Image-Based Zero-Day Malware Detection in IoMT Devices: A Hybrid AI-Enabled Method

Zhangying He, Hossein Sayadi
Department of Computer Engineering and Computer Science
California State University, Long Beach, CA, USA

***Abstract*—Healthcare systems have recently utilized the Internet of Medical Things (IoMT) to assist intelligent data collection and decision-making. However, the volume of malicious threats, particularly new variants of malware attacks to the connected medical devices and their connected system, has risen significantly in recent years, which poses a critical threat to patients' confidential data and the safety of the healthcare systems. To address the high complexity of conventional software-based detection techniques, Hardware-supported Malware Detection (HMD) has proved to be efficient for detecting malware at the processors' micro-architecture level with the aid of Machine Learning (ML) techniques applied to Hardware Performance Counter (HPC) data. In this work, we examine the suitability of various standard ML classifiers for zero-day malware detection on new data streams in the real-world operation of IoMT devices and demonstrate that such methods are not capable of detecting unknown malware signatures with a high detection rate. In response, we propose a hybrid and adaptive image-based framework based on Deep Learning and Deep Reinforcement Learning (DRL) for online hardware-assisted zero-day malware detection in IoMT devices. Our proposed method dynamically selects the best DNN-based malware detector at run-time customized for each device from a pool of highly efficient models continuously trained on all stream data. It first converts tabular hardware-based data (HPC events) into small-size images and then leverages a transfer learning technique to retrain and enhance the Deep Neural Network (DNN) based model's performance for unknown malware detection. Multiple DNN models are trained on various stream data continuously to form an inclusive model pool. Next, a DRL-based agent constructed with two Multi-Layer Perceptrons (MLPs) is trained (one acts as an Actor and another acts as a Critic) to align the decision of selecting the most optimal DNN model for highly accurate zero-day malware detection at run-time using a limited number of hardware events. The experimental results demonstrate that our proposed AI-enabled method achieves 99% detection rate in both F1-score and AUC, with only 0.01% false positive rate and 1% false negative rate.**

Keywords*—*Hardware Performance Counters, IoT/IoMT, Deep Learning, Reinforcement Learning, Zero-Day Malware Detection.

I. Introduction

Recent years have witnessed the growth of the Internet of Things (IoT) devices with an expected annual growth rate of more than 20% from 2022 to 2030 [1]. The capability to connect edge devices and process data on the device or in fog computing has enabled a broad range of applications for IoT from smart grid and smart cities to smart healthcare systems [2]. Internet of Medical Things (IoMT) is integrated with traditional healthcare systems to provide real-time enhanced healthcare service that has never been done before to patients, and data collected from the IoMT devices are processed within an edge cloud environment supporting healthcare providers to make smarter, and more accurate decisions. Meanwhile, recent years have also seen a vast growth of new security vulnerabilities making the IoMT systems accessible targets for an increasing number of complicated cyber attacks [3], [4], [5], [6], [7] ,[8]. With malicious software (a.k.a. malware) utilization continuing to rise across different application domains, the development of efficient malware detection techniques has grown to be more crucial as they feature as an early protection mechanism to guard the integrity and confidentiality of the authenticated users' data [9], [10], [11].

Due to inefficiency of software-based detection methods, security should have been delegated to the systems' hardware level building a bottom-up solution rather than an afterthought solution. Therefore, Hardware-Assisted Malware Detection (HMD) methods [12], [13], [14], [15], [16] have emerged to address the inefficiency of software-based solutions, including static analysis, incompetence in detecting obfuscated attacks, and excessive computational overheads on resource-limited systems. Recent developments in the field of Artificial Intelligence (AI) including Machine Learning (ML) and Deep Learning (DL) have been sparked by a huge increase in the volume of data in modern computer systems, leading to applications of these intelligent methodologies in a range of computing fields, including security [10], [17], [18]. HMD methods have demonstrated that malware can be differentiated from normal programs by classifying anomalies using AI/ML techniques in low-level microarchitectural feature spaces captured by Hardware Performance Counters (HPCs) [12], [13], [14], [19]. With AI-based protection countermeasures, the hardware systems could analyze patterns of applications to proactively respond to altering behavior at run-time and prevent conceivable attacks.

In this work, we address important challenges of hardware-assisted cybersecurity in the context of malware detection in emerging computing systems such as biomedical and IoMT devices. IoT-related malware attacks are skyrocketing. In 2020, among the 5.6 billion malware attacks detected by SonicWall [20], 56.9 million came to IoT attacks, a 66% increase from a year before, and this trend is predicted to grow even more in the coming years. These malware attacks homes, government, education, healthcare, and retail by finding vulnerabilities in networks and applications to launch attacks. According to [21], [22], 73% of healthcare providers currently use biomedical equipment with a legacy OS, which are using outdated technologies that are vulnerable to security. Attackers actively exploit vulnerabilities executing system commands with root privileges that deploy malware attacks, change file permission, and connect with the attacker-controller server to perform more malicious activities. When one device is infected, it could quickly spread to other devices in the same IoMT network to cause potential cybersecurity threats such as data interception or manipulation to further comprise the security of healthcare services. These attacks can be expanded to critical life support and medical devices used in healthcare systems [3], [4].

