HD-I-IoT: Hyperdimensional Computing for Resilient Industrial Internet of Things Analytics

Onat Gungor[1,3], Tajana Rosing[1], and Baris Aksanli[3]

[1]Department of Electrical and Computer Engineering, University of California, San Diego
[3]Department of Electrical and Computer Engineering, San Diego State University

***Abstract*—Industrial Internet of Things (I-IoT) enables fully automated production systems by continuously monitoring devices and analyzing collected data. Machine learning (ML) methods are commonly utilized for data analytics in such systems. Cyberattacks are a grave threat to I-IoT as they can manipulate legitimate inputs, corrupting ML predictions and causing disruptions in the production systems. Hyperdimensional (HD) computing is a brain-inspired ML method that has been shown to be sufficiently accurate while being extremely robust, fast, and energy-efficient. In this work, we use non-linear encoding-based HD for intelligent fault diagnosis against different adversarial attacks. Our black-box adversarial attacks first train a substitute model and create perturbed test instances using this trained model. These examples are then transferred to the target models. The change in the classification accuracy is measured as the difference before and after the attacks. This change measures the resiliency of a learning method. Our experiments show that HD leads to a more resilient and lightweight learning solution than the state-of-the-art deep learning methods. HD has up to 67.5% higher resiliency compared to the state-of-the-art methods while being up to 25.1× faster to train.**

I. Introduction

Industry 4.0 revolutionized monitoring, analysis, and automation of production systems through smart technology [1]. It is mainly powered by the Industrial Internet of Things (I-IoT). I-IoT is the interconnection of smart devices that enables full automation, remote monitoring, and predictive maintenance. However, small-scale I-IoT devices with their limited computation and communication capabilities make I-IoT system an easy target for possible cyberattacks. System vulnerabilities (e.g., network protocols, insecure data transfer and storage) can be discovered by an attacker, and used to sabotage communication, do physical damage, alter existing data, or prevent asset availability [2]. The average estimated losses were $10.7 million per breach of data among manufacturing organizations in Asia Pacific in 2019 [3]. To minimize these costs, cyber-security measures should be taken such as cyber-security awareness training, keeping software updated, installing firewalls, using strong passwords and others [4].

Abundant system monitoring data in I-IoT systems makes data-driven predictive maintenance (PDM) popular. Machine learning (ML) methods are commonly used for identifying best maintenance schedules [5]. Intelligent fault diagnosis (IFD) is a key data-driven PDM application that finds and classifies different fault types before they occur. The success of these ML-based methods heavily depends on input data. Adversarial attacks against ML methods manipulate legitimate inputs and force the trained model to produce incorrect outputs leading to incorrect predictions. Since ML is in the center of intelligent fault diagnosis, these attacks may have serious consequences such as undetected failures [6]. Hence, there is a need for novel intelligent learning solutions that can stay resilient against various adversarial attacks.

In this work, we propose hyperdimensional (HD) computing as a resilient learning solution against different black-box adversarial attacks for intelligent fault diagnosis (IFD). Our black-box attack is based on a transferable attack strategy [7]. We first train a substitute deep learning model, a wide deep convolutional neural network (WDCNN), and create artificial test samples using this trained model. We then transfer these instances to the target methods (e.g., LSTM, GRU, HD). In testing, we measure the classification accuracy of the target models before and after the attacks. The accuracy change gives us the resiliency of a target method. We show that HD is the most resilient and lightweight method, outperforming all deep learning (DL) methods on commonly used CWRU Bearing dataset [8]. HD is up to 67.5% more resilient and 25.1× faster during training compared to the state-of-the-art DL methods.

II. Related Work

Industrial Internet of Things (I-IoT) is an adaptation of traditional IoT for production environments focusing on machine-to-machine communication, big data, and machine learning for higher system efficiency and reliability. I-IoT systems are often insufficiently secure and vulnerable due to off-the-shelf communication protocols [2]. An adversary can exploit these vulnerabilities to arrange a cyberattack. There are various cyberattacks against I-IoT systems such as denial of service, side channel, and attacks against ML [2]. We focus on attacks against ML where an attacker corrupts the collected data or model parameters leading to worse prediction performance. These attacks are dangerous since data analytics is an indispensable part of I-IoT systems. They can result in serious outcomes, e.g., undetected failures in a system [6].

