IEEE ComSoc IEEE Communications Society | IEEE COMPUTER SOCIETY | IEEE Signal Processing Society | VTS | IEEE Transactions on Machine Learning in Communications and Networking

Received 18 September 2024; revised 12 March 2025; accepted 22 April 2025.
Date of publication 28 April 2025; date of current version 12 May 2025.

The associate editor coordinating the review of this article and approving it for publication was M. Jaber.

Digital Object Identifier 10.1109/TMLCN.2025.3564587

Evolving ML-Based Intrusion Detection: Cyber Threat Intelligence for Dynamic Model Updates

YING-DAR LIN[1] (Fellow, IEEE), YI-HSIN LU[1], REN-HUNG HWANG[2] (Senior Member, IEEE), YUAN-CHENG LAI[3], DIDIK SUDYANA[4], AND WEI-BIN LEE[5] (Senior Member, IEEE)

[1]Department of Computer Science, National Yang Ming Chiao Tung University, Hsinchu 300025, Taiwan
[2]College of Artificial Intelligence, National Yang Ming Chiao Tung University, Tainan 711010, Taiwan
[3]Department of Information Management, National Taiwan University of Science and Technology, Taipei 106335, Taiwan
[4]Computer and Network Center, National Cheng Kung University, Tainan 701401, Taiwan
[5]Hon Hai Research Institute, New Taipei 236401, Taiwan

CORRESPONDING AUTHOR: D. SUDYANA (didik@gs.ncku.edu.tw)

ABSTRACT Existing Intrusion Detection System (IDS) relies on pre-trained models that struggle to keep pace with the evolving nature of network threats, as they cannot detect new types of network attacks until updated. Cyber Threat Intelligence (CTI) is analyzed by professional teams and shared among organizations for collective defense. However, due to its diverse forms, existing research often only analyzes reports and extracts Indicators of Compromise (IoC) to create an IoC Database for configuring blocklists, a method that attackers can easily circumvent. Our study introduces a unified solution named Dynamic IDS with CTI Integrated (DICI), which focuses on enhancing IDS capabilities by integrating continuously updated CTI. This approach involves two key AI models: the first serves as the IDS Model, detecting network traffic, while the second, the CTI Transfer Model, analyzes and transforms CTI into actionable training data. The CTI Transfer Model continuously converts CTI information into training data for IDS, enabling dynamic model updates that improve and adapt to emerging threats dynamically. Our experimental results show that DICI significantly enhances detection capabilities. Integrating the IDS Model with CTI in DICI improved the F1 score by 9.29% compared to the system without CTI, allowing for more effective detection of complex threats such as port obfuscation and port hopping attacks. Furthermore, within the CTI Transfer Model, involving the ML method led to a 30.92% F1 score improvement over heuristic methods. These results confirm that continuously integrating CTI within DICI substantially boosts its ability to detect and respond to new types of cyber attacks.

INDEX TERMS Intrusion detection, cyber threat intelligence, sighting, machine learning, online learning.

I. INTRODUCTION

THE significance of Intrusion Detection Systems (IDS) has become crucial as network attacks continue to increase in frequency and sophistication. Traditional IDS relies on signature-based methods to detect attacks. However, these methods often fail to identify novel or evolving threats. Consequently, the need for Machine Learning-based Intrusion Detection Systems (ML-based IDS) has become apparent. ML-based IDS can outperform traditional IDS by learning to recognize patterns and anomalies in network traffic, which allows them to detect both known and unknown attacks more effectively [1].

Typically, ML-based IDS is an offline model that undergoes periodic updates to maintain its effectiveness against new threats. However, if a new attack occurs before the next update, the behavior deviates from previous patterns and becomes difficult to predict, posing challenges in establishing reliable behavioral norms. This raises concerns about the lifespan of learning models [2]. Consequently, ML-based IDS may exhibit defense gaps, rendering it unable to identify new types of network attacks promptly.

To address these challenges, integrating Cyber Threat Intelligence (CTI) with ML-based IDS is essential. CTI encompasses up-to-date information on cyber threats,

as analyzed by professional teams. In the field of CTI, 'sighting' refers to real-world network observations such as network flows and system logs [3]. When new network attacks occur, they are typically promptly identified by CTI, which also provides Indicators of Compromise (IoC) of the attackers. These IoCs are shared on the CTI platform, and if used to reinforce ML-based IDS, early prevention of attacks becomes feasible. Many research works focus on analyzing IoC from CTI reports to establish an IoC database [3], [4], [5] for blocklist purposes. However, relying solely on blocklists and IoC databases can be insufficient as attackers can circumvent these constraints, for example, by using proxy servers.

To address this need, our research introduces DICI (Dynamic IDS with CTI Integrated), a solution focused on integrating CTI to enhance ML-based IDS. DICI utilizes two ML models to achieve this integration. The first is an IDS pre-trained model designed to detect network traffic, while the second model, the CTI Transfer Model, analyzes CTI data to generate training datasets for the IDS Model. Given the complexity of CTI reports, which aggregate data from various sources, ML techniques within DICI are essential for analyzing and converting this information into effective training data.

During the inference phase, DICI integrates these two ML models to operate cohesively. The IDS Model is responsible for traffic detection, identifying whether traffic flows are benign or malicious. When it encounters outlier traffic–traffic that cannot be definitively classified–the system performs a CTI lookup using IoCs extracted from the outlier to retrieve a structured CTI report from the CTI platform. The CTI Transfer Model within DICI then analyzes this report and converts it into training data for the IDS Model, thereby enabling dynamic model updates and adaptation to new threats.

This approach differs from others that primarily focus on analyzing IoCs from CTI reports [6], [7], [8] to create IoC databases for configuring firewall blocklists. Instead, we utilize the CTI Transfer Model to generate targeted training data for the IDS Model, ensuring it continuously incorporates the latest threat information. This method transforms the IDS Model from a static, pre-trained system into a dynamic one that evolves in response to new threats through ongoing interaction with CTI.