In addition, unknown (zero-day) types of malware attacks are a major challenge to IoT devices [20]. A zero-day attack is a type of serious cybersecurity threat that exploits software security vulnerabilities that are undocumented in the training database of the detection mechanism [23], [24] and therefore is a long-standing challenge. Furthermore, existing ML-based HMD strategies typically select the best model to defend against intrusions, then periodically update the model to adapt it to new data variants. Other studies have found that besides adapting to data, we can also adapt to the model's reaction to data by training a run-time model selector to adaptively select the best model for that data, which proves to be more efficient because even a weak detector can contribute to unknown zero-day malware detection during run-time [25]. There is a need of a structured method to automate the malware detection process in IoT/IoMT devices and adapt the detection model to the ever-changing data and new zero-day malware types.

In this research, we propose a hybrid, intelligent, and adaptive framework based on deep learning and reinforcement learning techniques, referred to as *DRL-HMD*, for online hardware-assisted zero-day malware detection in IoMT devices. Today's biomedical computing systems are equipped with modern commercial microprocessors (e.g, Intel, ARM) to process different programs [26], [27]. Therefore, instead of expensive-to-implement, slow, static, and vulnerable anti-virus mechanisms, to detect malicious software signature, in this work we will propose a novel intelligent solution that utilizes the patterns of low-level hardware features captured by two hardware components including hardware performance counters and embedded trace buffers registers in biomedical devices' processors. *DRL-HMD* first converts tabular hardware performance counters data to small-size images, and leverages transfer learning technique to train accurate Deep Neural Network (DNN) based detectors. Multiple DNN models are trained on various stream data continuously to form a model pool. Then, *DRL-HMD* is followed by an effective value-based deep reinforcement learning-guided decision-maker that adaptively selects the best performing DNN model for detecting unknown malware signatures in the newly generated data stream from IoMT devices. In particular, *DRL-HMD* formulates the hardware malware detection as an RL problem by examining the ability of an autonomous agent in learning to take optimal actions/decisions for online malware detection to maximize a reward function while interacting with a stochastic environment. The proposed defense mechanism is equipped with a detection-sensitive selection model that takes into account the detection rates of the F-measure and AUC of the base DNN models and determines the best malware detector at run-time.

Furthermore, the DRL-based agent constructed with two Multi-Layer Perceptrons (MLPs) is trained (that one acts as an Actor and another acts as a Critic) to align the decision of selecting the most optimistic DNN model during the run-time, which enhances the system performance using a small number of micro-architectural features captured at run-time by existing HPCs. The adaptive DRL agent is trained based on the experience replay data (ERD) data with the guidance of the reward policy of $F1 * AUC$ to select the most intelligent DNN model that can detect malware and benign with higher performance. From the DRL agent's perspective, selecting the model with the highest $F1 * AUC$ achieves its goal of reaching the highest accumulated rewards. During the inference, zero-day test ERD data are fed to the DRL agent, and the agent's actions at each time step are tracked so the malware classification metrics are collected to evaluate the overall *DRL-HMD* system's detection performance. For a thorough analysis, eight classical and three DNN-based models and one DRL agent are implemented and their efficiencies are comprehensively analyzed across different evaluation metrics for detecting unknown malware.

The remainder of this paper is organized as follows. Section II presents an overview of the proposed methodology. Section III discusses the evaluation criteria and results analysis. Finally, Section IV concludes this study.

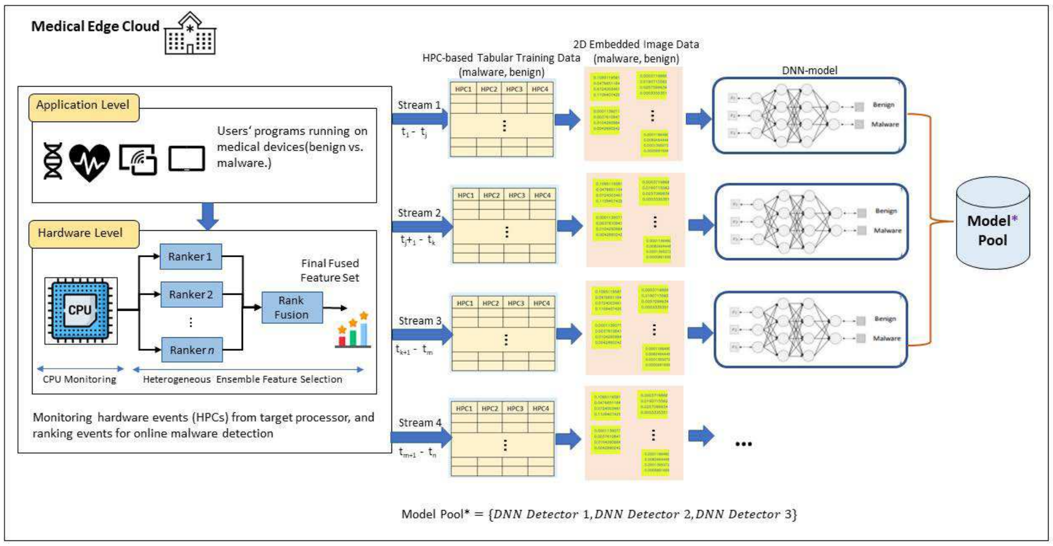
II. Proposed Methodology

This section presents the overview of the proposed deep learning and deep reinforcement learning-based approach for accurate image-based zero-day malware detection in IoMT devices. As depicted in Figure 1, first, the hardware events are monitored using an effective performance evaluation tool. Then, we analyze the collected HPC events to select the most prominent events addressing the issue of run-time malware detection using a limited number of HPC registers using a heterogeneous ensemble feature fusion method [25]. Functionality of different selection methods to determine the top HPC events are combined, and the final top four features are selected: L1-dcache-loads, node-stores, node-loads, and L1-dcache-stores. Next, we first employ a two-stage deep learning based approach introduced in [28] to embed tabular HPC data to image data and to leverage transfer learning techniques on ImageNet to boost up the detection accuracy for unknown zero-day malware detection.

