking the input dimensions from N to N_s and N_c. This reduction in dimensionality enables our solution to process sensor and control data more efficiently, with little sacrificing performance.
- Secondly, cin-attention fully exploits the correlation between control signals and sensor measurements by projecting them as $\boldsymbol{Q}$ and $\boldsymbol{K}$ and $\boldsymbol{V}$. This allows our solution to more effectively capture the complex relationships between sensor readings and control signals, leading to improved performance.
- Finally, cin-attention reduces interference in attention weights by avoiding the placement of the same components ($\boldsymbol{C}$ or $\boldsymbol{S}$) in $\boldsymbol{Q}$ and $\boldsymbol{K}$. This is important because there may be a certain lag between $\boldsymbol{C}$ and $\boldsymbol{S}$, and if $\boldsymbol{Q}$ or $\boldsymbol{K}$ contains both $\boldsymbol{C}$ and $\boldsymbol{S}$, it becomes difficult for the attention weights to effectively address both two phases. By avoiding this interference, cin-attention enables attack temporal diagnosis (see experiments in Section VI-D7 and rationale in Section VI-D8).

3) Skip connection: Since the main knowledge of $\boldsymbol{O}$ is from sensor measurements $\boldsymbol{S}$, we design a skip-connection from $\boldsymbol{S}$ to $\boldsymbol{O}$ to help to learn: $\hat{\boldsymbol{O}}^{t,w} = SiLU(\boldsymbol{\mathcal{A}}^t\boldsymbol{V}^{t,w}\boldsymbol{W}_o + \boldsymbol{S})$, where SiLU [72] is a recently proposed activation function that has been shown to outperform other commonly used activation functions. By allowing $\boldsymbol{S}$ to flow directly to the latent representation, the skip connection help prevent the vanishing gradient problem and promote a stable training process.

4) Split and attach: Another novel aspect of our design is the concatenation of control signals $\boldsymbol{C}^{t,w}$ to the attention latent representation: $\boldsymbol{O}^{t,w} = \{\boldsymbol{C}^{t,w}, \hat{\boldsymbol{O}}^{t,w}\}$. Its design follows the objective of enriching the latent representation with a more comprehensive understanding of control signals. Though the latent representation already conveys some degree of control information because it is reweighted by control signals, the concatenation allows for a more complete representation. The split and attach approach leads to improved accuracy and robustness in the proposed system and its effectiveness is proved by our ablation study. The "split", on the other hand, refers to the splitting of $\boldsymbol{C}$ and $\boldsymbol{S}$ before applying the cin-attention. Note that the split and attach mechanism is not another attention type nor an independent component. Instead, it is an enhancement of the proposed cin-attention mechanism.

5) Statistical attention score: The statistical attention score $\boldsymbol{a}$ is designed to represent attention weights, and it plays a critical role in our proposed solution, providing accurate timing diagnosis of attacks. The attention weights $\boldsymbol{\mathcal{A}}$ are observed to fluctuate as input changes, and if equipped with the specially designed cin-attention, the fluctuations become discernible, aligning their movement with that of the inputs. Furthermore, our exploration has revealed the following: 1) fluctuations typically disrupt the previously stable pattern within $\boldsymbol{\mathcal{A}}$, influencing neighboring time steps by either attracting or redistributing weights; 2) The distribution of weights across dimensions within these fluctuations is uneven and lacks predictability. Consequently, the need arises for a metric that can identify the time steps when fluctuations manifest in their corresponding $\boldsymbol{\mathcal{A}}$. We term this metric the "statistical attention score." Crafting such a metric is challenging, as it must adeptly capture the fluctuations, thereby facilitating precise temporal diagnosis while carefully balancing the associated computational overhead. We find that keeping track of the maxima and minima in $\boldsymbol{\mathcal{A}}$ is an effective approach that balances these factors. The inclusion of minima is essential because the attention mechanism may allocate small attention weights to attacks at the beginning in certain attack settings, which are not easily predictable. Formally:

$$\boldsymbol{a}^t = \{a_u^t, a_l^t\} = \{max(\boldsymbol{\mathcal{A}}^t) - 1/w,\ 1/w - min(\boldsymbol{\mathcal{A}}^t)\},$$

where a_u^t and a_l^t denote the maxima and minima in the attention weights at time step t, and $1/w$ is the average weights under the softmax function. The term $1/w$ is a baseline to reduce variance. If the softmax is replaced and $1/w$ is no longer meaningful, a moving average can be used instead.

In summary, by carefully defining and tracking sudden fluctuations in attention weights, we can effectively record the possible attack onsets and provide accurate temporal diagnoses. The process to diagnose and reduce misdiagnosis is implemented in the diagnosis module.

B. Diagnosis Module

The novel-designed diagnosis module provides temporal attack diagnosis after attack detection by analyzing statistical attention score $\boldsymbol{a}$. Note that the diagnosis module keeps analyzing $\boldsymbol{a}$ and recording the most possible attack onset ϕ but only provides diagnosis after attack detection because providing misleading diagnoses for undetected attacks is meaningless.

The statistical attention score $\boldsymbol{a}$ varies in a small range when the CPS is running normally and greatly fluctuates after attacks because the attacked inputs do not conform with the model's learned knowledge and thus allocate more attention to abnormal time steps, leading to polarized attention weights. Thus to diagnose attacks temporally, the diagnosis module keeps track of all time steps where $\boldsymbol{a}$ suddenly increases or decreases. The diagnosis module maintains two moving windows for each dimension of $\boldsymbol{a}$ and computes the fluctuation by comparing the current statistical attention score $\boldsymbol{a}^t$ to the averages of these two windows. When fluctuation exceeds threshold λ, the time step t is recorded as the current estimation of attack start. λ is a hyper-parameter fine-tuning the diagnostic sensitivity. The determination of its optimal value often entails a trial-and-error approach. An iterative process involving data collection, λ adjustment, and performance evaluation can be employed to pinpoint its optimal range. While distinct attack types might necessitate varied λ values to exploit diagnostic efficacy, generally, one should opt for a λ that accommodates noise-induced effects rather than targeting specific types. We conclude by experiments that its optimal value range can be readily ascertained by starting at a modest λ value and employing a straightforward technique (e.d. bisection method).

Algorithm 1: Sensor Attack Temporal Diagnosis

```
Input: $\lambda$, $w'$ ;      // $\lambda$: threshold; $w'$: max length of moving window
Output: $\phi$ ;              // $\phi$: attack diagnosis
1  $\phi \leftarrow 1, \psi \leftarrow 1$; ;   // $\psi$: latest fluctuation
2  while $t \geq 2$ do
3      $t' \leftarrow max(\psi, t - w')$ ;  // $t'$: window start
4      foreach $i$ in $\{u, l\}$ do         // Iterate a
5          $m_i \leftarrow \frac{\sum_{\tau=t'}^{t-1} a_i^\tau}{t - t'}$ ;   // Average of a
6          $f_i \leftarrow \frac{|a_i^t - m_i|}{m_i}$;   // Relative change
7          if $f_i \geq \lambda$ then
8              if $t > \psi + 1$ then    // If change exceeds threshold and isn't a continuation of previous one.
9                  $\phi \leftarrow t$ ;   // $t$ is attack start
10             $\psi \leftarrow t$ ;        // Update $\psi$
```

The diagnosis algorithm, Algorithm 1, takes λ and w' as inputs and outputs ϕ as the temporal diagnosis after the attack is detected. Though the basis of temporal diagnosis is fluctuations, we cannot directly use the latest fluctuation as the diagnosis because an attack tends to invoke continuous fluctuations that last multiple time steps. Thus ϕ only points to the first time step in the continuous fluctuations and ψ is used as an indicator pointing to the latest fluctuation. After initiation in line 1, the algorithm runs at every time step. We use a moving window with dynamically-changing length to record the historical average attention score (line 5). This dynamic moving window ensures that the algorithm captures subtle fluctuations. The start of the dynamic window is obtained in line 3, and it depends on the latest fluctuation and w'. If the relative change between the current statistical attention score and the moving average (line 6) exceeds λ (line 7), the current time step t is considered as a fluctuation, and the algorithm updates ψ (line 10). If this fluctuation is not a continuation of the previous ones, the algorithm updates ϕ (line 9), which is the attack diagnosis after the attack is detected.