In relation to existing research, this paper addresses key open challenges in ML-based IDS and CTI integration by making the following contributions:

- Traditional ML-based IDS often struggle with defense gaps due to their reliance on static models that fail to adapt to emerging threats. To address this, we propose DICI, an IDS framework that integrates real-time CTI updates to enhance threat detection capabilities. Unlike conventional ML-based IDS, which suffer from delayed model updates [2], our approach leverages continuously updated intelligence, improving adaptability and ensuring the detection of evolving malicious activities within network environments.
- Existing CTI utilization is typically limited to static IoC databases, which lack dynamic learning capabilities [3], [4], [5]. We introduce an online learning-based CTI Transfer Model, which processes structured CTI reports to enable adaptive updates in ML-based IDS. Unlike prior approaches, which rely on predefined threat indicators [6], [7], [8], our method automates the integration of sighting data with CTI, ensuring real-time intelligence processing and minimizing manual effort in updating IDS models.
- The lack of comparative analysis between rule-based CTI processing and ML-driven intelligence correlation remains an open research gap. Our study evaluates the benefits of ML in CTI processing by comparing an ML-based CTI Transfer Model against a traditional rule-based approach.

This study is guided by three key research questions to evaluate the effectiveness of integrating CTI into ML-based IDS and the role of the CTI Transfer Model in enhancing threat intelligence processing.

- **Does integrating CTI improve IDS performance?**
  To assess the impact of CTI, we compare an IDS with CTI integration against a standalone IDS without CTI as the baseline. Traditional IDS models operate independently and lack real-time intelligence updates, making them vulnerable to evolving threats. This evaluation determines whether CTI enhances detection capabilities and improves adaptability to emerging cyber threats.
- **How does the CTI Transfer Model compare to a traditional IoC database?**
  This question examines whether the CTI Transfer Model, which dynamically processes structured CTI reports, provides advantages over a static IoC database. The baseline for this evaluation is an IDS integrated with a precompiled IoC database. By leveraging real-time intelligence, we evaluate whether the CTI Transfer Model enhances IDS effectiveness compared to conventional IoC-based approaches.
- **Is an ML-based CTI Transfer Model better than a rule-based approach?**
  To assess the benefits of machine learning in CTI analysis, we compare an ML-based CTI Transfer Model against a rule-based approach, with the latter serving as the baseline. While rule-based systems rely on predefined conditions, ML techniques enable adaptive pattern recognition and correlation of complex threat indicators. This evaluation determines whether ML-based intelligence processing improves threat assessment, detection accuracy, and adaptability.

The work is organized into seven sections. Section II provides background information and related work on sighting and CTI integration. Section III formulates the problem, including notations and problem statements. Section IV outlines the proposed solutions and details the system architecture. Section V describes the system implementation.

TABLE 1. Survey on recent research in the CTI field.

Category	Papers	Year	Objective		CTI Approach					
			Automated Integration	Threat Discovery	Input		Supported By ML	Online Learning	New IoC Adaptability	Output
					Sighting-CTI Correlation	CTI Sources				
CTI Collection	[8]	2020	✓	-	-	Blog Vendor Report Official Report	BERT	-	-	Covert CTI Method
	[9]	2021	-	-	-	Dark Web	NLP	-	-	Covert CTI Method
	[10]	2022	✓	-	-	Social Media	CNN, NLP	-	-	CTI Extraction Framework
CTI Inference	[11]	2021	-	✓	-	-	-	-	-	Knowledge Assessment
	[12]	2022	✓	✓	-	Vender Report	NLP (NER)	-	-	CTI Knowledge Graph
CTI Sharing	[4]	2019	-	-	-	CTI Platform	-	-	-	CTI Standard
CTI Defense	[5]	2022	✓	-	-	CTI Platform	-	-	-	Attack Data Label
	[13]	2023	-	✓	-	OSINT	-	-	-	A Threat-Hunting Framework
	[3]	2023	-	-	-	CTI Platform	DBSCAN	-	-	CTI Augmentation Assessment
	Ours	2024	✓	✓	✓	CTI Platform	SVM, K-Means (IDS Models) K-means++ (CTI Models)	✓	✓	CTI for IDS Model

Section VI presents experimental results, comparing the performance of IDS with CTI. Finally, Section VII concludes the work and discusses future research directions.

II. BACKGROUND AND RELATED WORKS

A. INTRUSION DETECTION SYSTEM (IDS)

In the realm of cybersecurity, IDS plays a pivotal role in maintaining network integrity by monitoring hosts and network traffic to identify and counter security breaches. IDS can be categorized into misuse detection, which uses a signature-based approach with a low false alarm rate but a high missed alarm rate and only detects known attacks, and anomaly detection, which employs an ML approach with a low missed alarm rate but a high false alarm rate, capable of detecting both known and unknown attacks [1]. Enhancing the false alarm rate in ML-based methods could significantly improve detection performance.

Integrating ML into IDS has introduced various algorithms, such as Support Vector Machines (SVM), Naïve Bayes, Decision Trees, and Clustering techniques. The trend is towards ensemble or hybrid models, which combine the strengths of individual algorithms to develop more sophisticated and accurate intrusion detection capabilities. Research shows that these models generally outperform single algorithms [1]. Additionally, online learning, or incremental learning, has emerged as a strategy to adapt to the evolving nature of cyber threats. Despite challenges like data scarcity, our research aims to leverage real-time sightings and CTI data to enhance the efficacy and flexibility of IDS in addressing contemporary cybersecurity challenges.

B. CYBER THREAT INTELLIGENCE (CTI)

In modern cyber security, CTI is crucial, providing essential insights into emerging and existing cyber threats. CTI helps organizations predict, prevent, and respond to threats by encompassing key components: CTI Collection, Inference, Sharing, and Defending. CTI Collection gathers data from various sources such as threat feeds, open-source intelligence, internal logs, and dark web monitoring. CTI Inference processes this data to identify patterns, trends, and IoCs, turning raw data into actionable intelligence. CTI Sharing disseminates this intelligence within organizations and with external partners, enhancing collective knowledge. CTI Defending implements strategies based on these insights, such as updating security protocols, deploying new technologies, conducting training, and developing incident response plans, ensuring preparedness and swift response to incidents.