DNN-based models have shown a higher detection performance over zero-day malware detection than classical ML [28]. However, IoMT demands a highly effective cybersecurity defender to all the stream data for different connected devices, which makes it a more challenging issue. This is because a minor miss-detection over a malicious activity could pose a critical negative impact on patient's medical records, diagnoses, and proper treatments, which could cause a life or death situation. To enhance the performance of the detection, we train various DNN-based models on each stream data to form a model pool with effective DNN models. To achieve this goal, as shown in Figure 2 we train a deep reinforcement learning-based agent called Advantage Actor Critic (A2C) [29] to dynamically select the best model defender at run-time on a new stream of data collected from the biomedical devices. The DRL-based agent is then deployed to automate the process of adaptively selecting a single best DNN defender at run-time according to the new hardware-related data characteristics.

A. Experimental Setup

IoMT devices typically operate on a continuously basis, in which the data from these devices are collected on a daily continuous basis and accumulated after a period of time for later analytic purposes. To study IoMT security, we simulate the IoMT operational environment in our experiments in which we split the whole time into four periods $(t_1, t_2, t_3, t_4, t_5)$. The data collected during t_1 and t_2 are stream 1 data, and the data collected during t_2 to t_3 are stream 2 data. First applications (benign and malware) are executed and profiled on a computer system equipped with an Intel Xeon processor



Medical Edge Cloud
Application Level
Users' programs running on medical devices(benign vs. malware.)
Hardware Level
CPU
Ranker 1
Ranker 2
Ranker n
Rank Fusion
Final Fused Feature Set
CPU Monitoring
Heterogeneous Ensemble Feature Selection
Monitoring hardware events (HPCs) from target processor, and ranking events for online malware detection
Stream 1
$t_1 - t_j$
Stream 2
$t_{j+1} - t_k$
Stream 3
$t_{k+1} - t_m$
Stream 4
$t_{m+1} - t_n$
HPC-based Tabular Training Data (malware, benign)
HPC1
HPC2
HPC3
HPC4
2D Embedded Image Data (malware, benign)
DNN-model
Benign
Malware
Model* Pool
Model Pool* = {DNN Detector 1, DNN Detector 2, DNN Detector 3}

Fig. 1: Overview of the proposed methodology, Part 1: Two-stage image-based malware detection using DNN models to create a pool of DNN-based detectors

that is widely used in biomedical applications in healthcare systems. HPC events are collected using *Perf* tool available under Linux at sampling time of 10ms. Also, applications are run in a Linux Container (LXC) [30] which provides access to actual hardware performance counters data instead of emulating HPCs. Malware applications, collected from and categorized by VirusShare and VirusTotal online repositories, comprises nine types of malware including Worm, Virus, Botnet, Ransomware, Spyware, Adware, Trojan, Rootkit, and Backdoor. In our experiments, each stream data are split into three sets: training, known test, and zero-day test. stream 1, 2, and 3 training data are for training and validation purposes for ML and DNN models, then models are tested on both known test and zero-day test datasets; stream 4 represents a new data stream. We use its training dataset of stream 4 to train the online adaptive decision maker (*DRL-HMD*), and we use its known and zero-day testing datasets to test all models including MLs, DNN models, and the *DRL-HMD* and report the inference result.

B. *Threat Model*

Data generated from IoMT devices in the edge cloud is different than data gathered and stored in the centralized cloud. IoMT devices generate data on an incremental basis, while centralized data are gathered together from all sources at all times into one pile of data. We can specify the generated data from a period as one batch of data, which we call a stream of data. Such a period can be one day, one week, or one month depending on how much data are needed for different use cases. For training the ML/DL/DRL models, we need to have some quantity of data, meanwhile, the model is updated at a reasonable frequency. We observe IoMT stream data has several characteristics: 1) the quantity of stream data is much less than centralized data, 2) according to recent studies, IoT/IoMT data are more likely to contain new variants of data and unknown malware types than previous stream data, due to the fast-growing IoT devices available to use, and the cybersecurity vulnerabilities of these devices, 3) different IoMT devices' HPC data may have similar or different feature spaces. In this paper, we mimic the medial edge cloud operations in which we have five IoMT devices that have similar feature distribution, and generated data stream in four time periods. From all of the data we have, we randomly divide it into five devices, and each device's data are further divided into four streams with the same ratio. We select one device to experiment with and report our experiment results.

To model the zero-day malware threat type in our experiments, among all nine malware types in each stream, we held out all four types of malware from rootkit, backdoor, virus, and ransomware as the target zero-day test data. These four types of malware are not presented in the training and known test datasets, thus, the zero-day malware set is unknown from the training dataset. For benign, we held out 30% of all benign data of each stream data aside as a zero-day test benign dataset. We kept both malware and benign aside to imitate the zero-day testing in real-world scenarios where the malware is undocumented in the training database of the detection mechanism. The rest of the five types of malware including trojan, spyware, botnet, worm, and adware as well as the rest of benign samples are considered for training and known test purposes for each stream, and we randomly split them into 70% for training and 30% for known testing. The difference between the known-test and zero-day-test in our experiments is that the known-test data contains the same malware types as the training dataset but with different unseen data and the zero-day-test data contains different malware types from the training dataset that are considered as new unknown attacks. After data are split, we relabel all types of malware as malware and leave benign as benign. Notably, our *DRL-HMD* framework uses the same datasets during training, known-test, and zero-day test same as all classifiers. Also, classical ML models use the tabular format data, and *DRL-HMD* employs the image data converted from corresponding HPC-based tabular data in training the DL-based model. In training/testing the DRL model, we use the known-test dataset at each stream to generate the experience replay dataset for training and use the zero-day test data to create the experience replay dataset to test the DRL agent. Note that across all the

ML, DL, and DRL models, the three datasets are consistent for each stream, but the data are randomly selected and each stream's data contain substantially different malware types.