Note attention scores fluctuate both when attacks start and the reference state changes. However, the latter case is not caused by attacks, and detection will not raise an alarm and thus the diagnosis will not be triggered. Recorded fluctuations provide diagnosis only after detection.

C. Preprocessing Module

This module scales the inputs into a range of $[0, 1]$ with respect to the maxima and minima in the training set:

$$\hat{x}_i^t = (x_i^t - min(\tilde{\boldsymbol{x}}_i)/max(\tilde{\boldsymbol{x}}_i) - min(\tilde{\boldsymbol{x}}_i)),$$

where $\hat{\boldsymbol{x}}$ and $\boldsymbol{x}$ are the scaled and original value, and $\tilde{\boldsymbol{x}}$ denotes data in the training set. This module records and retains the maximum and minimum values, and uses the same values to scale all inputs during both training and online running. Given the same value in the same dimension of inputs, the module outputs the same normalized value. In the rest of the paper, we use $\boldsymbol{X}$ to denote the scaled inputs when not causing confusion.

D. Prediction Module

The prediction module utilizes a neural network to forecast the sensor reading at the next time step. We acknowledge that designing adaptable prediction neural networks can be challenging, but our work does not focus on novel network architectures. Thus, we select three commonly used models (FCN, RNN, and CNN) as representatives of prediction models. These models have a strong track record in time series prediction and demonstrate good performance on various datasets. By implementing these representative models, we can assess the effectiveness of our solution across different neural network architectures and showcase its compatibility with existing data-driven models. The prediction module takes the attention latent representation $\boldsymbol{O}^{t,w}$ as input and generates a forecast $\boldsymbol{y}^t$ for the sensor reading at the t^{th} time step.

Please note that neural network architectures are task-specific, but the purpose of this work is to show compatibility of the proposed diagnosis system. The comparison is fair as long as the same architecture is used. The hyper-parameter choices and architecture optimizations are out of the scope.

1) FCN: Fully connected networks (FCN), or feed-forward networks, consist of multiple layers with a weight matrix and a bias matrix for computing hidden states. Our implementation sequentially extracts temporal and spatial information. We employ an L-layer FCN to condense temporal information and an L'-layer FCN to extract spatial knowledge. The output of the first FCN is flattened and serves as input to the second FCN. SiLU activation is used for all layers. The output dimension of each layer is 60% (rounded down) of its previous layer for condensing, and the last layer has an output dimension of N_s. The values of L and L' are task-specific.

2) RNN: Long Short-Term Memory (LSTM) [73] is a famous variant of RNN that holds multiple hidden gates and states. We employ a $2N$-dimensional LSTM with a length of w, and its final output is fed into a 2-layer FCN (layer dimensions are N and N_s; activated by SiLU).

3) CNN: The Convolutional Neural Network (CNN), introduced by [74], applies convolutions to parts of the input and slides filters to cover the entire input. Our CNN implementation consists of two parts. The first part consists of L blocks, each composed of a 1D-convolution layer, a SiLU activation layer, and an average pooling layer. The last block's convolution layer has a spatial dimension of $2N$ and uses a kernel size of 3 with a padding length of 1 on both sides. The parameters of the previous layers increase linearly based on the value of L. The average pooling layers average every two consecutive time steps. The second part is an L'-layer FCN with SiLU activation. Each layer reduces the dimensions by half, except for the last layer, which has a dimension of N_s. L' depends on the dimension of the flattened layer.

E. Detection Module

Similarly, various methods can be applied in detection module such as [75]–[78]. The Cumulative Sum (CUSUM) method is used to balance the effectiveness, accuracy and overhead [15]. It computes residuals by element-wise squared error and assigns CUSUM score for each dimension:

$$e_i^t = (y_i^t - s_i^t)^2,\ \boldsymbol{d}^0 = \boldsymbol{0},\ d_i^t = max(0, d_i^{t-1} + e_i^t - \omega_i),$$

where $i \in [1, N_s]$, $\boldsymbol{0}$ is a vector of zeros, d_i^t is the CUSUM score for sensor i at time step t, and $\boldsymbol{\omega}$ is the drift to filter out noises and reduce false alarms. When d_i^t exceeds threshold η_i, the detection module resets d_i^t to 0 and raises an alarm, and thus detects the attack. In this work, we fix the $\boldsymbol{\omega}$ and $\boldsymbol{\eta}$ after offline training. How to adapt the drift and threshold on-the-fly is out of the scope, but adaptive methods such as [23], [61] can be easily applied. We determine the drift for each sensor as the minimum value that exceeds 99% of its prediction errors on the validation set. Then, the threshold is calculated by multiplying the drift with a ratio: $\boldsymbol{\eta} = R\boldsymbol{\omega}$. The R mainly determines the detection sensitivity. Decreasing R universally increases sensitivity for all systems. However, as this work primarily focuses on sensor attack temporal diagnosis, we use a fixed value for R and do not explore its optimization.

VI. Experiment

We conduct extensive experiments using a high-fidelity simulator and two numerical ones to avoid the unrealistic attack limitation discussed in Section II. A new metric, ATDE, is proposed for better evaluation of diagnosis errors. We evaluate the system performance by ATDE along with other suitable metrics, and experimental results show that our system provides accurate diagnosis with less than $2.4ms$ additional overhead on resource-limited Raspberry Pi.

A. Implementation

The sensor reading predictor (including the attention module and prediction module) is implemented under PyTorch v1.13.1 and CUDA v11.6, and its parameters are optimized by Adam [79]. The optimization target is to minimize the total prediction residual by adjusting model weights. After each epoch of training, the predictor is evaluated on the validation set. If its average validation loss (i.e. prediction residual on the validation set) stops decreasing, it reaches a local minimum. We set the initial learning rate to 1×10^{-3} and scale it down to its one-twentieth (5×10^{-5}) after the predictor reaches a local minimum for the first time. We stop training and finalize the predictor after the second time it reaches a local minimum.

To ensure the generalization of our approach, we train every system five times and then evaluate the performance by taking the average of their results. We keep the evaluation on the same platform that is equipped with Intel i7-8086K CPU @ 4.00 GHz and NVIDIA GeForce RTX 3080. Also, a Raspberry Pi 4 Model B is used when evaluating the computational overhead.

B. Datasets

We employ an ArduCopter drone [53] to collect a high-fidelity dataset, two numerical simulators [80] to collect numerical datasets, and a testbed for real-world evaluation. The key parameters are shown in Table I.