Despite significant research on CTI extraction and analysis, applying ML to interpret and reformulate these narratives into structured data remains underexplored. This transformation allows structured information to be shared among intelligence platforms, enabling IT security practitioners to investigate IoCs and tailor defensive strategies methodically. However, exploring ML's capacity to analyze CTI and its correlation with sightings data to reinforce IDS is still needed.

CTI platforms such as Threat Miner, GREYNOISE, and VirusTotal facilitate threat intelligence collection, inference, and sharing. These platforms gather CTI from multiple sources, provide results indicating attacker activities, and offer functionalities to help organizations combat cyber threats. They improve cybersecurity defenses and provide APIs for integration with defensive systems. Analyzing these platforms' functionalities, advantages, and applications in academic papers would enhance understanding of CTI's significance and how CTI platforms can bolster organizational cybersecurity.

Prior research has used ML techniques to analyze CTI and generate IoCs for IoC databases. However, simply creating

TABLE 2. Notations.

Dataset		
Sighting dataset	D_S	$D_S = D_S^R \cup D_S^T$
CTI dataset	D_C	$D_C = D_C^R \cup D_C^T$
Sighting training dataset	D_S^R	$D_S^R = \{(x_i^S, y_i^S) \mid i \in [1, RS]\}$; RS : size of sighting training dataset
CTI training dataset	D_C^R	$D_C^R = \{(x_j^C, y_j^C) \mid j \in [1, RC]\}$; RC : size of CTI training dataset
Sighting testing dataset	D_S^T	$D_S^T = \{(x_i^S, y_i^S) \mid i \in [1, TS]\}$; TS : size of sighting testing dataset
CTI testing dataset	D_C^T	$D_C^T = \{(x_j^C, y_j^C) \mid j \in [1, TC]\}$; TC : size of CTI testing dataset
Data input	x_i^S, x_j^C	x_i^S is the $i-$th sample in D_S, x_j^C is the $j-$th sample in D_C $x_i^S \in \mathbb{R}^{N_S}$, $x_j^C \in \mathbb{R}^{N_C}$, where N_S, N_C are the number of features
Indicator of compromise (IoC) in Sighting	O_h^S	IoC of sighting data; $h = \{1, 2, \ldots, \lvert O^S \rvert\}$; $\lvert O^S \rvert$: Number of sighting data in D_S
Indicator of compromise (IoC) in CTI	O_k^C	IoC of CTI data; $k = \{1, 2, \ldots, \lvert O^C \rvert\}$; $\lvert O^C \rvert$: Number of CTI data in D_C
Sighting label	y_i^S	$y_i^S \in \{0, 1, 2\}$, where 0, 1, 2 are benign, malicious, and outlier respectively
CTI label	y_j^C	$y_j^C \in \{0, 1\}$, where 0, 1 are benign, malicious respectively
Machine Learning		
Sighting Classifier	M_t^S	$M_t^S, t = 0, 1$, 0: supervised classifier, 1: unsupervised classifier
IDS Model	M^S	M^S is the IDS model consisting of M_0^S and M_1^S, where $M^S \bigcup_t M_t^S$
CTI Transfer Model	M^C	-
ML Model	M_n	$M_n = ML_n(D^R)$, where n indexes distinct ML models derived from different algorithms
ML Algorithm	ML_n	The n-th machine learning algorithm used, where $0 \leq n < \lvert ML \rvert$, $\lvert ML \rvert$: Total number of algorithms

blocklists of domains and IP addresses is insufficient for identifying all cyber threats. This limitation highlights the need for advancing defensive strategies and exploring alternative approaches beyond relying solely on IoC databases. Notably, no existing study has explored the advantage of continuously training an ML-based IDS using the IoC database, which promises continuous enhancements in IDS efficacy.

C. RELATED WORKS

The field of CTI has been the focus of extensive research aimed at fortifying defense mechanisms against cyber threats. Recent literature has often segmented studies into four primary categories: CTI Collection, CTI Inference, CTI Sharing, and CTI Defense. These categories represent key aspects of CTI, each addressing a distinct component of understanding, analyzing, and counteracting cyber threats. Table 1 compares the methodologies and objectives of these works across several dimensions, such as automated integration, threat discovery, sighting-CTI correlation, online learning capability, and adaptability to new IoCs. Automated integration refers to the ability to seamlessly incorporate CTI into a system, reducing manual intervention and enhancing real-time threat intelligence processing. Threat discovery highlights whether a solution can identify previously unknown attacks. Sighting-CTI correlation evaluates the ability to combine internal network observations (sightings) with external threat intelligence for more accurate detection. Online learning indicates the solution's capacity to continuously adapt by learning from new data, while new IoC adaptability measures how well the system adjusts to newly discovered attack patterns and indicators.

In the area of CTI Collection, researchers such as [8], [9], and [10] focused on gathering CTI from various sources, including blogs, social media platforms, and the dark web. These studies used ML techniques, such as NLP and CNN, to transform unstructured CTI into structured data, making it more useful for cybersecurity applications. However, these works primarily dealt with static data sources, without incorporating dynamic IoC updates or real-time adaptability. For CTI Inference, [11] and [12] developed methods to analyze collected CTI data. Reference [11] focused on knowledge assessment by analyzing red team tactics, while [12] built a CTI knowledge graph using Named Entity Recognition (NER) to map relationships between threats. Though these methods provided useful insights into potential attack trajectories, they did not leverage real-time sighting data or support continuous learning through online methods.

In CTI Sharing, [4] explored ways to enhance the efficiency of CTI exchange among organizations. Their research focused on standardizing the sharing process to enable more effective collaboration across cybersecurity teams. However, their approach did not involve real-time data or adaptability to new IoCs, limiting its capacity to address rapidly evolving threats. In CTI Defense, [3], [5], and [13] worked on applying CTI to strengthen defense mechanisms. Reference [5] developed a framework for generating intrusion detection datasets using CTI, while [13] proposed a threat-hunting framework to detect unknown threats. Reference [3] worked on augmenting CTI with real-time assessments to improve defense strategies. Although these works made significant contributions, they relied on traditional data sets and lacked the flexibility of continuous learning and adaptability.