C. Overview of DRL-HMD Framework

Our proposed framework, *DRL-HMD*, contains two parts: the first part is a two-stage deep learning-based malware detection method that produces a pool of highly effective DNN-based detectors. After that, an intelligent and salable deep reinforcement learning-based agent is trained to select the best DNN detector during run-time from the pool of detectors to defend against the newly generated unknown edge stream data. The general overview of the *DRL-HMD* is depicted in Figure 1 and Figure 2. As shown in Figure 1, during the first part of the *DRL-HMD*, tabular hardware events (HPCs) data (that are monitored from the underlying processor) are converted to image formatted data using an effective 2D embedding algorithm [28], [31]. Then, we investigate the state-of-art ResNet model architecture [32] and transfer learning over the data of ImageNet, which consists of 1,000 categories with a total of 1.3 million training images, 50,000 validation images, and 100,000 testing images. We use ResNet18 as a base architecture with customized last fully connected layer and the SoftMax function retrained the whole model over three HPC data streams generated from the IoMT device. The pre-trained model on ImageNet has already learned to recognize patterns from millions of generic images. While we feed our malware and benign embedded images, the transferred model applies the pattern recognition of images and continues learning the high-level feature difference between malware and benign. Use of transfer learning saves training time, boosts test accuracy over new biomedical device's stream data, and is more generalized towards the IoMT hardware features domain.

In part 2 of *DRL-HMD* as shown in Figure 2, a deep reinforcement learning agent called Advantage Actor Critic (A2C) [29] is trained to learn from a reward policy and adaptively select the best DNN model at run-time from the model pool for unknown malware detection based on the new data stream generated from the IoMT devices. Figure 3 depicts the architecture of the A2C model used in *DRL-HMD* and its agent interacting with the model pool and DRL environment. As the agent takes actions and moves through the environment, it learns to map the observed state of the environment to two possible outputs: 1) recommended action output from the Actor, which is a probability value for each DNN detector in the action space to maximize the reward from the state, where the reward is the mapping between state and the action according to the defined reward policy in the DRL environment at the current time step; 2) estimated reward produced from the Critic, which is a sum of all rewards expected to receive in the future state. Actor and Critic learn to perform their tasks, such that recommended actions from the Actor maximize the total rewards for all steps. We use two Multi-Layer Perceptron (MLP) in A2C, that the Critic contains five layers with three hidden layers (32*16*16), and the Actor is a four-layer MLP containing two hidden layers with 32 and 16 hidden nodes. The input layer contains the dimension of action space, and the output layer contains one action result. We use the Sparse Categorical Cross Entropy loss function with Adaptive Moment Estimation (Adam) optimization algorithms for the Actor and the Mean Squared Error loss function for the Critic.

D. DRL Environment

We customize the *DRL-HMD*'s environment based on OpenAI's Gym [33]. OpenAI Gym is an open-source interface that provides RL environments for researchers to develop and benchmark new algorithms. It also offers a structured interface for customizing the RL environment. Our DRL environment contains several important elements including the reward policy π, state space S, action space A, and the step function of mapping state to action according to the reward policy.

Algorithm 1 Process of Part 2 in *DRL-HMD*

```
Pre-process experience replay data for stream 4;            ▷ See Section II-D1
Initialize DRL agent parameters: discount factor $\gamma$, learning rate for Actor
and Critic, and update interval;
Initialize state S, reward policy $\pi$=F1*AUC;
Let $d \leftarrow$ selected malware detector;
Let $DNN \leftarrow$ DNN model;
while training of RL system do
    Agent reads state $s_t$;
    Actor predicts on state $s_t$, receives probabilities of all actions;
    Agent selects the action with the highest probability;
    Agent receives reward $r_t$ and new state $s_{t+1}$;
    Critic predicts future rewards based on current action;
    Agent computes advantage value to evaluate how good is the action;
    Compute Actor loss value;
    Computer Critic loss value;
    Compute and apply gradients to update Actor and Critic networks
    weights;
end
```

```
while Inference of RL system do
    for zero-day test data: $0 \rightarrow n$ do
        Import trained Actor model;
        Agent reads state $s_t$;
        Agent predict all actions' probabilities;
        Agent assigns the best $d_t$ to defend, gets $y_t^- = (a_t|s_t)$;
        if $y_t^-$ == $y_t$ true label then
            increment detection rate;
            record selected detector $d_t$;
        else
    end
    calculate DRL system detection metrics (F1, Accuracy, AUC, etc) for all
    test data.
end
```

Algorithm 1 shows the DRL-based pipeline for part 2 in *DRL-HMD* as depicted in Figure 2. State S consists of one set of experience replay data pre-processed as an output of the model pool and embedded images for training and testing the DRL agent. It contains each 2D image generated from each row of four HPCs data, the predictions (correct/incorrect predictions) from the DNN detectors, and each detector's detection metrics (F1, AUC). The state space (S) corresponds to the three ResNet-based detectors regarding their experience replay data. Action Space A is a set of three detectors available to the DRL agent. Each action corresponds to one of the three pre-trained ResNet-based malware detectors in the model pool. During the operation of IoMT, the number of action spaces can grow as many as per design according to the complexity of the IoMT environment. Various actions for each piece of data will result in different consequences for getting a reward or not, and how many rewards the agent can receive. After being trained, the DRL agent is inclined to select the detector d_t that receives the highest rewards among the action space during run-time.