TABLE I: Dataset Key Parameters

Dataset	Plant	Hz	Nc	Ns
high-fidelity	ArduCopter drone	20	14	45
numerical	aircraft pitching	20	1	3
	quadrotor taking-off	10	1	9
testbed	Ackermann car	1	3	12

1) High-fidelity Dataset: We operate the drone within a defined area, specified by latitude ($[40.76368, 40.76729]$), longitude ($[-113.81360, -113.80882]$), and altitude ($[10, 50]$) ranges. We gather 1.1 million time steps of benign data for analysis. Out of these, 1.04 million time steps involve the drone following rapidly changing references to explore all possible system behaviors. The remaining data are taken with references set close to the area edges to capture terrain information. To create training and validation sets, we split the data with a 4:1 ratio. For the test set, we subject the drone to three types of sensor attacks (bias attack, stealthy attack, and random behavior attack) on six sensors (velocities in three directions on two IMUs). Each test record consists of 400 time steps of normal operation, followed by a 200-time step attack period ($\hat{w} = 200$). For the first two directions, $v_i^t = 2 \times 10^{-2}$ for the bias attacks, where $t \in [\tau - \hat{w}, \tau)$; $v_i^{\tau-\hat{w}} = 6 \times 10^{-5}$ for stealthy attacks, and $v_i^{\tau-\hat{w}} = 1 \times 10^{-2}$, $\hat{\nu} = 3 \times 10^{-3}$ for random behavior attacks, where i denotes the attacked sensor. The third direction has a small noise level, so we scale down all the magnitude by half. As a reference, the noise level of the three directions are $9 \times 10^{-3}, 4 \times 10^{-2}$, and 2×10^{-3}.

2) Numerical Dataset: Both simulators are restricted to safe sets during operation. We gather 200,000 seconds of training data and 50,000 seconds of validation data for each simulator. Additionally, we perform sensor attacks of varying magnitudes on three sensors for the aircraft simulator and four sensors for the quadrotor simulator. These attacks are used to construct test sets. Each test record comprises 400 time steps of normal operation, followed by a 200-time step attack period.

3) Testbed Dataset: We use an Ackerman car, depicted in Figure 3, to gather the testbed dataset. It employs an STM32 and a Raspberry Pi for control. The STM32 operates at 10Hz, acquiring data from onboard sensors (e.g. IMU, wheel speed) and managing motors and servos. Communication between the STM32 and Raspberry Pi occurs via UART. The Raspberry Pi runs Robot Operating System (ROS) and employs Proportional-Integral-Derivative (PID) controllers. The proposed diagnosis system is implemented on the Raspberry Pi to provide temporal diagnoses. The car rests on a stage, with wheels free to move. 5700 seconds of training data are collected, and $1/4$ of them are split out as the validation set. Random forces are enforced on the right back wheel as attacks.

C. Baselines and Metrics

Since there is no other system capable of handling temporal sensor attack diagnosis, we establish baselines that are lack of attention and diagnosis modules but share the same other three modules. The baselines use the detection time as their temporal diagnosis results. They are representations of existing common sensor reading predictors that cannot provide diagnoses.

Because temporal diagnosis is a new field and existing metrics cannot perfectly evaluate diagnosis models, we propose a new metric for evaluation:

- Average Temporal Diagnosis Error (ATDE): the absolute error between the attack starting time step and ϕ. When the system fails to deliver a temporal attack diagnosis, the corresponding ATDE is instead assigned the value of the total attack duration. While a small ATDE may not always be attainable for attacks of small magnitude (like the ones we are evaluating), a proficient temporal attack diagnosis system aims to minimize the ATDE. This showcases its ability to accurately diagnose the starting point of the attack.

Besides ATDE, three conventional metrics are considered for evaluation. However, given the difference between real-time continuous tasks (where we focus) and discrete machine learning tasks, some adjustments are made. We first describe the real-time tasks and then present the adjusted metrics. Every record contains a normal operation period followed by an attack period. The evaluated system runs at every time step. The record is false positive if the system raises alarms before the attack, and is true positive if the alarms are after the attack. The system does not stop after alarms, and it continues until the record ends. Thus it is possible for a record to be both false and true positive at the same time. Also, the system may raise multiple false (true) alarms for a single record, but such a record will still be considered as one false (true) positive record. As mathematical results, there is no true negative in this setting, and the sum of false negative rate and true positive rate is 1. The adjusted metrics are as follows:

- Average Detection Delay (ADD). The detection delay refers to the delay from the attack starting point to the earliest alarm from the detection module. If the system fails to detect the attack in a test record, the corresponding detection delay is set to the total duration of the attack.
- False Positive Duration (FPD): the ratio of the accumulated false alarm time steps to the total normal operation time steps. The FPD equals the false positive rate (FPR) used in discrete machine learning tasks.

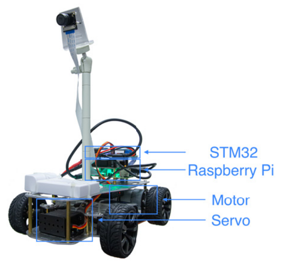

Fig. 3: **Testbed Architecture.**

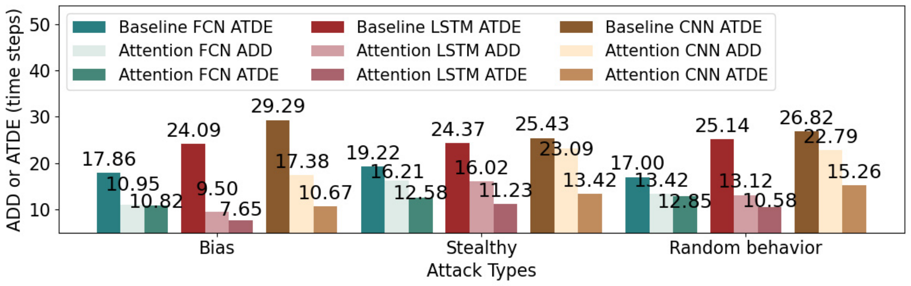

Fig. 4: **System Diagnosis Performance.** ADD: average detection delay. ATDE: average timing diagnosis error. The proposed diagnosis system provides accurate timing diagnosis, with up to 60.0% average improvement from baselines.

• True Positive Rate (TPR): $TPR = T^+/M$, where T^+ is the number of true positive records, and M is the number of total records. False positive rate (FPR) equals $1 - TPR$; thus we omit FPR in the evaluation.

D. High-fidelity Simulator Dataset Experimental Results

We set the window length w to 60, detection threshold ratio R to 10, diagnosis threshold λ to 0.1, and diagnosis max length of moving window w' to 20. For the systems using FCN as the prediction module, $L = L' = 4$. Thus the spatial output dimensions (last axis) of the first part are $36, 21, 12$, and 7, while the temporal output dimensions (second last axis) equal w. Inputs being flattened, the output dimensions of the second part are $252, 151, 90$, and 45. The systems using CNN have 3 blocks in the first part. Among them, the spatial dimensions of convolution layers are $78, 97$, and 118, kernel sizes are $7, 5$ and 3, and the padding lengths are $3, 2$, and 1. Thus the output dimensions of the 4-layer FCN are $472, 236, 118$, and 45.

TABLE II: **System Detection Performance and Sensor Diagnosis.** FPD: false positive duration. TPR: frue positive rate. Rnd Behavior: random behavior. "Attention" denotes the proposed attention and diagnosis module.