Predictive maintenance determines an optimal maintenance schedule based on time-to-failure prediction of industrial assets [5]. Data-driven predictive maintenance utilizes historical data to create ML models. Intelligent fault diagnosis (IFD) is a branch of data-driven PDM which classifies different fault types in advance. There are various IFD methods, such

as convolutional neural network (CNN) [9], long short-term memory (LSTM) [10], gated recurrent unit (GRU) [11], ensemble learning [12], etc. However, adversarial attacks against deployed ML models can lead to serious consequences for a PDM system such as delayed maintenance or replacement of a machine [6]. Mode and Hoque [6] analyze the impact of different adversarial attacks in remaining useful life (RUL) prediction domain. They show that adversarial attacks can lead to 5× worse prediction performance. This work is limited in terms of the number of attack and DL models. Gungor et al. [13] propose stacking ensemble learning framework as a resilient learning solution which can select the most resilient base learners efficiently. Their framework can perform well in the presence of cyber-attacks and has up to 60% higher resiliency compared to the most resilient individual ML method. In our paper, we propose a lightweight learning solution HD which can stay resilient against different adversarial attacks.

Hyperdimensional (HD) computing was introduced as a brain-inspired learning solution for robust and efficient learning [14]. HD encodes raw data into high-dimensional vectors (i.e., hypervectors) and then performs simple operations in this space, e.g., element-wise addition, dot product. HD has been employed in a range of applications such as activity recognition [15], speech recognition [16], and biomedical signal processing [17]. However, the security aspect of emerging HD classifiers has not been clearly understood. Yang and Ren [18] showed that HD can be vulnerable to even minimally-perturbed adversarial samples. To strengthen the security of HD, they used adversarial training and retraining. In our work, we show that by using non-linear encoding, HD can stay resilient against different black-box adversarial attacks. To the best of our knowledge, HD has not been used in PDM domain where it can provide both lightweight and robust learning solution. In our work, we propose HD as a resilient and efficient learning solution to adversarial attacks. Our experiments show that HD is more resilient against various adversarial attacks compared to the state-of-the-art DL methods while bringing significant computational advantage.

III. Adversarial Attack Methods

We adapt a transferable black-box attack strategy [7] where we select a single substitute model and craft examples using this model. We then transfer the created examples to target models. In this strategy, attacker does not need to know anything about the target models, yet have an access to the input data. Black-box attack represents a more realistic attack scenario in a way that the attacker can be an outsider, with limited or no knowledge about the internal system details [7].

We select 4 different state-of-the-art adversarial attack methods: fast gradient sign method (FGSM), basic iterative method (BIM), momentum iterative method (MIM), and robust optimization method (ROM). The selected methods utilize loss gradient information to craft adversarial examples by adding different amounts of perturbation. In that sense, they represent different attack scenarios. An adversary, who is able to access the I-IoT system data (e.g., collected sensor measurements), can implement these methods using a substitute model and harm the prediction performance of target models without being detected. We define the following variables to explain different attack strategies: θ (parameters of the substitute model), x (collected sensor input data), y (fault types), $J(\theta, x, y)$ (cost function to train the substitute model).

A. Fast Gradient Sign Method (FGSM)

FGSM first calculates the gradient of the cost function with respect to the input of the neural network [19]. Adversarial examples are then created based on the gradient direction:

$$\ddot{x} = x + \epsilon * sign(\nabla_x J(\theta, x, y)) \tag{1}$$

where $\ddot{x}$ represents the crafted adversarial examples and ϵ denotes the amount of the perturbation.

B. Basic Iterative Method (BIM)

BIM is an improved version of FGSM where FGSM is applied multiple times with really small step size [20]. BIM perturbs the original data in the direction of the gradient multiplied by the step size α:

$$\ddot{x} = x + \alpha * sign(\nabla_x J(\theta, x, y)) \tag{2}$$

where α is obtained by dividing the amount of perturbation (ϵ) by the number of iterations (I): $\alpha = \epsilon / I$. Then, BIM clips the obtained time series elements to ensure that they are in the ϵ-neighborhood of the original time series.