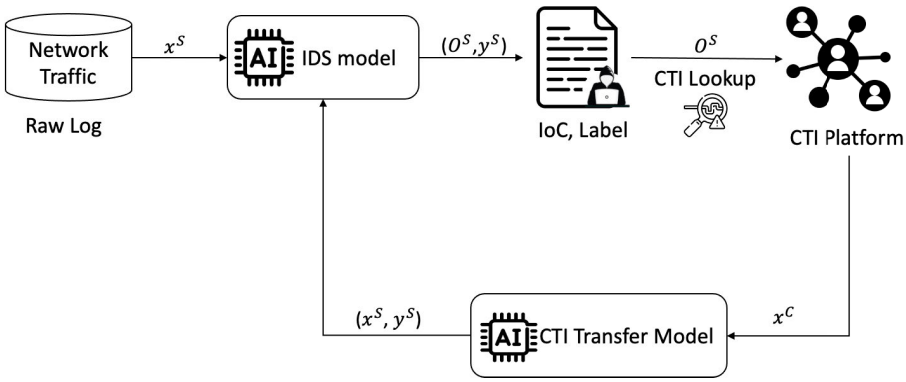


FIGURE 1. System architecture.

Our proposed solution, DICI, addresses the limitations observed in prior research by introducing a system that supports continuous online learning and real-time adaptability to new IoCs. Unlike previous approaches that rely on static IoC databases or pre-trained models, DICI integrates both sighting data and real-time CTI, enabling the IDS system to dynamically update and refine its detection capabilities. This ensures that DICI can not only recognize known threats but also adapt to new and emerging attack patterns. By utilizing multiple CTI sources, analyzing them with the ML model, and continuously adapting to new IoCs, DICI offers a more flexible and comprehensive defense mechanism than traditional CTI-based systems, which often struggle to keep pace with evolving cyber threats. Through its integration of advanced ML techniques and real-time updates, DICI sets itself apart as a robust solution capable of addressing both current and emerging cybersecurity challenges.

III. PROBLEM FORMULATION

In this section, the problem formulation is divided into two parts: firstly, the notations for defining the problem as a consistent symbol set, and secondly, the problem statements, which utilize these notations to articulate the issues at hand.

A. NOTATIONS

These notations delineate two categories, as referenced in Table 2. The first category is a dataset, which represents the sighting and CTI datasets. The sighting dataset comprises the actual collection of network traffic flow by a threat intelligence firewall, providing labels accordingly. These traffic flows are segmented into subsets designated for training and testing. To correlate with cyber threat intelligence, each source IP address associated with a traffic flow is considered an IoC. This facilitates the association of network traffic with potential threats and aids in identifying and mitigating cyber risks.

Furthermore, the CTI dataset utilizes IP addresses as IoCs acquired from sightings for CTI lookup within the CTI Platform, thereby obtaining structured CTI reports. These reports are similarly partitioned into subsets for training and testing. Additionally, since it is necessary to correlate with Sightings, the IP addresses within the CTI dataset are also designated as IoCs. This ensures a comprehensive alignment between CTI and Sightings, enhancing the accuracy and effectiveness of threat intelligence analysis.

The last category is ML. This category primarily defines two models: one tailored for the IDS Model and another for the CTI Transfer Model. Each model includes its own optimized ML algorithms and configurations. These models will be instrumental in elucidating the issues and formulating effective solutions in the following sections.

B. PROBLEM OVERVIEW

The objective of this study is to develop an evolving ML-based IDS that automates CTI integration and enhances threat detection through continuous learning. Figure 1 illustrates the system architecture, where network traffic, denoted as x^S, is processed by the IDS Model to classify activities as benign, malicious, or outlier traffic that cannot be confidently categorized.

For outlier traffic, the system performs a CTI lookup, retrieving structured CTI reports from a CTI platform. The CTI Transfer Model, denoted as M^C, analyzes the retrieved data, extracts relevant features, and determines whether the IoCs O^C are associated with known threats. This refined intelligence is then incorporated into the IDS training dataset, enabling real-time model updates.

This feedback loop ensures that the IDS Model continuously learns from newly observed threats, improving detection accuracy and adaptability. The approach focuses on optimizing the integration between the IDS Model and the CTI Transfer Model, analyzing them as distinct components to identify the most effective learning strategy. The performance of the integrated system is evaluated using the F1 score, ensuring a balance between precision and recall.

C. PROBLEM STATEMENT

The goal of this study is to develop the IDS Model M^S with a CTI Transfer Model M^C to maximize detection accuracy while efficiently utilizing CTI resources. Specifically, the

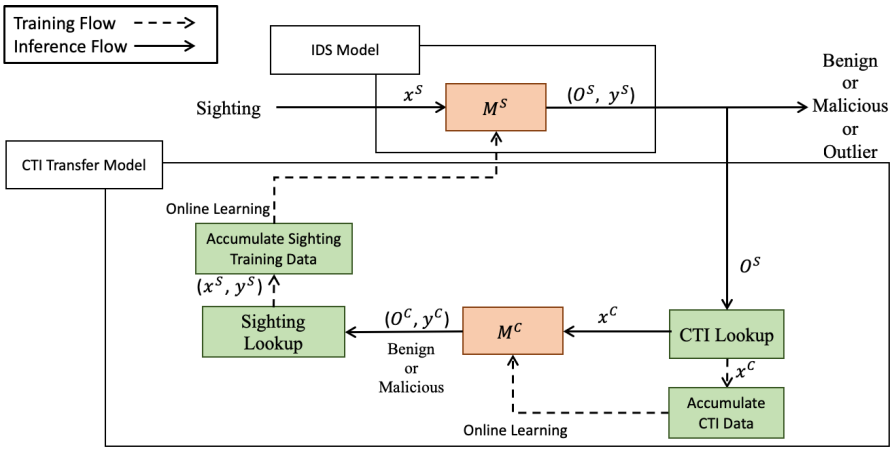


FIGURE 2. Solution overview.

objective is to improve the F1 score of M^S, which consists of supervised M_0^S and unsupervised M_1^S classifiers, using both sighting data D_S and structured CTI reports D_C, while maintaining efficiency under resource constraints for CTI queries. The formal problem definition is as follows:

- Input
  - Sighting dataset: D_S
  - CTI dataset: D_C
  - ML algorithms: ML_n
- Output
  - Optimized IDS Model $M^{S'}$ and CTI Transfer Model M^C
- Objective
  - Maximize F1 score on $M^{S'}$, incorporating sighting data D_S^T and CTI data D_C^T.
- Constraints
  - Limited CTI query resources

The CTI Transfer Model serves as the key component for integrating external threat intelligence into the IDS, enhancing its detection capabilities. By optimizing both models, the goal is to create an IDS that is adaptable, scalable, and capable of continuously learning from real-time intelligence, ensuring proactive and effective intrusion detection.