Attack	Bias		Stealthy		Rnd Behavior	
Metric(%)	FPD	TPR	FPD	TPR	FPD	TPR
FCN	0.44	100.00	0.58	100.00	0.69	100.00
Attention FCN	0.46	100.00	0.62	100.00	0.72	100.00
LSTM	0.07	99.44	0.37	100.00	0.36	100.00
Attention LSTM	0.20	100.00	0.46	100.00	0.49	100.00
CNN	0.04	97.33	0.35	100.00	0.35	100.00
Attention CNN	0.02	100.00	0.34	100.00	0.33	100.00

1) Validating attack diagnosis: We validate the proposed diagnosis system on three types of attacks and present the results in Figure 4 and Table II. Figure 4 illustrates the average timing diagnosis errors of the proposed systems, where FCN, LSTM, and CNN are used in prediction module, as well as their respective baselines. The average detection delay of the proposed system is also shown as a comparison. It is clear that our diagnosis system is capable of providing accurate timing diagnosis on sensor attacks, with an average of 32.8% (FCN) to 60.0% (LSTM) improvement on ATDE from baselines. It's worth noting that the attack magnitude is small, requiring at least seventeen time steps for detection. As a result, the noise can obscure attacks, making it difficult for the diagnosis system to provide further accurate timing diagnosis. Table II displays other metrics, indicating that the proposed system typically results in a limited increase in false positive duration. The FPD even decreases for systems using CNN. This experiment concludes that the proposed temporal diagnosis system provides accurate diagnosis while incurring negligible overheads.

Our evaluation primarily focuses on the improvement made by our diagnosis system compared to the baselines. We do not make direct comparisons among FCN, LSTM, and CNN since we do not claim any novelty in their implementation. However, we observe that the systems whose prediction module uses CNN exhibit smaller false positive duration while maintaining accurate timing diagnosis. Therefore, we solely consider the use of CNN for some of the subsequent experiments.

2) Computational overhead: We compare the computational overhead introduced by the proposed diagnosis system on the high-fidelity dataset and plot the result in Figure 5. Inference, i.e. predicting sensor readings, takes up the most computational time. The "other" refers to various additional computational costs, such as scaling inputs in the preprocessing module, applying the split and attach design, preparing data for devices, and conducting attack detection and diagnosis. We assess the additional computational overhead, i.e. the difference in time usage between the proposed system and its corresponding baseline. It is because the design of the preprocessing module, prediction module, and detection module is not restricted. While the total computational overhead fluctuates as their designs, the additional computational overhead remains consistent. The results also demonstrate that the proposed system incurs an additional computational time of less than $2.4ms$ on a Raspberry Pi 4 Model B, which is less than 5% of a control time step of a CPS operating at a frequency of $20Hz$.

3) Comparison of diagnosis threshold: The diagnosis threshold λ controls the sensitivity of the timing diagnosis, and its choice depends on task and scenarios. We evaluate its impact on all three attack types using CNN as the prediction module to conclude a suggestion about how to choose it. The λ varies in $[0.06, 0.18]$ with a step length of 0.02.

As plotted in Figure 6, different attack types have their

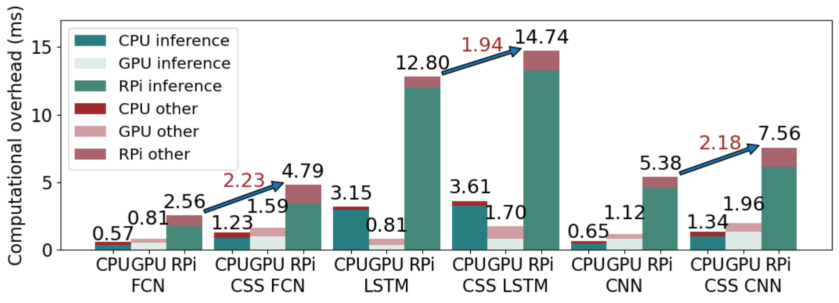

Fig. 5: **Computational Overhead.** RPi: Raspberry Pi. Blue arrows with red annotations: additional computational overhead on Raspberry Pi. For reference, a typical CPS controlled at a frequency of 20Hz has control time steps of 50ms.

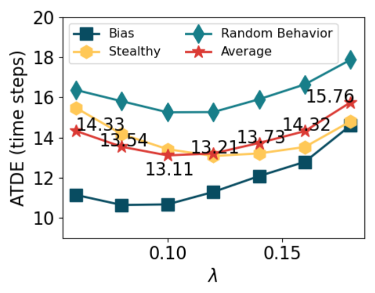

Fig. 6: **ATDE v.s. λ.** Optimal λ exists and must derive from experiments. Simple searching methods can be used.

respective optimal ranges of λ, but all of them are centered around 0.1. The differences can be attributed to the diverse nature of each attack type. Stealthy attacks gradually increase in magnitude, while bias attacks have sudden but stable modifications. Therefore, a larger λ can filter out more noise-invoked attention score fluctuations and provide more accurate timing diagnoses for stealthy attacks. Conversely, for bias attacks, a smaller λ is optimal. Random behavior attacks fall in between, with an intermediate optimal λ.

Even though an optimal value of λ leads to the best diagnostic performance, one cannot presume the attack types in real-world applications. Thus a conservative choice of λ is generally suitable for a wide range of situations as the difference among different λ is limited as long as the diagnosis algorithm is applied. The optimal λ improves less than 9% on ATDE than a conservative choice (0.06).

4) Comparison of the number of heads: We explore the relationship between system performance and the number of heads in MHA (H) and present the results in Table III. The results are averaged from all test records. Since the high-fidelity simulator is equipped with forty-five sensors, H can be chosen from $\{1, 5, 9, 45\}$ to keep the dimension of every head integral in MHA. We observe that a small and large H show similar performance, while an intermediate H degrades temporal diagnosis. Therefore, we conclude that H is a task-specific hyper-parameter that requires adjustment by experiments based on the scenario and expectations.

5) Comparison of window size: The window size w is a hyper-parameter that is highly dependent on tasks and systems. We compare the performance on different window sizes and the results are shown in Table IV. Please note that in this experiment, we only adjust the window size parameter (w) and do not adjust other parameters, such as the number of layers in CNN (L and L'). We observe that setting w to 60 yields the best performance for our proposed diagnosis system. The optimal window size may vary depending on the specific scenario and the overall system settings. One general rule is that a larger w means a larger amount of information in the inputs, which needs a more complex neural network architecture. Thus increasing w without adjusting the network structure accordingly cannot promise better performance. Also, a large w along with a complex network incurs heavy overhead for edge devices. Therefore, it is important to carefully evaluate and tune the window size parameter by experiments based on the specific use case and requirements of the system.

TABLE III: **System Performance v.s. H.** H: number of heads in MHA. Different choices of H result in non-equal performance. Intermediate H have worse temporal diagnoses.

H	Attack	ATDE(ts)	ADD(ts)	FPD(%)	TPR(%)
1	Bias	10.67	17.38	0.02	100.00
	Stealthy	13.42	23.09	0.34	100.00
	Random behavior	15.26	22.79	0.33	100.00
	Average	**13.11**	21.09	0.23	100.00
5	Bias	13.79	19.69	0.02	99.89
	Stealthy	15.55	23.90	0.34	100.00
	Random behavior	17.76	23.89	0.33	100.00
	Average	15.70	22.49	0.23	99.96
9	Bias	13.54	18.73	0.02	100.00
	Stealthy	18.43	24.27	0.35	100.00
	Random behavior	19.49	24.77	0.33	100.00
	Average	17.15	22.59	0.24	100.00
45	Bias	10.84	16.66	0.02	100.00
	Stealthy	15.55	23.45	0.34	100.00
	Random behavior	16.97	22.83	0.32	100.00
	Average	14.46	20.98	0.23	100.00

TABLE IV: **Comparison of window size.** w: input window size. Previous experiments set w=60.