C. Momentum Iterative Method (MIM)

Momentum Iterative Method (MIM) solves underfitting and overfitting problems in FGSM and BIM respectively by integrating momentum into the BIM [21]. At each iteration i, the variable g_i gathers the gradients with a decay factor μ:

$$g_{i+1} = \mu * g_i + \frac{\nabla_x J(\theta, \ddot{x}_i, y)}{\|\nabla_x J(\theta, \ddot{x}_i, y)\|_1} \tag{3}$$

where the gradient is normalized by its L_1 distance. The perturbed data for the next iteration is created in the direction of the sign of g_{i+1} with a step size α:

$$\ddot{x}_{i+1} = \ddot{x}_i + \alpha * sign(g_{i+1}) \tag{4}$$

D. Robust Optimization Method (ROM)

The goal of a supervised learning problem is to find model parameters θ that minimize the empirical risk $\mathbb{E}_{(x,y)\sim\Xi}[J(\theta, x, y)]$ where Ξ is the underlying supervised data distribution. However, this formulation cannot handle the change in input data. To solve this problem, set of allowed perturbations Δ is introduced. Then, empirical risk formulation is modified by feeding samples from the distribution Ξ directly into the loss function J leading to the following min-max (saddle point) optimization formulation [22]:

$$\min_{\theta} \zeta(\theta), \ where \ \zeta(\theta) = \mathbb{E}_{(x,y)\sim\Xi}[\max_{\delta\in\Delta} J(\theta, x+\delta, y)] \tag{5}$$

Here, while inner maximization finds an adversarial version of a given data point x that achieves a high loss, outer

minimization discovers model parameters to minimize the adversarial loss given by the inner attack problem. To solve the robust optimization problem, ROM replaces every instance with its FGSM-perturbed counterpart.

IV. Proposed Framework

In this section, we present our hyperdimensional (HD) computing and black-box attack frameworks.

A. Hyperdimensional (HD) Computing

HD is a computing paradigm inspired by the fact that brains take sensing data and map it into a high dimensional sparse representation before analysis. HD mimics this by mapping each data point into high-dimensional space $\mathcal{D}$. All computational tasks are performed in $\mathcal{D}$-space using simple operations such as element-wise additions and dot products. Fig. 1 illustrates HD learning framework which contains 3 main stages: encoding, training, and inference.

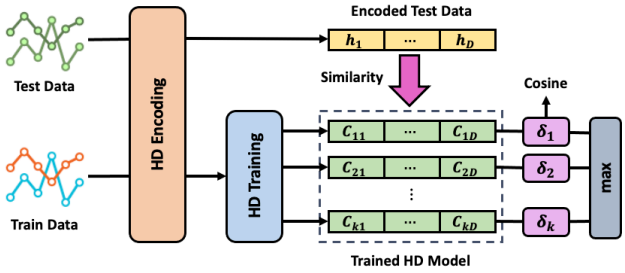


Fig. 1: HD Learning Framework

1) Encoding: The first step of HD is to encode input data into hyper-vectors. Most of the proposed encoding methods [23] linearly combine the hyper-vectors corresponding to each feature, resulting in sub-optimal classification quality [24]. In this work, we use non-linear encoding which considers the non-linear interactions between the feature values with different weights and exploits the kernel trick. This encoding approach is based on a study which shows that the Gaussian kernel function can be approximated by the dot product of two vectors [25]. Assume that an input vector in original space $\vec{\Xi} = \{\xi_1, \xi_2, \dots \xi_n\} \in \mathbb{R}^n$. Encoded high-dimensional vector is represented as $\vec{\mathcal{H}} = \{h_1, h_2, \dots, h_\mathcal{D}\} \in \mathbb{R}^\mathcal{D}$ where $\mathcal{D} \gg n$. Encoding from $\vec{\Xi}$ to h_i is based on the following equation:

$$h_i = cos(\vec{\Xi} \cdot \vec{\mathcal{B}}_i + b_i) sin(\vec{\Xi} \cdot \vec{\mathcal{B}}_i) \tag{6}$$

where $\vec{\mathcal{B}_k}$s are randomly chosen of dimension $\mathcal{D} \simeq 10k$ and $b_i \sim \mathcal{U}(0, 2\pi)$. That is, $\vec{\mathcal{B}}_{kj} \sim \mathcal{N}(0, 1)$ and $\delta(\vec{\mathcal{B}}_{k1}, \vec{\mathcal{B}}_{k2}) \simeq 0$, where δ is the cosine similarity.