IV. DICI: DYNAMIC IDS WITH CTI INTEGRATED SOLUTION

This section is divided into two main parts to thoroughly explain our proposed solution. First, we provide an overview of the DICI solution, including its key components: the IDS Model and the CTI Transfer Model. Then, we delve into the specifics of the CTI Transfer Model, detailing its functions and how it integrates with the overall system.

We have also included multiple figures to illustrate the proposed solution. Figure 1 provides a system overview, showing the integration of the IDS Model with the CTI Transfer Model. Figure 2 details the inference and training process, while Figure 3 breaks down how threat intelligence is processed.

A. SOLUTION OVERVIEW

Our proposed solution, DICI, enhances the IDS Model by integrating the CTI Transfer Model, enabling continuous online learning. This integration allows the IDS to dynamically gather, process, and utilize CTI-based training data, ensuring that the model adapts to emerging threats in real-time.

Figure 2 illustrates the overall solution overview. The IDS Model, denoted as M^S, consists of two components, M_0^S and M_1^S, which classify network traffic flows x^S as benign, malicious, or outliers–where outliers represent traffic that cannot be confidently classified. Section V provides further details on the IDS Model.

When the IDS Model encounters outlier traffic, it extracts IoCs O^S from these flows and performs a CTI Lookup on a CTI platform to retrieve structured threat intelligence reports x^C. The CTI Transfer Model, denoted as M^C, is responsible for processing this data. First, M^C analyzes and extracts relevant features from the CTI reports to determine whether the IoCs O^C are benign or malicious, improving the IDS Model's detection accuracy.

Next, M^C performs a sighting lookup, referencing original network traffic records associated with O^C to validate the information provided by the CTI report x^C. This correlation enhances the accuracy and reliability of the training data.

Finally, since raw CTI report data is not directly suitable for IDS training, the inference results y^C from M^C undergo further processing through sighting lookups. This step ensures that the IDS Model is updated with validated and meaningful threat intelligence, allowing it to continuously improve and adapt to evolving attack patterns.

B. CTI TRANSFER MODEL

The CTI Transfer Model is a critical component in the DICI solution because it addresses the limitations of traditional IoC databases, which may miss new or evolving IoCs from

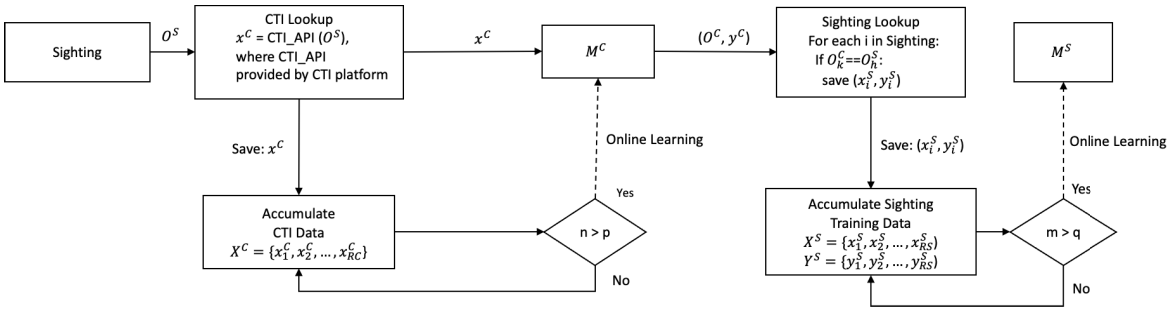


FIGURE 3. **Detailed procedure of the CTI transfer model.**

attackers due to their reliance on static data. By integrating sighting information with real-time CTI data, the CTI Transfer Model enables the IDS Model to detect recurring or new forms of attacks, ensuring that the system remains effective in a constantly changing threat landscape.

The CTI Transfer Model plays a critical role in processing structured CTI reports. Some CTI platforms that can provide such reports include VirusTotal, ThreatMiner, and GreyNoise. These platforms aggregate threat intelligence from multiple cybersecurity vendors and public feeds, providing data across multiple timelines and sources. However, in our experiments, we use only VirusTotal due to its broader threat intelligence coverage, higher API query limits, and ability to provide structured reports aligned with our CTI processing requirements. The detailed justification for this selection is provided in Section V-B.

While these reports offer valuable insights, they often come with significant challenges. Many cybersecurity vendors provide an initial assessment of an IP's threat status based on a specific incident, but this status may not be updated if the IP becomes benign at a later time. This creates inconsistencies and potential gaps in threat intelligence over time.

The complexity and volume of structured CTI reports make ML essential for processing and analysis in the CTI Transfer Model. ML automates the extraction of meaningful patterns from diverse data sources, handling variations in report structures and formats that would be difficult to process with traditional heuristic methods. By leveraging both historical data and real-time inputs, ML enhances the system's adaptability to new and evolving threats.

Unlike rule-based approaches that rely on predefined conditions, ML enables more flexible and nuanced analysis. Instead of being restricted to fixed sets of features, the ML model learns from multiple data points across different sources and timeframes, improving the accuracy of threat assessments. This capability is especially critical for identifying attack vectors that evolve over time, allowing the system to refine its understanding dynamically rather than depending on static rule sets.

Additionally, ML strengthens the CTI Transfer Model's ability to correlate various threat indicators. By continuously learning from new data, the model can uncover relationships between seemingly unrelated IoCs, detecting patterns that signal emerging threats. This real-time intelligence processing enhances the IDS's ability to respond proactively, ensuring that the system remains effective in an ever-changing cybersecurity landscape.