Model	w	ATDE(ts)	ADD(ts)	FPD(%)	TPR(%)
Attention CNN	40	16.10	23.19	0.23	100.00
	50	16.13	23.88	0.22	100.00
	60	**13.11**	21.09	0.23	100.00
	70	13.95	22.05	0.25	99.63
	80	14.71	22.63	0.21	100.00

6) Comparison of prediction module network architecture: Similar to window size, network architecture in the prediction module also depends on tasks and systems. Even though we do not claim novelty in the prediction module, we compare the system performance on different CNN architectures and present the results in Table V. We conclude that too large or too small a model results in worse prediction and thus larger diagnosis error and detection delay. The optimal network architecture derives from experiments.

7) Ablation studies: Ablation studies are conducted to validate the improvement of our novel designs, i.e. cin-attention and split and attach. We use CSS to denote the proposed cin-attention because the $\boldsymbol{Q}$, $\boldsymbol{K}$, and $\boldsymbol{V}$ are $\boldsymbol{C}$, $\boldsymbol{S}$ and $\boldsymbol{S}$, respectively. Five variants of the attention module are designed

TABLE V: **Comparison of CNN Architecture.** L: number of blocks in CNN. Previous experiments set L=3.

Model	L	ATDE(ts)	ADD(ts)	FPD(%)	TPR(%)
Attention CNN	2	13.90	22.28	0.23	100.00
	3	**13.11**	21.09	0.23	100.00
	4	19.46	27.92	0.21	99.33
	5	20.33	29.93	0.21	99.48

for comparisons: 1) SCC: $\boldsymbol{Q} = \boldsymbol{S}$, $\boldsymbol{K} = \boldsymbol{V} = \boldsymbol{C}$, 2) AAA: $\boldsymbol{Q} = \boldsymbol{K} = \boldsymbol{V} = \boldsymbol{X}$, 3) SSS: $\boldsymbol{Q} = \boldsymbol{C}$, $\boldsymbol{K} = \boldsymbol{V} = \boldsymbol{S}$, 4) CCC: $\boldsymbol{Q} = \boldsymbol{C}$, $\boldsymbol{K} = \boldsymbol{V} = \boldsymbol{C}$, and 5) CSS^{-}: using cin-attention without split and attach (i.e., attaching $\boldsymbol{C}$ to $\boldsymbol{O}$). The results are shown in Figure 7 and Table VI. CNN is used as the prediction module, and the results are averaged from all attack records.

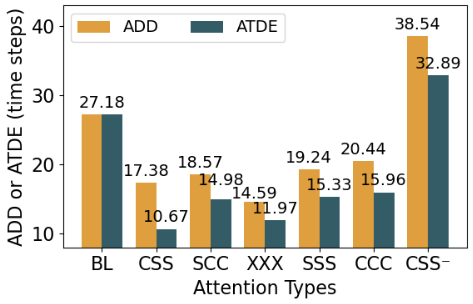

Fig. 7: **Ablation Study.** BL: baseline. CSS: cin-attention, i.e. $\boldsymbol{Q} = \boldsymbol{C}$, $\boldsymbol{K} = \boldsymbol{V} = \boldsymbol{S}$. SCC: $\boldsymbol{Q} = \boldsymbol{S}$, $\boldsymbol{K} = \boldsymbol{V} = \boldsymbol{C}$. XXX: $\boldsymbol{Q} = \boldsymbol{K} = \boldsymbol{V} = \boldsymbol{X}$. SSS: $\boldsymbol{Q} = \boldsymbol{K} = \boldsymbol{V} = \boldsymbol{S}$. CCC: $\boldsymbol{Q} = \boldsymbol{K} = \boldsymbol{V} = \boldsymbol{C}$. CSS^{-}: cin-attention without attaching $\boldsymbol{C}$ to $\boldsymbol{O}$.

TABLE VI: **Ablation Study.** CSS: the proposed model.

Type	BL	**CSS**	SCC	AAA	SSS	CCC	CSS-
FPD(%)	0.25	0.23	0.23	0.23	0.24	0.26	0.25
TPR(%)	99.11	100.00	99.93	100.00	100.00	100.00	94.44

Figure 7 illustrates that our novel designs lead to the smallest timing diagnosis error. The comparison between CSS and XXX proves the advantage of splitting the entire inputs $\boldsymbol{X}$ into control signals $\boldsymbol{C}$ and sensor measurements $\boldsymbol{S}$ as different inputs for MHA, while the comparison between CSS and CSS^{-} demonstrates the vital role of $\boldsymbol{C}$ and the benefit of attaching it to $\boldsymbol{O}$. Combing these two observations, we conclude that our novel split and attach mechanism yield the best attack diagnosis performance. On the other hand, Table VI indicates that the FPDs remain at a similar level across all systems. It is noteworthy that our novel designs result in a larger ADD compared to the XXX, as well as a worse SDA compared to the SCC and CCC. The former observation can be explained by the fact that information-rich attention module outputs $\boldsymbol{O}$ (XXX) can lead to better sensor predictions, but interference within the attention weights $\boldsymbol{\mathcal{A}}$ can impede timing diagnosis. This interference is due to the different patterns and phases between control signals and sensor measurements. Thus, directly applying MHA without adjustment with this interference leads to discordant attention scores, which impedes sensor attack diagnosis. The possible reason for the second observation is that $\boldsymbol{O}$ in SCC and CCC models are reweighted from $\boldsymbol{C}$; thus, the prediction module can digest the attached original $\boldsymbol{S}$ and become sensitive for sensor diagnosis. The reason behind the second observation could be that the attention module outputs $\boldsymbol{O}$ in SCC and CCC models are reweighted from $\boldsymbol{C}$, allowing the prediction module to incorporate the original sensor measurements $\boldsymbol{S}$ (by split and attach) and become more sensitive to sensor diagnosis.

8) Rationale of Cin-attention: We use a visual example to illustrate why cin-attention is an innovative design that empowers temporal diagnosis. Attention weights obtained from the proposed diagnosis system using cin-attention and from a baseline in the previous ablation study that uses self-attention are plotted, when the systems run on a bias attack test record. As shown in Figure 8, the attention weights on the left column, i.e. cin-attention, are more sensitive to sudden fluctuations and have less interference. However, the weights on the right column, i.e. self-attention, present a scenario-independent pattern that cannot distinguish normal inputs and attacked inputs. It concludes that the novel designed attention module along with the diagnosis module enables accurate temporal sensor attack diagnosis.

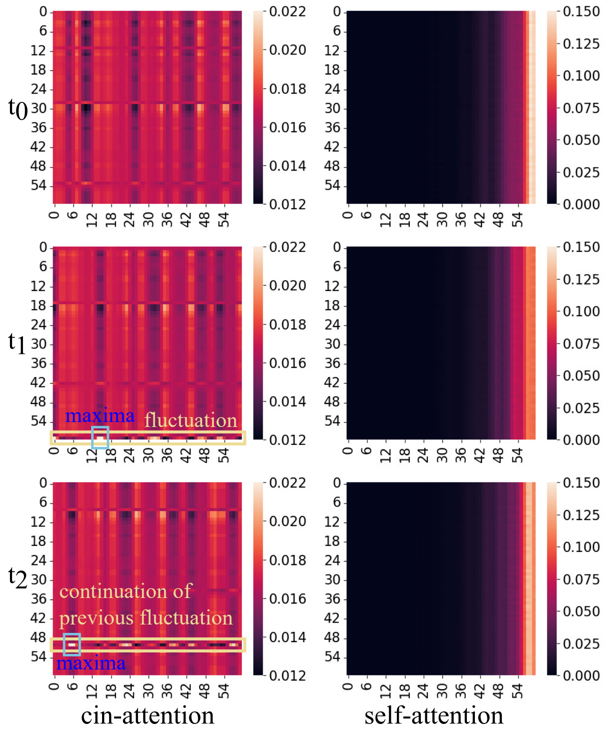

Fig. 8: **Rationale of Using Cin-attention.** t_0: 10 time steps before attack; t_1: when the attack happens; t_2: 10 time steps after attack. The left column explains why the novel-designed cin-attention empowers temporal sensor attack diagnosis: it aligns the movement of the attention weights with that of the inputs. The annotated maxima that are invoked by the attack help the statistic attention score $\boldsymbol{a}$ to record the fluctuation and diagnose the attack.