2) Training: The second step of HD, model training consists of two steps to generate hyper-vectors representing each class. The first step, **initial training**, performs element-wise addition of all encoded hyper-vectors in each existing class. Let's assume that $\vec{\mathcal{H}}_i$ is the encoded hyper-vector of input i. We know that each input i belongs to a class j. Hence, $\vec{\mathcal{H}}_i^j$ denotes the hyper-vector for input i from class j. In the initial training, HD simply adds all hyper-vectors of the same class to generate the final model hyper-vector:

$$\vec{C}^j = \eta\vec{\mathcal{H}}_0^j + \eta\vec{H}_1^j + \dots = \sum_m \eta\vec{\mathcal{H}}_m^j \tag{7}$$

where η is the learning rate. This process is also called as one-pass training since each input is visited only once and added to class hyper-vectors. The second step of HD training, **retraining**, performs model adjustment by iteratively going through the training dataset. Retraining is beneficial for HD to improve the prediction accuracy. During this step, the encoded hyper-vector of each input is created again, and its similarity with the existing class hyper-vectors is checked. If HD misclassifies, say that $\vec{\mathcal{H}}^j$ from class $\vec{C}^j$ is predicted as class $\vec{C}^k$, it updates its model as follows:

$$\begin{aligned} \vec{C}^j &= \vec{C}^j + \eta\vec{\mathcal{H}}^j \\ \vec{C}^k &= \vec{C}^k - \eta\vec{\mathcal{H}}^j \end{aligned} \tag{8}$$

which means that the information of $\vec{\mathcal{H}}^j$ causing misclassification to $\vec{C}^k$ is discarded.

3) Inference: In the last step, HD checks the similarity of each encoded test data with the class hyper-vector. Most commonly, cosine similarity is used for the similarity check although other metrics (e.g., Hamming distance) could be suitable based on the problem. To calculate cosine similarity between hyper-vector $\vec{\mathcal{H}}$ and class hyper-vector $\vec{C}^j$:

$$cos(\vec{\mathcal{H}}, \vec{C}^j) = \frac{\vec{\mathcal{H}} \cdot \vec{C}^j}{\|\vec{\mathcal{H}}\| \cdot \|\vec{C}^j\|} \tag{9}$$

which is calculated by the dot product of the $\vec{\mathcal{H}}$ and $\vec{C}^j$ divided by the product of these two vectors' lengths. As an output of this step, HD provides the most similar class.

B. Black-box Attack Framework

In this work, we use a black-box transferable attack strategy which first trains a substitute model and crafts new test instances using the trained substitute model. For full list of DL methods used in this paper, you can refer to the Section V-B. As our substitute model, we select wide deep convolutional neural network (WDCNN) since it is one of the most commonly used DL methods in intelligent fault diagnosis [12], [26]. For our black-box attack setting, we assume that an adversary has access to the training and test data, yet does not know anything about the attacked (target) models. We illustrate our black-box attack framework in Fig. 2. Attacker first trains the substitute model (WDCNN) and use the trained model to create perturbed test data. Attacker can employ different attack strategies to obtain perturbed test data (see Section III for the attack strategies used in this paper). Afterwards, adversary sends these crafted examples to the target models in testing time. In Fig. 2, we give long short-term memory (LSTM) as the target model for illustration purposes. However, note that there is a pool of pretrained target models adversary is not aware of (thus black-box attack). Attacker simply sends the created examples to the target models to see if the attack will

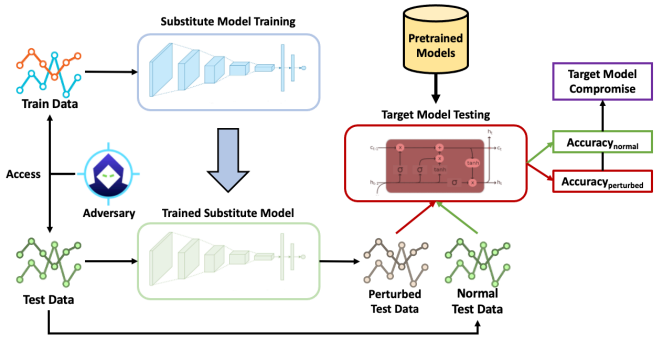


Fig. 2: Black-box Attack Framework

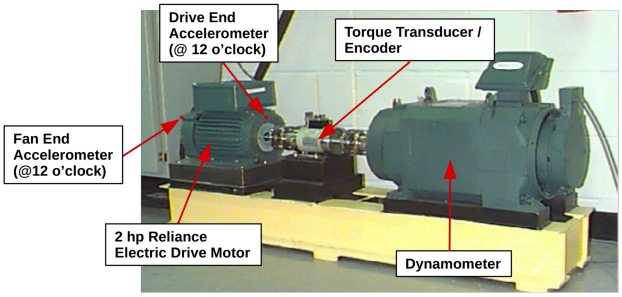


Fig. 3: CWRU Experimental Test Apparatus [27]

be successful or not. We measure the attack success based on change in test data classification accuracy before and after the attack where accuracy is defined as:

$$Accuracy = \frac{\text{Number of correct predictions}}{\text{Total number of test samples}} \quad (10)$$