Reward Policy π is a pre-defined rule that maps $s_t \xrightarrow{R} a_t$,

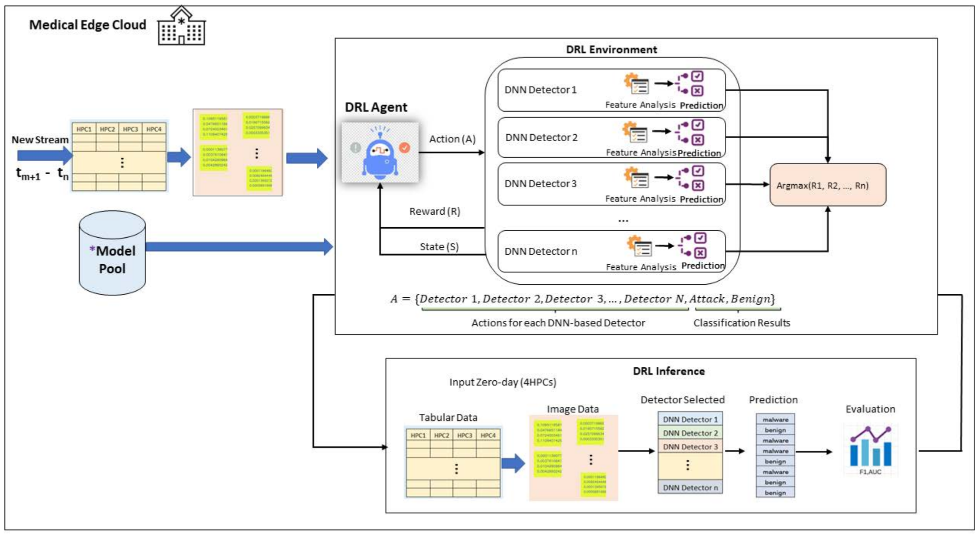


Fig. 2: Overview of the proposed methodology, Part 2:, Training a DRL Agent from the pool of DNN detectors and testing on new IoMT stream data. As seen, zero-day test HPC data comes in tabular format, and it is firstly embedded in a 2D image. The Model Pool* contains DNN models trained on various stream data from Part 1.

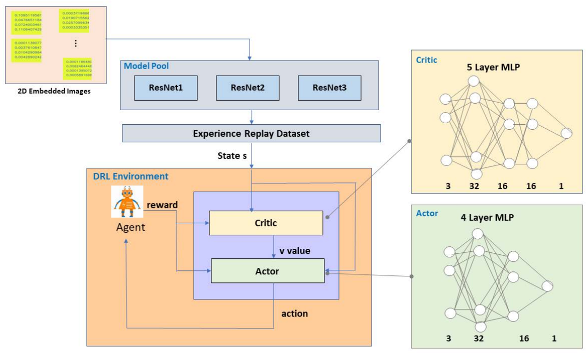


Fig. 3: A2C Architecture used in *DRL-HMD*. As pointed out on the right side of the graph, the Critic uses a 5-layer MLP, the Actor uses a 4-layer MLP.

where s_t is a state from experience replay data, a_t is the best action from the three detectors receiving the maximum reward according to the reward policy. For highly effective detection performance, we define the reward policy as $F1 * AUC$, to ensure that the DRL agent takes action on the most inclusive performing detector with the highest detection rate at each time step. The reward policy based on $F1 * AUC$ leads the agent selecting the detector with the highest F-measure and AUC among all correct-predicted models at time step t. In return, the agent receives the next state s_{t+1} and the reward r_t. The process continues until it reaches the terminal state. In our environment, we use a one-step update that each time step receives one piece of experience replay data rather than a batch of multiple data. We consider each piece of the data independent and the DRL agent obtains a reward that only directly affects the value of the current state-action pair. The value of other state action pairs is affected indirectly through the update of the rewards [29].

1) Experience Replay Data: To simulate the interaction between the DRL agent and environment, *DRL-HMD* uses an algorithm defined to collect all possible outcomes for different malware detectors as the experience replay data. The presented pipeline reads each state, goes through each classifier for each state, and records the model detection performance (F1, AUC). Then, we collect all the experience replay datasets from stream 4 for both known-test data (train RL) and zero-day data (test RL). As mentioned earlier, we consider three dataset partitions including train, known-test, and zero-day test. The training dataset is used to train and validate the DNN detectors. The known-test dataset is a dataset that is set aside during the initial dataset split and is used to train the DRL agent. In this setting, we train the DRL agent such that each branch of the DNN detector has not seen the data before so that the DRL agent can learn from the new dataset. Once the DRL agent is well-trained, the zero-day test data is used to examine the RL agent in an unseen DRL environment. This is because the zero-day test dataset contains new malware types and new benign data, and both malware and benign data are never seen by either the DNN detectors or the DRL agent.