E. Numerical Datasets Experimental Results

We set $w = 60$, $R = 10$, $\lambda = 0.1$, and $w' = 20$. We use CNN in the prediction module, which has 3 blocks in the first part, whose convolution spatial dimensions are $5, 6$, and 8 for aircraft; $13, 16$, and 20 for quadrotor. Its kernel sizes and padding lengths are the same as the ones used in the high-fidelity dataset. Thus the output dimensions of the 4-layer FCN are $32, 16, 8$, and 3 for aircraft; $80, 40, 20$, and 9 for quadrotor.

1) Comparison of attack magnitude: We evaluate the ATDE of the proposed diagnosis system by attacks with nine attack magnitudes and plot the results in Figure 9. The bias attacks and random behavior attacks take $[0.6, 0.65, ..., 0.95, 1.0]$ times of the biggest magnitudes that are $2 \times 10^{-2}, 6.5 \times 10^{-4}$ and 1.2×10^{-2} for the three sensors. And the stealthy attacks take $[0.2, 0.3, ..., 0.9, 1.0]$ times of the biggest incremental that are $2 \times 10^{-3}, 8 \times 10^{-5}$ and 5×10^{-3}.

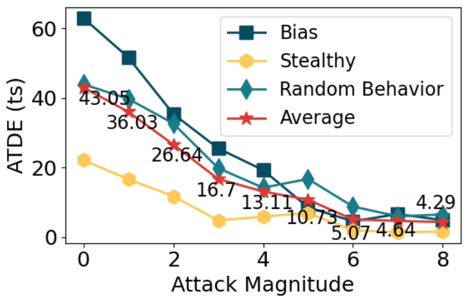

Fig. 9: **ATDE V.S. Attack Magnitude.** As attack magnitude increase, the ATDE for all attacks decreases.

We observe that as the attack magnitude increases, the temporal diagnosis accuracy improves. However, there are limits to these metrics, as environmental uncertainty and noise can contribute to less-than-perfect results. In some cases, extreme or outlier records may yield results that are not representative of the overall performance of the system.

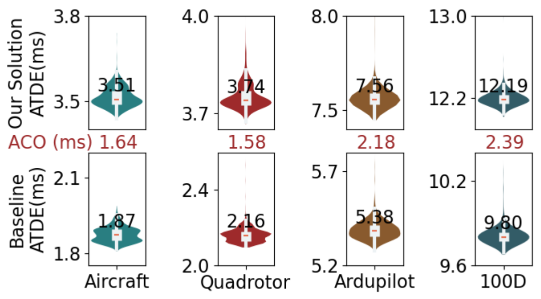

Fig. 10: **Computational Overhead V.S. Dimension.** ACO: additional computational overhead. 100D: a dummy 100-dimensional simulator. Annotated red line is the mean.

2) Computational Overhead of Various Dimensions: We present our diagnosis system's computational overhead (on Raspberry Pi) of all datasets we evaluated and a dummy simulator that has 100 dimensions ($N_c = 25$, $N_s = 75$, denoted by 100D). CNN is used. The result is shown in Figure 10 and it concludes that as the input dimension increases, the additional computational overhead of the proposed diagnosis system increases. However, even in a 100-dimensional highly complex system, our design only incurs less than 2.4ms overhead, concluding that the proposed system is lightweight.

F. Testbed Experimental Results

The CNN is used in the prediction module, and the settings of hyper-parameters remain the same as the previous experiments. A testbed experimental result is presented in Figure 11. As the attack starts at time step 152, the fluctuation is recorded by the proposed statistical attention score. Even though the detection module alarms at time step 166, the diagnosis module can provide an accurate temporal diagnosis.

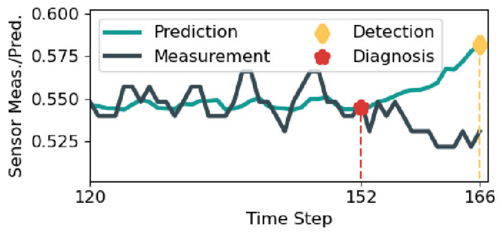

Fig. 11: **Testbed Sensor Attack Temporal Diagnosis.** An attack is launched at time step 152. Though it is not detected until time step 166, our diagnosis system records and diagnoses it accurately. The continuous fluctuation is omitted.

VII. CONCLUSION AND DISCUSSION

In this paper, we highlight the significance of sensor attack diagnosis in the field of CPS. Accurately identifying the sensor attack onset is a fundamental aspect ensuring CPS reliability and security. It also enables further research, including system estimation and attack recovery. Therefore, it is critical to develop efficient and accurate sensor attack diagnosis techniques.

To address this problem, we propose an attention-based novel sensor attack temporal diagnosis system that accurately determines the timing of sensor attacks. The system provides precise temporal diagnosis while incurring only minimal computational overhead, making it suitable for deployment on resource-limited edge systems. The proposed system is never a fixed system and its prediction and detection modules are easy to upgrade. State-of-the-art neural networks and detection methods can be applied as improvements. Our extensive experiments demonstrate the effectiveness and robustness of the proposed diagnosis system, highlighting its potential as a reliable tool for enhancing the security of CPS.

The proposed approach, like most other DL based ones, relies on comprehensive training datasets. Transfer learning, incremental learning, active learning, domain adaptation, and other techniques can be applied separately or jointly to mitigate dataset problems. We also notice that when the attack magnitude is small, the proposed sensor attack diagnosis system may record inaccurate attack onsets before the attack is detected. This can be attributed to environmental uncertainties and noise, which can mislead the system and lead to inaccurate diagnoses. The quest for further improvement remains an open problem, and various approaches such as denoising and pattern recognition can be explored to enhance the performance of the proposed temporal diagnosis system.

Acknowledgment

This work was supported in part by NSF CNS- 2333980 and NSF IUCRC ASIC center CNS-1822165. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the National Science Foundation (NSF).

References

[1] M. Pajic, J. Weimer, N. Bezzo, P. Tabuada, O. Sokolsky, I. Lee, and G. J. Pappas, "Robustness of attack-resilient state estimators," in *2014 ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS)*. IEEE, 2014, pp. 163–174.

[2] S. Chaterji, P. Naghizadeh, M. A. Alam, S. Bagchi, M. Chiang, D. Corman, B. Henz, S. Jana, N. Li, S. Mou *et al.*, "Resilient cyberphysical systems and their application drivers: A technology roadmap," *arXiv preprint arXiv:2001.00090*, 2019.

[3] T. He, L. Zhang, F. Kong, and A. Salekin, "Exploring inherent sensor redundancy for automotive anomaly detection," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*, 2020, pp. 1–6.

[4] L. Zhang, Z. Wang, and F. Kong, "Work-in-progress: Optimal checkpointing strategy for real-time systems with both logical and timing correctness," in *2022 IEEE Real-Time Systems Symposium (RTSS)*. IEEE, 2022, pp. 515–518.