Change in accuracy gives us the resiliency of a learning method which we measure by the metric called *Compromise* which is formulated as:

$$Compromise = \frac{Accuracy_{normal}}{Accuracy_{perturbed}} \quad (11)$$

where $Compromise > 1$ (under the assumption that attacks lead to worse prediction performance). The **smaller** the compromise value is, the **more resilient** the model becomes against the adversarial attack. For instance, given two methods RNN and LSTM, and their compromise values 5 and 2 respectively, we can conclude that LSTM is **more resilient** against the adversarial attack. If we have M number of adversarial attacks ($M > 1$), then we need to calculate the mean compromise value for each learning method as follows:

$$Compromise_{mean} = \left(\sum_{i=1}^{M} \frac{Accuracy_{normal}}{Accuracy^{i}_{perturbed}} \right) / M \quad (12)$$

Because we have multiple attack strategies (where $M = 4$), mean compromise gives a more accurate idea about single model resiliency. Overall, we obtain mean compromise value for each learning method and use this metric for our experimental analysis. Furthermore, to show the HD resiliency improvement, we define the following improvement metric:

$$Improvement = \left(\frac{Compromise_{DL} - Compromise_{HD}}{Compromise_{DL}} \right) \quad (13)$$

where $Compromise_{DL}$ denotes the single DL model mean compromise value, and $Compromise_{HD}$ is the HD mean compromise. We report the improvement in percentage (%). Improvement signifies the resiliency of our HD learner against adversarial attacks compared to a single DL model.

V. Experimental Analysis

A. Dataset Description

We use Case Western Reserve University (CWRU) Bearing dataset [8], a widely used benchmark for fault diagnosis. Fig. 3 represents the experimental test apparatus to collect this dataset. The data were collected from both the drive end accelerometer and the fan end accelerometer at 12k samples/second over a range of motor loads (from 0 hp to 3 hp). Both datasets (drive end and fan end) contain 19,800 training and 750 test samples. Bearing used in this experiment has three components: rolling element, inner race, and outer race. 9 different fault types are provided in the dataset based on the fault diameter (0.007, 0.014, and 0.021 inches) and the component (plus the normal bearing condition).

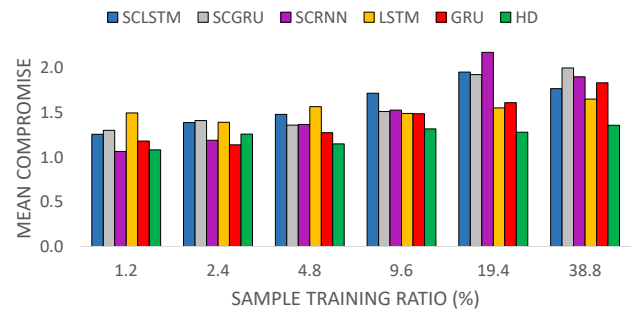
B. Experimental Setup

Selected Deep Learning (DL) Methods: We select 9 different DL methods: long short-term memory (LSTM) [10], gated recurrent unit (GRU) [11], wide deep convolutional neural network (WDCNN) [28], convolutional recurrent neural network (CRNN, CLSTM, CGRU) [27], and simplified CRNN (SCRNN, SCGRU, SCLSTM) [27]). We cover a good range of DL methods, increasing the generalizability of our study.

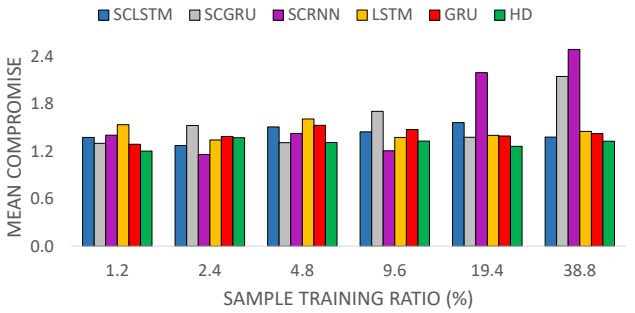
Adversarial Attack Methods: We select the following parameters for our adversarial methods: fast gradient sign, basic iterative, momentum iterative, and robust optimization [21], [22]: amount of perturbation (ϵ): 0.1, step size (α): 0.001, number of iterations (I): 100, decay factor (μ): 1.