E. Online Inference

We evaluated *DRL-HMD* on 1) the sum of rewards, and 2) the RL system's malware detection rate and its benchmark with all ML and DNN models. Figure 2 (right bottom) further shows a case study for the inference and evaluation process. As seen, first each row of the four zero-day test HPCs is embedded into a 2D image, then it runs through the three DNN-based detectors in different branches to perform predictions. Among all three models, any model which predicts the sampled HPCs correctly is eligible to receive a reward from the environment. If all DNN models predict wrongly, the system assigns the detector with the highest $F1 * AUC$ value with a small portion of a full reward to guide the DRL agent that if all detectors fail, the detector with the highest overall $F1 * AUC$ rate is the action to take. We assigned a 0.1 reward, which is much lower than the full reward of 1 when the prediction is correct. In many situations, multiple DNN models perform a correct prediction in which the DRL system grants the DNN model with the highest $F1 * AUC$ with receiving one reward, while other models receive zero rewards. The selected detector is then recorded one by one for all test data until the adaptive branching selection through the RL system is complete. Lastly, to evaluate the effectiveness of the DRL system, we use the recorded selection of malware detector at each time step to reproduce the defending process to obtain the detection result. We first load the DNN model selected by the DRL system at each time step and feed with the same row of four HPCs data to the model to run prediction. We record each prediction for all test data and calculate the DRL system's detection metrics.

III. Experimental Results and Evaluation

This section presents the experimental results and evaluation of the proposed intelligent malware detection method. We use F-Measure (F1-score) and Area Under the Curve (AUC) metrics to evaluate the performance of the models. F-Measure in ML is interpreted as a weighted average of the precision and recall. F-measure is a more comprehensive evaluation metric over accuracy (percentage of correctly classified samples) since it takes both the precision and the recall into consideration. In addition, AUC is another important evaluation metric for checking any ML model's performance at various threshold settings that demonstrates how effectively a classification model can differentiate between various classes.

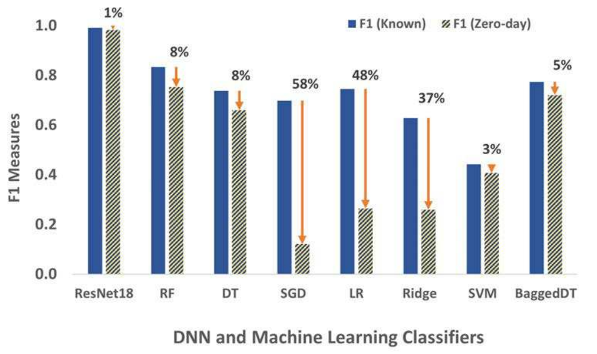


Fig. 4: F-Measure comparison of various ML-based HMD models for detecting known and unknown (zero-day) malware

1) All Models' Evaluation: Figure 4 shows the F-Measure results of DNN and various classical ML classifiers (used for HMD with 4 HPC features) for both known and unknown malware detection. As observed, the performance of all models drops when used for recognizing zero-day malware with a few algorithms with relatively lower drops. For example, DNN-based ResNet18 drops only 1%, Random Forest and Decision Tree drop 8%, and BaggedDT drops 5%. Other weaker ML models such as SGD, Logistic Regression, and Ridge Classifier drop significantly by more than 30% when examined by the unknown (zero-day) test data such that the trained machine learning classifiers have never seen the malware variants. This significant performance reduction highlights the challenge of detecting unknown malware in IoMT devices, and calls for an adaptive and generalized zero-day malware detection method during run-time to combat such growing cybersecurity threats.

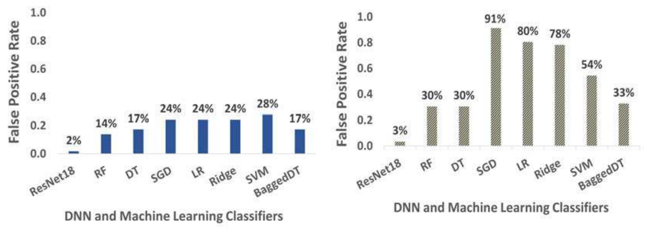


Fig. 5: False Positive Rate (FPR) between Known Test and Zero-day Test among DNN-based and ML-based Detectors. Left: FPR on Known Test, Right: FPR on Zero-day Test

To accurately detect malware, the defending model needs to have a lower false positive rate (FPR) which wrongly identifies benign applications as malware. Our study shows that a stronger detector tends to have a lower FPR, while a weaker detector shows higher FPR to wrongfully identify benign as malware. As shown in Figure 5, our proposed detector DNN-based ResNet18 can achieve 2% FPR in known malware detection task, however, its FPR only increases 1% when detecting unknown zero-day malware. The Random Forest-based detector has 14% FPR in known tests but it increases to 30% FPR when detecting zero-day malware. The weaker ML models like SGD, Logistic Regression, Ridge Classifier, and SVM have a high FPR even in known tests with 24%, 24%, 24%, and 28% respectively, and when used for detecting unknown zero-day malware, their FPR increases significantly at 91%, 80%, 78%, and 54%. It can be concluded that to detect malware accurately, the ability to differentiate complex malware and new variants is an important factor to measure the high performance and robustness of the detector.

2) DRL-HMD Evaluation: Table I reports the performance results of the proposed *DRL-HMD* benchmarked with ML and DNN-based malware detectors. We trained the ML-based detectors on stream 1 data, train three different DNN models on stream 1, stream 2, and stream 3 data respectively, and train the deep reinforcement learning-based A2C agent on stream 4 known test data. All of the models are tested on stream 4 unseen zero-day test dataset. Stream 4 data represents a new IoMT data stream newly generated from the biomedical computing device that is used to represent challenging zero-day data characteristics.