[5] Z. Yu, Z. Kaplan, Q. Yan, and N. Zhang, "Security and privacy in the emerging cyber-physical world: A survey," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1879–1919, 2021.

[6] M. Wolf and D. Serpanos, "Safety and security in cyber-physical systems and internet-of-things systems," *Proceedings of the IEEE*, vol. 106, no. 1, pp. 9–20, 2017.

[7] J. Giraldo, D. Urbina, A. Cardenas, J. Valente, M. Faisal, J. Ruths, N. O. Tippenhauer, H. Sandberg, and R. Candell, "A survey of physics-based attack detection in cyber-physical systems," *ACM Computing Surveys (CSUR)*, vol. 51, no. 4, pp. 1–36, 2018.

[8] F. Akowuah and F. Kong, "Physical invariant based attack detection for autonomous vehicles: Survey, vision, and challenges," in *4th International Conference on Connected and Autonomous Driving (MetroCAD)*. IEEE, 2021.

[9] H. Zhu, Z. Yu, W. Cao, N. Zhang, and X. Zhang, "Powertouch: A security objective-guided automation framework for generating wired ghost touch attacks on touchscreens," in *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design*, 2022, pp. 1–9.

[10] Slammer worm crashed ohio nuke plant network: https://www.securityfocus.com/news/6767. [Online]. Available: https://www.securityfocus.com/news/6767

[11] J. P. Farwell and R. Rohozinski, "Stuxnet and the future of cyber war," *Survival*, vol. 53, no. 1, pp. 23–40, 2011.

[12] D. U. Case, "Analysis of the cyber attack on the ukrainian power grid," *Electricity Information Sharing and Analysis Center (E-ISAC)*, vol. 388, pp. 1–29, 2016.

[13] A. H. Rutkin, "spoofers use fake gps signals to knock a yacht off course," *MIT Technology Review*, 2013.

[14] T. Kim, C. H. Kim, A. Ozen, F. Fei, Z. Tu, X. Zhang, X. Deng, D. J. Tian, and D. Xu, "From control model to program: Investigating robotic aerial vehicle accidents with {MAYDAY}," in *29th USENIX Security Symposium (USENIX Security 20)*, 2020, pp. 913–930.

[15] R. Quinonez, J. Giraldo, L. Salazar, E. Bauman, A. Cardenas, and Z. Lin, "SAVIOR: Securing autonomous vehicles with robust physical invariants," in *29th USENIX Security Symposium (USENIX Security 20)*, 2020.

[16] A. A. Cardenas, S. Amin, and S. Sastry, "Secure control: Towards survivable cyber-physical systems," in *The 28th International Conference on Distributed Computing Systems Workshops (ICDCSW)*. IEEE, 2008, pp. 495–500.

[17] Y. Liu, P. Ning, and M. K. Reiter, "False data injection attacks against state estimation in electric power grids," *ACM Transactions on Information and System Security (TISSEC)*, vol. 14, no. 1, pp. 1–33, 2011.

[18] J. Giraldo, E. Sarkar, A. A. Cardenas, M. Maniatakos, and M. Kantarcioglu, "Security and privacy in cyber-physical systems: A survey of surveys," *IEEE Design & Test*, vol. 34, no. 4, pp. 7–17, 2017.

[19] Y. Luo, Y. Xiao, L. Cheng, G. Peng, and D. Yao, "Deep learning-based anomaly detection in cyber-physical systems: Progress and opportunities," *ACM Computing Surveys (CSUR)*, vol. 54, no. 5, pp. 1–36, 2021.

[20] L. Zhang, Z. Wang, and F. Kong, "Optimal checkpointing strategy for real-time systems with both logical and timing correctness," *ACM Transactions on Embedded Computing Systems*, 2023.

[21] H. Choi, W.-C. Lee, Y. Aafer, F. Fei, Z. Tu, X. Zhang, D. Xu, and X. Deng, "Detecting attacks against robotic vehicles: A control invariant approach," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 801–816.

[22] P. Guo, H. Kim, N. Virani, J. Xu, M. Zhu, and P. Liu, "Roboads: Anomaly detection against sensor and actuator misbehaviors in mobile robots," in *2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 2018, pp. 574–585.

[23] L. Zhang, Z. Wang, M. Liu, and F. Kong, "Adaptive window-based sensor attack detection for cyber-physical systems," in *2022 59th ACM/IEEE Design Automation Conference (DAC)*, 2022, pp. 1–6.

[24] C. M. Ahmed, M. Ochoa, J. Zhou, A. P. Mathur, R. Qadeer, C. Murguia, and J. Ruths, "Noiseprint: Attack detection using sensor and process noise fingerprint in cyber physical systems," in *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*, 2018, pp. 483–497.

[25] A. Goodge, B. Hooi, S.-K. Ng, and W. S. Ng, "Robustness of autoencoders for anomaly detection under adversarial impact." in *IJCAI*, 2020, pp. 1244–1250.

[26] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, and D. Pei, "Robust anomaly detection for multivariate time series through stochastic recurrent neural network," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 2828–2837.

[27] S. V. Thiruloga, V. K. Kukkala, and S. Pasricha, "Tenet: Temporal cnn with attention for anomaly detection in automotive cyber-physical systems," 2021.

[28] J. Gao, X. Song, Q. Wen, P. Wang, L. Sun, and H. Xu, "Robusttad: Robust time series anomaly detection via decomposition and convolutional neural networks," 2020.

[29] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Soderstrom, "Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding," *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, Jul 2018. [Online]. Available: http://dx.doi.org/10.1145/3219819.3219845

[30] M. O. Ezeme, Q. H. Mahmoud, and A. Azim, "Hierarchical attention-based anomaly detection model for embedded operating systems," in *2018 IEEE 24th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, 2018, pp. 225–231.

[31] M. Liu, L. Zhang, V. Phoha, and F. Kong, "Learn-to-respond: Sequence-predictive recovery from sensor attacks in cyber-physical systems," in *2023 IEEE Real-Time Systems Symposium (RTSS)*. IEEE, 2023.

[32] J. Audibert, P. Michiardi, F. Guyard, S. Marti, and M. A. Zuluaga, "Usad: Unsupervised anomaly detection on multivariate time series," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 3395–3404.

[33] D. Li, D. Chen, J. Goh, and S.-k. Ng, "Anomaly detection with generative adversarial networks for multivariate time series," *arXiv preprint arXiv:1809.04758*, 2018.

[34] J. Goh, S. Adepu, M. Tan, and Z. S. Lee, "Anomaly detection in cyber physical systems using recurrent neural networks," in *2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE)*. IEEE, 2017, pp. 140–145.

[35] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Soderstrom, "Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding," in *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 2018, pp. 387–395.

[36] S. Tariq, S. Lee, Y. Shin, M. S. Lee, O. Jung, D. Chung, and S. S. Woo, "Detecting anomalies in space using multivariate convolutional lstm with mixtures of probabilistic pca," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp. 2123–2133.

[37] J. Wang, D. Shi, Y. Li, J. Chen, H. Ding, and X. Duan, "Distributed framework for detecting pmu data manipulation attacks with deep autoencoders," *IEEE Transactions on smart grid*, vol. 10, no. 4, pp. 4401–4410, 2018.

[38] D. Li, D. Chen, B. Jin, L. Shi, J. Goh, and S.-K. Ng, "Mad-gan: Multivariate anomaly detection for time series data with generative adversarial networks," in *International Conference on Artificial Neural Networks*. Springer, 2019, pp. 703–716.