Parameter Selection: For both DL methods and HD, we use a sliding time window of size 100 with a number of epochs of 100. We replicate each experiment 10 times and report average values where we run all experiments on a PC with 16 GB RAM and an 8-core 2.3 GHz Intel Core i9 processor. For **DL methods**, the following hyper-parameters are selected: *Adam* optimizer with learning rate 0.001, *relu* activation function, batch size of 16. For **HD**, we select the following parameters: encoding: non-linear, hyper-vector dimension size: 10,000, learning rate: 0.005, number of epochs: 100, similarity metric: cosine.

Number of Training Samples: We measure the selected methods' resiliency by using different number of training samples while using the whole test data. Specifically, our smallest experiment configuration uses 1.2% (240 samples)



(a) **Drive End Dataset**



(b) **Fan End Dataset**

Fig. 4: Mean Compromise Analysis

of the whole training data where we double this ratio until we reach approximately 38.8% (7680 samples). We call this ratio sample training ratio (STR) for the rest of this section. Considering different STRs is crucial for intelligent fault diagnosis (IFD) since it might not always be feasible to label fault data for the whole training dataset. IFD methods should perform well under limited supervision [12].

C. Resiliency Analysis

Mean Compromise Comparison: We analyze the resiliency of selected DL models and HD using mean compromise metric defined in Equation 12. Fig. 4 shows the mean compromise values of the 6 most resilient learning methods under different sample training ratios for drive end (Fig. 4a) and fan end (Fig. 4b) datasets. In these figures, x-axis represents the selected STRs and y-axis gives the mean compromise values where each color represents a different learning method. We can observe that the mean compromise of a DL method changes significantly. To illustrate, while SCRNN (represented with purple color) is the most resilient method for really small STRs (e.g., 1.2, 2.4), it becomes the least resilient algorithm as we reach the maximum STR. For drive end dataset, we can observe that HD is the most resilient method for STRs greater than 2.4%. As the number of training samples increases, HD becomes a more resilient method compared to the DL algorithms. We can make a similar observation for fan end dataset as well. For STRs larger than 9.6%, HD is the most resilient method against adversarial attacks. To present a single mean compromise

TABLE I: Average Compromise Comparison

Method / Dataset	Drive end	Fan end
CGRU	2.77	2.52
CLSTM	2.52	2.69
CRNN	2.30	2.77
SCLSTM	1.59	1.42
SCGRU	1.58	1.56
SCRNN	1.54	1.69
LSTM	1.52	1.45
GRU	1.42	1.42
HD	**1.24**	**1.30**

TABLE II: Average and Maximum Resiliency Improvement of HD over DL Methods

	Drive end		Fan end	
DL Method	**Average (%)**	**Maximum (%)**	**Average (%)**	**Maximum (%)**
CGRU	**51.34**	**61.9**	44.25	64.5
CLSTM	48.1	54.1	48.3	**67.5**
CRNN	42.1	52.4	**49.9**	64.4
SCLSTM	21.1	34.5	8.1	19.3
SCGRU	20.2	33.5	14.4	38.1
SCRNN	15.3	41.1	14.8	52.4
LSTM	18.5	27.6	10.0	21.7
GRU	10.9	25.9	8.0	14.2

value (for better understanding), we calculate the average of mean compromise values over all STRs. Table I presents these average compromise values for all the methods. When we compare DL methods (i.e., all methods excluding HD), we can observe that recurrent neural network structures are the most resilient. Specifically, GRU is the most resilient DL method with an average compromise value of 1.42 for both datasets. This observation can be attributed to the fact that our hybrid DL model structures contain convolutional layers. Since our crafted examples are based on wide deep convolutional neural network, more test examples are able to deceive hybrid methods. Most importantly, according to Table I, **HD** is the **most resilient** method on average outperforminng other DL methods at both datasets. This shows that HD provides a resilient learning solution performing well even under different black-box adversarial attack configurations.