The detailed procedure of the CTI Transfer Model is illustrated in Figure 3. The process begins with the sighting phase, where the IDS Model detects outlier traffic that cannot be clearly classified. From this outlier traffic, IoCs O^S are extracted. These IoCs are used to perform a CTI Lookup via an API on a CTI platform, retrieving structured CTI reports x^C.

Once these reports are obtained, they are accumulated as CTI data $X^C = \{x_1^C, x_1^C, \ldots, x_{RC}^C\}$ within the CTI Transfer Model M^C. This model uses machine learning to analyze the structured CTI reports, determining if the IoCs O^C within these reports are benign or malicious. The machine learning component is crucial because it allows for the nuanced analysis of complex data sets that include varied IoC characteristics and threat indicators from multiple CTI sources.

When the number of accumulated CTI reports n exceeds a predetermined threshold p, the model initiates online learning to refine its detection algorithms based on the most current data. As new CTI reports are analyzed, each IoC O^C within a report undergoes inference to produce an inference result y^C. These IoCs O^C are also used for a sighting lookup to retrieve relevant sighting data x^S from the original network traffic records, categorizing this data as y^S to indicate whether it is benign or malicious.

The training data generated from this analysis, consisting of sighting data x^S and the CTI Transfer Model inference results y^S, is then accumulated as sighting training data $X^S = \{x_1^S, x_2^S, \ldots, x_{RS}^S\}$. Exceeding another threshold q with this training data triggers online learning for the IDS Model M^S, allowing it to continuously update and enhance its capabilities.

Through this online learning process, we enable the IDS Model to adapt continuously by utilizing real-time threat intelligence and the inference results from the CTI Transfer Model. This method not only enhances the detection capabilities of the IDS system but also ensures it remains dynamic and effective in a rapidly evolving threat landscape.

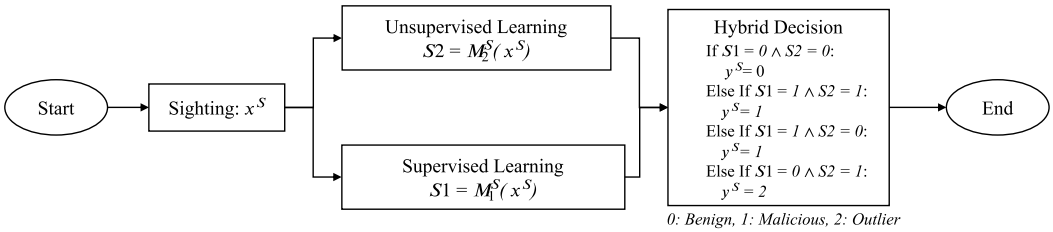


FIGURE 4. IDS Model with hybrid approach.

V. IMPLEMENTATION

This section is organized into three main sections to detail the implementation of our work. Firstly, we explain the hybrid ML approach in the IDS Model. Secondly, we introduce the CTI API. Thirdly, we discuss the open-source tools and libraries, focusing on data processing tools and ML libraries selected for their extensive support and alignment with our research objectives. Lastly, we present detailed information on the datasets utilized.

A. IDS MODEL

This approach aims to achieve the highest F1 score for the IDS Model, considering the complexity and diversity of the traffic flow. Supervised learning offers the advantage of high precision but lacks adaptability; thus, it may fail to recognize attack patterns with rich variations. On the other hand, unsupervised learning performs exceptionally well in adaptability but tends to have higher false positives. Moreover, according to [1], ensemble and hybrid classifiers tend to outperform single classifiers. By combining these two types of ML algorithms, we can create a more robust model. Thus, in this work, we integrate supervised and unsupervised learning into a hybrid model, leveraging the strengths of both methods to enhance performance.

Figure 4 illustrates the proposed IDS Model with a hybrid algorithm approach. In this proposed solution, called a hybrid model, there are two types of ML models for prediction: supervised model M_1^S, which produces results denoted as $S1$, and unsupervised model M_2^S, which produce results denoted as $S2$. Subsequently, a hybrid decision is employed to amalgamate decisions from both types of models, predicting the final output of y^S.

Based on Figure 4, the hybrid model operates on the consensus of its constituent supervised and unsupervised models. If both models classify traffic flow as malicious, the hybrid model similarly labels it as malicious. Likewise, if both models agree the traffic is benign, the hybrid system classifies it as benign, reflecting the consistency in their assessments. This design leverages the equivalent inference results of the two machine learning approaches for clear-cut decisions.

However, discrepancies between the models trigger a different protocol. If the supervised model identifies traffic as malicious but the unsupervised model does not, the hybrid model still deems the traffic malicious, valuing the higher reliability typically associated with supervised learning. Conversely, if the unsupervised model alone flags traffic as malicious while the supervised model does not, the hybrid system categorizes the traffic as an outlier. This approach acknowledges that while the unsupervised model may have a higher rate of false positives, its detection signals potential ambiguity in the traffic's nature, suggesting it may neither be clearly benign nor malicious. Such outlier traffic flows are earmarked for further analysis via the CTI Transfer Model, accommodating the unsupervised model's greater adaptability without prematurely labeling the traffic.

In ML-based IDS, hybrid models have been shown to be more effective than single classifiers [1]. Selecting appropriate supervised and unsupervised learning algorithms within the hybrid framework is essential for achieving optimal performance.

For the supervised component, Support Vector Machines (SVM) has been chosen due to its effectiveness in handling high-dimensional data and classification tasks. While tree-based models have demonstrated strong performance in IDS [14], [15], they come with limitations such as node depth constraints and the need for pruning to prevent overfitting, particularly in online learning environments. SVM, on the other hand, provides a more efficient and robust approach by defining optimal decision boundaries while avoiding excessive complexity [16]. These characteristics make SVM well-suited for online learning in IDS applications.

For the unsupervised component, K-means clustering is widely used for anomaly detection in network traffic [17], [18]. K-means is particularly beneficial in online learning as it continuously updates cluster centers, allowing it to adapt to evolving traffic patterns in real time. Its ability to self-adjust based on new data distributions makes it an effective choice for enhancing the adaptability of ML-based IDS.

By combining SVM for precise classification and K-means for dynamic anomaly detection, the hybrid IDS model effectively balances detection accuracy and adaptability, making it a robust approach for identifying both known and emerging threats.