[39] A. A. Cárdenas, S. Amin, Z.-S. Lin, Y.-L. Huang, C.-Y. Huang, and S. Sastry, "Attacks against process control systems: risk assessment, detection, and response," in *Proceedings of the 6th ACM symposium on information, computer and communications security*, 2011, pp. 355–366.

[40] F. Kong, M. Xu, J. Weimer, O. Sokolsky, and I. Lee, "Cyber-physical system checkpointing and recovery," in *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*. IEEE, 2018, pp. 22–31.

[41] L. Zhang, X. Chen, F. Kong, and A. A. Cardenas, "Real-time attack-recovery for cyber-physical systems using linear approximations," in *2020 IEEE Real-Time Systems Symposium (RTSS)*. IEEE, 2020, pp. 205–217.

[42] R. Ma, S. Basumallik, S. Eftekharnejad, and F. Kong, "Recovery-based model predictive control for cascade mitigation under cyber-physical attacks," in *2020 IEEE Texas Power and Energy Conference (TPEC)*. IEEE, 2020, pp. 1–6.

[43] H. Choi, S. Kate, Y. Aafer, X. Zhang, and D. Xu, "Software-based realtime recovery from sensor attacks on robotic vehicles," in *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)*, 2020, pp. 349–364.

[44] L. Zhang, P. Lu, F. Kong, X. Chen, O. Sokolsky, and I. Lee, "Real-time attack-recovery for cyber-physical systems using linear-quadratic regulator," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 20, no. 5s, pp. 1–24, 2021.

[45] R. Ma, S. Basumallik, S. Eftekharnejad, and F. Kong, "A data-driven model predictive control for alleviating thermal overloads in presence of possible false data," *IEEE Transactions on Industry Applications*, 2021.

[46] S. K. Zhang, Lin, M. Liu, P. Lu, X. Chen, F. Kong, O. Sokolsky, and I. Lee, "Real-time data-predictive attack-recovery for complex cyber-physical systems," in *29th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*. IEEE, 2023.

[47] R. Wang, F. Kong, H. Sudler, and X. Jiao, "Hdad: Hyperdimensional computing-based anomaly detection for automotive sensor attacks," in *27th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS), Brief Industry Paper Track*. IEEE, 2021.

[48] F. Akowuah and F. Kong, "Real-time adaptive sensor attack detection in autonomous cyber-physical systems," in *27th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*. IEEE, 2021.

[49] W. Aoudi, M. Iturbe, and M. Almgren, "Truth will out: Departure-based process-level detection of stealthy attacks on control systems," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 817–831.

[50] M. Liu, L. Zhang, P. Lu, K. Sridhar, F. Kong, O. Sokolsky, and I. Lee, "Fail-safe: Securing cyber-physical systems against hidden sensor attacks," in *2022 IEEE Real-Time Systems Symposium (RTSS)*. IEEE, 2022, pp. 240–252.

[51] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, "Dive into deep learning," *arXiv preprint arXiv:2106.11342*, 2021.

[52] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[53] Ardupilot. [Online]. Available: https://ardupilot.org/

[54] M.-J. Kang and J.-W. Kang, "Intrusion detection system using deep neural network for in-vehicle network security," *PloS one*, vol. 11, no. 6, p. e0155781, 2016.

[55] M. Ashrafuzzaman, Y. Chakhchoukh, A. A. Jillepalli, P. T. Tosic, D. C. de Leon, F. T. Sheldon, and B. K. Johnson, "Detecting stealthy false data injection attacks in power grids using deep learning," in *2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC)*. IEEE, 2018, pp. 219–225.

[56] M. Kravchik and A. Shabtai, "Efficient cyber attack detection in industrial control systems using lightweight neural networks and pca," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 4, pp. 2179–2197, 2021.

[57] M. Keshk, B. Turnbull, N. Moustafa, D. Vatsalan, and K.-K. R. Choo, "A privacy-preserving-framework-based blockchain and deep learning for protecting smart power networks," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 8, pp. 5110–5118, 2019.

[58] E. Habler and A. Shabtai, "Using lstm encoder-decoder algorithm for detecting anomalous ads-b messages," *Computers & Security*, vol. 78, pp. 155–173, 2018.

[59] H. M. Song, J. Woo, and H. K. Kim, "In-vehicle network intrusion detection using deep convolutional neural network," *Vehicular Communications*, vol. 21, p. 100198, 2020.

[60] X. Guo, S. Han, X. S. Hu, X. Jiao, Y. Jin, F. Kong, and M. Lemmon, "Towards scalable, secure, and smart mission-critical iot systems: Review and vision:(special session paper)," in *2021 International Conference on Embedded Software (EMSOFT)*. IEEE, 2021, pp. 1–10.

[61] F. Akowuah and F. Kong, "Real-time adaptive sensor attack detection in autonomous cyber-physical systems," in *2021 IEEE 27th Real-Time and Embedded Technology and Applications Symposium (RTAS)*. IEEE, 2021, pp. 237–250.

[62] F. Akowuah, R. Prasad, C. O. Espinoza, and F. Kong, "Recovery-by-learning: Restoring autonomous cyber-physical systems from sensor attacks," in *2021 IEEE 27th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*. IEEE, 2021, pp. 61–66.

[63] A. Humayed, J. Lin, F. Li, and B. Luo, "Cyber-physical systems security—a survey," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 1802–1831, 2017.

[64] A. R. Javed, M. Usman, S. U. Rehman, M. U. Khan, and M. S. Haghighi, "Anomaly detection in automated vehicles using multistage attention-based convolutional neural network," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 4291–4300, 2020.

[65] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[66] B. Zhang, D. Xiong, and J. Su, "Neural machine translation with deep attention," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 1, pp. 154–163, 2018.

[67] J.-B. Cordonnier, A. Loukas, and M. Jaggi, "On the relationship between self-attention and convolutional layers," *arXiv preprint arXiv:1911.03584*, 2019.

[68] J. Baan, M. ter Hoeve, M. van der Wees, A. Schuth, and M. de Rijke, "Understanding multi-head attention in abstractive summarization," *arXiv preprint arXiv:1911.03898*, 2019.

[69] J. Vig, "Visualizing attention in transformer-based language representation models," *arXiv preprint arXiv:1904.02679*, 2019.

[70] S. Wiegreffe and Y. Pinter, "Attention is not not explanation," *arXiv preprint arXiv:1908.04626*, 2019.

[71] E. Voita, D. Talbot, F. Moiseev, R. Sennrich, and I. Titov, "Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned," *arXiv preprint arXiv:1905.09418*, 2019.

[72] S. Elfwing, E. Uchibe, and K. Doya, "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning," *Neural Networks*, vol. 107, pp. 3–11, 2018.

[73] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[74] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, "Handwritten digit recognition with a back-propagation network," *Advances in neural information processing systems*, vol. 2, 1989.

[75] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: identifying density-based local outliers," in *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, 2000, pp. 93–104.

[76] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[77] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation-based anomaly detection," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 6, no. 1, pp. 1–39, 2012.

[78] H. Permuter, J. Francos, and I. Jermyn, "A study of gaussian mixture models of color and texture features for image classification and segmentation," *Pattern recognition*, vol. 39, no. 4, pp. 695–706, 2006.

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

[80] L. Zhang, M. Liu, and F. Kong, "Simulation and security toolbox for cyber-physical systems," in *2023 IEEE 29th Real-Time and Embedded Technology and Applications Symposium (RTAS)*. IEEE Computer Society, 2023, pp. 357–358.