HD Resiliency Improvement: We calculate HD resiliency improvement over the selected DL models using Equation 13 for each STR configuration. Then, we find the maximum and average improvement for each DL method. Table II demonstrates the HD average and maximum resiliency improvement over the selected DL methods. For drive end experiment, HD improves DL model resiliency by up to 61.9% where this number rises to 67.5% for fan end dataset. Compared to the most resilient DL method (GRU), HD improves the resiliency by up to 25.9% and 14.2% for drive end and fan end data sets respectively. We are able to verify that HD provides a resilient learning solution against adversarial attacks.

Training Overhead Comparison: Table III presents target models' training time (in seconds). In this table, each row represents a different target model (where the models are ordered in decreasing training overhead) and each column corresponds to the selected STR. We can observe that **HD** is the **most lightweight** model across all STRs. In the last column of this table, we share the average (across sample

TABLE III: Target Models Training Time Comparison

	Sample Training Ratio (%)						Average
Method	**1.2**	**2.4**	**4.8**	**9.6**	**19.4**	**38.8**	**Normalized**
LSTM	151.0	366.1	514.8	980.4	1882.4	3294.8	25.1
GRU	161.5	307.7	451.5	861.5	1623.3	3165.5	23.0
CLSTM	16.0	31.8	81.0	174.6	413.7	820.2	5.4
CGRU	15.7	61.2	94.2	196.9	360.8	727.5	5.1
SCLSTM	15.6	28.9	52.2	119.5	246.4	488.6	3.3
SCGRU	15.2	27.6	50.6	106.8	228.1	448.5	3.1
CRNN	10.8	19.4	36.1	77.4	167.6	327.1	2.2
SCRNN	7.7	13.8	26.9	53.6	98.6	198.3	1.4
HD	**6.0**	**10.2**	**18.6**	**37.5**	**72.0**	**141.8**	**1.0**

training ratios) normalized training time with respect to HD. HD can achieve up to 25.1× training speed up compared to DL methods. Compared to the most resilient DL method (GRU), HD brings 23× faster training. By this analysis, we can conclude that HD also provides a computationally efficient learning solution while being resilient to adversarial attacks. Training overhead is especially critical for I-IoT systems since data is collected continuously. When new data arrives, learning models require retraining to keep their prediction performances at a certain level [29]. HD can alleviate this retraining overhead due to its lightweight feature.

VI. Conclusion

Industrial Internet of Things (I-IoT) is a notion that facilitates monitoring, automation and reliability of smart devices in production environments. I-IoT ensures that these devices are connected to the Internet which helps collecting big data, and utilizing this data to extract useful information. However, these interconnection brings numerous security vulnerabilities which can be exploited by an attacker to harm the system. Attacks against ML deceive ML methods with fake inputs leading to worse prediction performance. Hyperdimensional (HD) computing is a novel learning solution which is robust against noise. In this paper, we utilize HD to stay resilient against created black-box attack scenarios. Our experiments show that HD can improve the resiliency of the state-of-the-art DL methods by up to 67.5%. HD can also achieve up to 25.1× training speed up compared to DL methods, providing a lightweight learning solution. This means that HD can still perform well and efficiently under adversarial attacks which leads to more accurate replacement and maintenance decisions even under cyberattacks.

Acknowledgement

This work has been funded in part by NSF, with award numbers #1911095, #2003277, and #2003279.