Furthermore, to prevent outliers from being misclassified as inliers, detected outliers undergo a CTI lookup to verify their threat status before being incorporated into the model. The CTI Transfer Model processes structured intelligence reports to ensure that only validated threats contribute to training, reducing the risk of incorrect classifications. Additionally, the hybrid IDS model mitigates overfitting,

47.241.[redacted]"{
""data"": {
""attributes"": {
""network"": ""47.240.0.0/14"",
""tags"": [],
""whois"": ""inetnum: 47.0.0.0 - 47.255.255.255\nnetname: IANA-NETBLOCK-47\ndescr: This network r
ange is not allocated to APNIC.\ndescr:\ndescr: If your whois search has returned this message, then you have
\ndescr: searched the APNIC whois database for an address that is\ndescr: allcated by another Regional Inter
net Registry (RIR).\ndescr:\ndescr: Please search the other RIRs at whois.arin.net or whois.ripe.net\ndescr:
for more information about that range.\ncountry: AU\nadmin-c: IANA1-AP\ntech-c: IANA1-AP\nremarks: For genera
l info on spam complaints email spam@apnic.net.\nremarks: For general info on hacking & abuse complaints emai
l abuse@apnic.net.\nmnt-by: MAINT-APNIC-AP\nmnt-lower: MAINT-APNIC-AP\nstatus: ALLOCATED PORTABLE\nlast-modif
ied: 2008-09-04T06:51:28Z\nsource: APNIC\nrole: Internet Assigned Numbers Authority\naddress: see http://www.
iana.org.\nadmin-c: IANA1-AP\ntech-c: IANA1-AP\nnic-hdl: IANA1-AP\nremarks: For more information on IANA serv
ices\nremarks: go to IANA web site at http://www.iana.org.\nmnt-by: MAINT-APNIC-AP\nlast-modified: 2018-06-22
T22:34:30Z\nsource: APNIC\n"",
""last_analysis_date"": 1696508108,
""as_owner"": ""[redacted]."",
""last_analysis_stats"": {
""harmless"": 63,
""malicious"": 3,
""suspicious"": 0,
""undetected"": 23,
""timeout"": 0
},

FIGURE 5. Example of a structured CTI report. Sensitive information has been obscured for privacy reasons.

as the combination of SVM and K-means clustering prevents the system from becoming too specialized in detecting only certain attack patterns. While SVM effectively handles high-dimensional traffic data, K-means clustering enhances adaptability by identifying new attack behaviors without relying solely on predefined labels. This hybrid approach also helps counteract bias by continuously integrating real-time CTI updates, ensuring the model does not become overly focused on specific attack types but remains responsive to evolving threats.

B. CTI API

The CTI API facilitates automated threat intelligence lookups by retrieving structured reports from CTI platforms. Our study evaluates various CTI platforms to select the most comprehensive source for integration. Table 3 compares several platforms based on their capabilities, supported IoC types, and API query limits, including Shodan [19], MalwareBazaar [20], Threat Miner [21], GREYNOISE [22], and VirusTotal [23].

Each platform has distinct functionalities. Shodan [19] specializes in querying internet-connected devices, making it more suitable for reconnaissance tasks rather than direct CTI correlation. MalwareBazaar [20] only supports hash-based queries, limiting its usefulness for network-based threat detection. While Threat Miner [21] and GREYNOISE [22] support IP lookups, their restrictive API query limits reduce their practicality for large-scale correlation analysis. VirusTotal [23], on the other hand, provides broader capabilities with higher API limits, making it the most suitable platform for our CTI lookups.

VirusTotal [23] offers an extensive repository of threat intelligence, integrating multiple security tools, including antivirus engines and file analysis tools, within a single interface. With real-time updates and a large global cybersecurity community, VirusTotal provides valuable insights into emerging threats, vulnerabilities, and attack patterns, allowing organizations to strengthen their cybersecurity defenses.

The CTI API from VirusTotal enables seamless integration into security infrastructures through RESTful HTTP requests, such as the GET method. Figure 5 presents an example of a structured CTI report, containing key information such as Whois details, last analysis date, ownership data, and aggregated security assessments from multiple sources. In this study, we utilize IP addresses as input to acquire structured CTI reports, extract relevant features, and analyze critical threat information to enhance IDS detection accuracy.

C. OPEN-SOURCE TOOLS AND LIBRARIES

Table 4 details the open-source tools and libraries utilized in this work. Nfdump [24] played a crucial role in capturing NetFlow records directly from the router and processing the data for subsequent analysis. This method facilitated efficient collection and in-depth examination of NetFlow data, allowing us to gain a deeper understanding of network traffic patterns and behaviors. Additionally, we used Imbalanced-learn and Scikit-learn to develop the ML models, and Seaborn and Matplotlib to visualize the data.

D. DATASET

In this section, we introduce the datasets utilized in this study. Our research leverages two primary datasets: the Sighting Dataset D_0, which captures actual network traffic in a real network as sighting, and the CTI Dataset D_1, designed to address contemporary IoCs. Each dataset serves a specific purpose and is integral to our experimental framework.

1) Sighting Dataset

The sighting dataset is used to train the IDS models by incorporating real-world IoCs obtained through CTI lookups. However, publicly available intrusion detection datasets often exclude IP information due to privacy concerns, making them unsuitable for CTI integration. While some public datasets, such as CIC-IDS-2017, do contain IP information within the PCAP files, they represent historical snapshots of network traffic rather than real-time threat intelligence. Since CTI platforms continuously update their databases with the latest indicators of compromise (IoCs), querying a CTI platform with IPs from an outdated dataset may not yield meaningful intelligence. Many of these IPs may no longer be relevant or lack available information in CTI platforms because they originate from local IP addresses within controlled testbed environments. This temporal limitation makes it impractical to effectively integrate CTI with public datasets for real-time threat intelligence evaluation. To address this, we use a proprietary threat intelligence firewall to collect NetFlow traffic data over two periods: January 6, 2023, to June 31, 2023, and October 1, 2023, to October 31, 2023, totaling eight months. The dataset, processed with Nfdump, contains 2,774,241 traffic flows with a total size of 137.3 MB. The firewall's built-in threat intelligence mechanisms label traffic as benign or malicious, with network traffic features summarized in Table 5.