As the results indicate, our proposed *DRL-HMD* method outperforms a single best model of DNN detectors and traditional ML detector such as Random Forest and Decision Tree. *DRL-HMD* is the strongest classifier among all tested models and achieves F1-score and AUC of 98.6% and 99.35% on the unknown dataset, respectively, outperforming the regular best DNN-model by 5%, and RF classifier by 24% in F-measure. The proposed *DRL-HMD* method also offers only 0.01% false positive rate and 1% false negative rate for zero-day malware detection on randomly partitioned new stream data, which contains more challenging unknown malware variants. The proposed method enhances the system performance by 5% in F-measure as compared to any single best DNN model despite using a small number of hardware events that are captured at run-time by existing hardware performance counter registers.

In Figure 6 we illustrate the training speed in episode rewards (left) and the sum of rewards (right) for training the A2C agent used in *DRL-HMD* for the design criteria

TABLE I: Performance results of different ML-based, DNN-based detectors, and *DRL-HMD* for zero-day malware detection

Model	ACC	F1	AUC	FPR	FNR
DRL-HMD (This work)	0.99	0.99	0.99	0.0001	0.01
ResNet1	0.94	0.90	0.91	0.09	0.05
ResNet2	0.92	0.87	0.89	0.07	0.08
ResNet3	0.96	0.94	0.95	0.07	0.02
RandomForest	0.92	0.75	0.78	0.30	0.03
DecisionTree	0.88	0.66	0.69	0.30	0.08
SGD	0.79	0.12	0.22	0.91	0.07
Logistic Regression	0.82	0.26	0.37	0.80	0.06
RidgeClassifier	0.80	0.26	0.33	0.78	0.09
SVM	0.78	0.41	0.46	0.54	0.15
BaggedDT	0.91	0.72	0.75	0.33	0.04

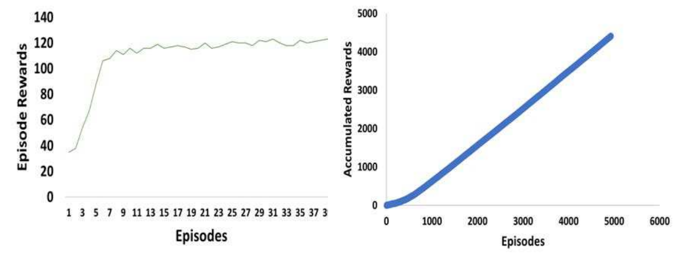


Fig. 6: Learning speed (left) and Sum of Rewards (right) of the Deep-RL agent in part 2 of *DRL-HMD*. Both analysis are based on the design criteria of $F1 * AUC$

(also called reward policy) of $F1 * AUC$. As seen, the agent gradually learns and reaches stable rewards after episode 7. As discussed before, reward policy guides the agent to learn to make decisions on its action at each step to achieve the goal of the most accumulated rewards in all training episodes. When the agent learns from its living environment (data and reward policy), its episode rewards tend to be stabilized and converged. The accumulated rewards grow steadily along the training process. Since the agent has learned from the training experience replay data, when it is time for online inference, the agent can apply the learned knowledge toward the reward policy to select the most optimized DNN model for the zero-day test malware detection to achieve the same goal of getting the most accumulated rewards along all test data. Since the reward policy reflects the most accurate detection performance, the proposed deep reinforcement learning-guided decision maker can increase the malware detection rate even on zero-day test data.

Furthermore, Table II shows the latency overhead of using the proposed deep reinforcement learning agent for online inference and adaptive malware detection. Since *DRL-HMD* selects the most optimal model from the model pool during run-time as the defender, the overall latency is around 0.117 seconds which includes the overhead for the DNN ResNet18 model to perform inference plus the overhead for the Deep-RL agent to complete the inference process. Moreover, adding the Deep-RL agent only adds an overhead of a total size of 49 KB (22 KB for the Actor, 27 KB for the Critic) on top of each ML-based detection model.

IV. Conclusion

In this work, we propose *DRL-HMD*, a hybrid intelligent image-based framework based on Deep Learning and Deep Reinforcement Learning (DRL) for securing IoMT devices against zero-day malware at the hardware level. The proposed

TABLE II: Overhead analysis

Phase	Tasks	Latency (s)	Total Latency (s)
Training	processing ERD*	0.009	0.227
	Train DRL agent	0.218	
Inference	processing ERD*	0.009	0.117
	DRL agent's online inference	0.108	

*ERD refers to Experience Replay Data, which is the processed data used in reinforcement learning environment for training and inference of Deep-RL agent.

AI-enabled detection method dynamically chooses the best DNN defender from a pool of highly effective models continuously trained on all stream data according to the current malware and benign data captured from IoMT devices and each model's performance as a response to the current data's characteristics. *DRL-HMD* first converts tabular hardware-based data (HPC events) into small-size images and then leverages a transfer learning technique to retrain and enhance the Deep Neural Network (DNN) based model's performance for unknown malware detection. Next, it trains and deploys a DRL agent to adaptively select the best DNN model for detecting unknown malware with high detection rate. The results indicate that our novel framework obtains a superior detection performance (99% in both F1-score and AUC) for recognizing unknown malware using a limited number of hardware events facilitating an accurate and adaptive zero-day malware detection at the processor hardware level.

V. Acknowledgment

This work is supported by the National Science Foundation under Award No. 2139034.

References

[1] "Market analysis report." [Online]. Available: https://www.grandviewresearch.com/industry-analysis/industrial-internet-of-things-iiot-market

[2] S. M. P. Dinakarrao *et al.*, "Cognitive and scalable technique for securing iot networks against malware epidemics," *IEEE Access*, vol. 8, pp. 138 508–138 528, 2020.