References

[1] "What is industry 4.0." https://insights.sap.com/what-is-industry-4-0.
[2] N. Tuptuk and S. Hailes, "Security of smart manufacturing systems," *Journal of manufacturing systems*, vol. 47, pp. 93–106, 2018.
[3] "Understanding the cybersecurity threat landscape in asia pacific." https://news.microsoft.com/apac/features/cybersecurity-in-asia/.
[4] W. He, I. Ash, M. Anwar, L. Li, X. Yuan, L. Xu, and X. Tian, "Improving employees' intellectual capacity for cybersecurity through evidence-based malware training," *Journal of Intellectual Capital*, 2019.
[5] O. Gungor, T. S. Rosing, and B. Aksanli, "Opelrul: Optimally weighted ensemble learner for remaining useful life prediction," in *2021 IEEE International Conference on Prognostics and Health Management (ICPHM)*, pp. 1–8, IEEE, 2021.
[6] G. R. Mode and K. A. Hoque, "Crafting adversarial examples for deep learning based prognostics (extended version)," *arXiv preprint arXiv:2009.10149*, 2020.
[7] S. Bhambri *et al.*, "A survey of black-box adversarial attacks on computer vision models," *arXiv preprint arXiv:1912.01667*, 2019.
[8] "Case western reserve university bearing data center website." https://csegroups.case.edu/bearingdatacenter/pages/project-history.
[9] X. Wang, D. Mao, and X. Li, "Bearing fault diagnosis based on vibro-acoustic data fusion and 1d-cnn network," *Measurement*, vol. 173, p. 108518, 2021.
[10] J. Lei, C. Liu, and D. Jiang, "Fault diagnosis of wind turbine based on long short-term memory networks," *Renewable energy*, vol. 133, pp. 422–432, 2019.
[11] Y. Tao, X. Wang, R.-V. Sánchez, S. Yang, and Y. Bai, "Spur gear fault diagnosis using a multilayer gated recurrent unit approach with vibration signal," *IEEE Access*, vol. 7, pp. 56880–56889, 2019.
[12] O. Gungor, T. Rosing, and B. Aksanli, "Enfes: Ensemble few-shot learning for intelligent fault diagnosis with limited data," in *2021 IEEE Sensors*, pp. 1–4, IEEE.
[13] O. Gungor, T. Rosing, and B. Aksanli, "Stewart: Stacking ensemble for white-box adversarial attacks towards more resilient data-driven predictive maintenance," *Computers in Industry*, vol. 140, p. 103660, 2022.
[14] M. Imani *et al.*, "Exploring hyperdimensional associative memory," in *HPCA*, pp. 445–456, IEEE, 2017.
[15] O. J. Räsänen and J. P. Saarinen, "Sequence prediction with sparse distributed hyperdimensional coding applied to the analysis of mobile phone use patterns," *IEEE transactions on neural networks and learning systems*, vol. 27, no. 9, pp. 1878–1889, 2015.
[16] M. Imani *et al.*, "Voicehd: Hyperdimensional computing for efficient speech recognition," in *ICRC*, pp. 1–8, IEEE, 2017.
[17] A. Moin *et al.*, "A wearable biosensing system with in-sensor adaptive machine learning for hand gesture recognition," *Nature Electronics*, vol. 4, no. 1, pp. 54–63, 2021.
[18] F. Yang and S. Ren, "Adversarial attacks on brain-inspired hyperdimensional computing-based classifiers," *arXiv preprint arXiv:2006.05594*, 2020.
[19] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.
[20] A. Kurakin, I. Goodfellow, S. Bengio, *et al.*, "Adversarial examples in the physical world," 2016.
[21] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li, "Boosting adversarial attacks with momentum," in *IEEE conference on computer vision and pattern recognition*, pp. 9185–9193, 2018.
[22] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.
[23] A. Rahimi, A. Tchouprina, P. Kanerva, J. d. R. Millán, and J. M. Rabaey, "Hyperdimensional computing for blind and one-shot classification of eeg error-related potentials," *Mobile Networks and Applications*, vol. 25, no. 5, pp. 1958–1969, 2020.
[24] Z. Zou, H. Alimohamadi, F. Imani, Y. Kim, and M. Imani, "Spiking hyperdimensional network: Neuromorphic models integrated with memory-inspired framework," *arXiv preprint arXiv:2110.00214*, 2021.
[25] A. Rahimi, B. Recht, *et al.*, "Random features for large-scale kernel machines.," in *NIPS*, vol. 3, p. 5, Citeseer, 2007.
[26] A. Zhang, S. Li, Y. Cui, W. Yang, R. Dong, and J. Hu, "Limited data rolling bearing fault diagnosis with few-shot learning," *IEEE Access*, vol. 7, pp. 110895–110904, 2019.
[27] A. Shenfield and M. Howarth, "A novel deep learning model for the detection and identification of rolling element-bearing faults," *Sensors*, vol. 20, no. 18, p. 5112, 2020.
[28] W. Zhang, G. Peng, C. Li, Y. Chen, and Z. Zhang, "A new deep learning model for fault diagnosis with good anti-noise and domain adaptation ability on raw vibration signals," *Sensors*, vol. 17, no. 2, p. 425, 2017.
[29] O. Gungor, T. S. Rosing, and B. Aksanli, "Dowell: Diversity-induced optimally weighted ensemble learner for predictive maintenance of industrial internet of things devices," *IEEE Internet of Things Journal*, 2021.