[3] M. Wazid *et al.*, "Iomt malware detection approaches: Analysis and research challenges," *IEEE access*, vol. 7, pp. 182 459–182 476, 2019.

[4] R. U. Rasool *et al.*, "Security and privacy of internet of medical things: A contemporary review in the age of surveillance, botnets, and adversarial ml," *Journal of Network and Computer Applications*, vol. 201, p. 103332, 2022.

[5] R. Kumar *et al.*, "A multimodal malware detection technique for android iot devices using various features," *IEEE Access*, vol. 7, pp. 64 411–64 430, 2019.

[6] J. Su *et al.*, "Lightweight classification of iot malware based on image recognition," 2018.

[7] H.-T. Nguyen *et al.*, "Iot botnet detection approach based on psi graph and dgcnn classifier," in *2018 IEEE International Conference on Information Communication and Signal Processing (ICICSP)*, 2018, pp. 118–122.

[8] P. Dinakarrao *et al.*, "Lightweight node-level malware detection and network-level malware confinement in iot networks," in *2019 Design, Automation & Test in Europe Conference Exhibition (DATE)*, 2019, pp. 776–781.

[9] A. Bettany and M. Halsey, "What is malware?" in *Windows Virus and Malware Troubleshooting*. Springer, 2017, pp. 1–8.

[10] H. Sayadi *et al.*, "Recent advancements in microarchitectural security: Review of machine learning countermeasures," in *MWSCAS'20*, 2020, pp. 949–952.

[11] Y. Ye *et al.*, “A survey on malware detection using data mining techniques,” *ACM Computing Surveys (CSUR)*, vol. 50, no. 3, pp. 1–40, 2017.

[12] J. Demme *et al.*, “On the feasibility of online malware detection with performance counters,” in *ISCA’13*. ACM, 2013, pp. 559–570.

[13] A. Tang *et al.*, “Unsupervised anomaly-based malware detection using hardware features,” in *RAID’14*. Springer, 2014, pp. 109–129.

[14] H. Sayadi *et al.*, “Ensemble learning for effective run-time hardware-based malware detection: A comprehensive analysis and classification,” in *Design Automation Conference (DAC’18)*, 2018, pp. 1–6.

[15] M. Ozsoy *et al.*, “Malware-aware processors: A framework for efficient online malware detection,” in *HPCA’15*, 2015, pp. 651–661.

[16] H. Sayadi *et al.*, “2smart: A two-stage machine learning-based approach for run-time specialized hardware-assisted malware detection,” in *DATE’19*, March 2019, pp. 728–733.

[17] H. Sayadi *et al.*, “Towards ai-enabled hardware security: Challenges and opportunities,” in *2022 IEEE 28th International Symposium on On-Line Testing and Robust System Design (IOLTS)*, 2022, pp. 1–10.

[18] H. Wang *et al.*, “Comprehensive evaluation of machine learning countermeasures for detecting microarchitectural side-channel attacks,” in *Proceedings of the 2020 on Great Lakes Symposium on VLSI*, 2020, pp. 181–186.

[19] H. Sayadi *et al.*, “Towards accurate run-time hardware-assisted stealthy malware detection: a lightweight, yet effective time series cnn-based approach,” *Cryptography*, vol. 5, no. 4, p. 28, 2021.

[20] “2021 sonicwall cyber threat report.” [Online]. Available: https://www.sonicwall.com/resources/white-papers/2021-sonicwall-cyber-threat-report/

[21] “Telehealth take-up: the risks and opportunities,” kaspersky Lab, kaspersky.com, Tech. Rep., 2021.

[22] Jill McKeon, “Outdated operating systems remain key medical device security challenge,” 2023. [Online]. Available: https://healthitsecurity.com/features/outdated-operating-systems-remain-key-medical-device-security-challenge

[23] L. Bilge and T. Dumitras, “Before we knew it: An empirical study of zero-day attacks in the real world,” in *CCS’12*. ACM, 2012, p. 833–844.

[24] Z. He *et al.*, “When machine learning meets hardware cybersecurity: Delving into accurate zero-day malware detection,” in *2021 22nd International Symposium on Quality Electronic Design (ISQED)*. IEEE, 2021, pp. 85–90.

[25] Z. He *et al.*, “Breakthrough to adaptive and cost-aware hardware-assisted zero-day malware detection: A reinforcement learning-based approach,” in *ICCD’22*, October 2022.

[26] “Intel healthcare technology overview for developers.” [Online]. Available: https://www.intel.com/content/www/us/en/embedded/healthcare/overview.html

[27] A. Healthcare, “Healthcare technology for improved insights and outcomes.” [Online]. Available: https://www.arm.com/solutions/healthcare

[28] Z. He *et al.*, “Deep neural network and transfer learning for accurate hardware-based zero-day malware detection,” in *Proceedings of the Great Lakes Symposium on VLSI 2022*, ser. GLSVLSI ’22. New York, NY, USA: Association for Computing Machinery, 2022, p. 27–32.

[29] V. Mnih *et al.*, “Asynchronous methods for deep reinforcement learning,” in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ser. ICML’16. JMLR.org, 2016, p. 1928–1937.

[30] M. Helsely, “Lxc: Linux container tools,” in *IBM developer works technical library*, 2009.

[31] B. Sun *et al.*, “Supertml: Two-dimensional word embedding for the precognition on structured tabular data,” 2019.

[32] K. He *et al.*, “Deep residual learning for image recognition,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[33] G. Brockman *et al.*, “Openai gym,” 2016.