The dataset underwent comprehensive preprocessing to ensure quality and suitability for analysis. Missing values were handled through a combination of imputation and removal of records exceeding a predefined threshold.

TABLE 3. Threat intelligence platform comparison.

Platform	Type	IoC Lookup	API Query Limit
Shodan [19]	Search Engine for Internet-Connected Devices	IP, URL, Domain	500 queries/day (Academic)
MalwareBazaar [20]	Malware Samples and Analysis	Hash	2,000 queries/day
Threat Miner [21]	Threat Intelligence Aggregator	IP, URL, Domain, Hash	10 queries/minute
GREYNOISE [22]	Threat Intelligence Aggregator	IP	50 queries/week
VirusTotal [23]	Threat Intelligence Aggregator	IP, URL, Domain, Hash	20,000 queries/day (Academic)

TABLE 4. Open source tool and library.

Category	Name	Functionality
Tool	Nfdump [24]	NetFlow record.
Library	Imbalanced-learn [25]	Data imbalancing support.
	Scikit-learn [26]	Machine learning library.
	Seaborn [27]	Data visualization support.
	Matplotlib [28]	Data visualization support.

To prevent dependency on specific network identifiers, metadata fields such as time_start, time_end, src_ip, dest_ip, and src_port were removed. Categorical variables were encoded using one-hot encoding for seamless integration into ML models, while feature standardization ensured consistent scaling, improving model performance and interpretability. Class imbalance was addressed using undersampling techniques to reduce bias in model training. These preprocessing steps were carefully implemented to enhance the reliability and robustness of the dataset.

The sighting dataset provides a detailed view of network traffic, making it valuable for intrusion detection and security analysis. Retaining essential traffic characteristics allows for meaningful correlation between network activity and IoCs, enabling more effective CTI integration. The dataset supports advanced intrusion detection by facilitating nuanced behavioral analysis and identifying emerging security threats.

2) CTI Dataset

The CTI dataset ensures the reproducibility of our experiments by incorporating structured CTI reports downloaded locally, as CTI information undergoes continual changes over time. We utilized the VirusTotal CTI API to perform lookups on all IPs in the sighting dataset, storing a total of 2,112 structured CTI reports with a file size of 48.4 MB. By using locally stored CTI reports, the dataset provides consistent results over multiple experiments, ensuring that repeated CTI lookups yield stable and reproducible evaluations. The CTI dataset includes all extracted properties from structured CTI reports as features. During training, the model assigns higher weights to important features while reducing the influence of less relevant ones. The extracted CTI features include network information, last analysis date, ownership details, security vendor assessments, reputation scores, country, and total votes for malicious or harmless classifications, amounting to a total of 105 features. While IP addresses were initially used to map CTI records to their corresponding labels, they were excluded from the ML model's training process, ensuring that the model learns meaningful behavioral patterns rather than relying on static IoC indicators.

To ensure quality and suitability for analysis, the dataset underwent comprehensive preprocessing. Missing values were handled through imputation and removal of records with excessive missing data. Categorical variables were transformed using one-hot encoding to integrate them effectively into machine learning models. The dataset was standardized for uniform feature scaling, improving model performance and interpretability. Additionally, undersampling was applied to address the class imbalance and mitigate potential biases in model training. These preprocessing steps enhance the reliability and robustness of the dataset for threat intelligence analysis.

The structured CTI reports obtained from the VirusTotal CTI API allow for consistent and reliable experimental evaluations. By providing a wide range of extracted threat intelligence features, the dataset facilitates an in-depth analysis of the impact of different CTI attributes on ML-based IDS. Additionally, by leveraging well-defined preprocessing steps, the dataset maintains high-quality inputs, ensuring valid and reproducible research findings. The dataset is also used for comparison between the CTI Transfer Model and an IoC database approach, as IP addresses and vendor analysis results are utilized for blocklisting.

VI. RESULTS AND ANALYSIS

This section begins with a description of the parameter configuration for all classifiers that we used. The remainder of this section presents the experimental results addressing the key issues under investigation. Specifically, we examine the following: (1) the comparative performance of the IDS Model with and without CTI integration, to assess the impact of CTI on IDS effectiveness; (2) the defensive capabilities of integrating the IDS Model with the CTI Transfer Model versus using an IoC database, to evaluate which method provides superior threat detection and mitigation; and (3) the overall performance of the CTI Transfer Model when applied with

TABLE 5. Description of dataset features.

Feature	Description
time_start	The start time of the network session.
time_end	The end time of the network session.
duration	The duration of the network session in seconds.
src_ip	The source IP address from which the traffic originated.
dest_ip	The destination IP address to which the traffic was directed.
src_port	The source port number used by the originating device.
dest_port	The destination port number on the receiving device.
protocol	The protocol used for the communication (e.g., TCP, UDP).
flags	The TCP flags set during the communication, indicating various control and status information.
forwarding_status	The status of the packet forwarding (0 indicates no issues).
source_type_of_service	The type of service (ToS) field in the IP header, which is used to specify the priority of the packet.
ingress_packet_count	The number of packets received during the session.
ingress_byte_count	The total number of bytes received during the session.

ML techniques against the rule-based method, to analyze its effectiveness in real-world scenarios.

A. PARAMETER SETTINGS

In this research, the proposed IDS Model used a hybrid approach that combines SVM for supervised learning and K-means clustering for unsupervised learning. The SVM component allows us to classify traffic flow patterns accurately, leveraging the power of supervised learning to detect known threats. Meanwhile, K-means clustering provides the ability to detect outliers and emerging anomalies, which may not conform to known attack patterns, thus augmenting the system with unsupervised learning capabilities. This hybrid framework enables the IDS Model to effectively identify both known threats and emerging anomalies with precision.

For the development of the CTI Transfer Model, we employed K-means++, a variant of K-means clustering. The key difference between K-means and K-means++ lies in how the initial centroids are determined. K-means++ selects the initial centroids at the start of the algorithm, while K-means does not. Since the CTI Transfer Model begins with an empty dataset and progressively learns through online learning, it is crucial to establish an initial feed to make the model's learning process controllable and structured, which is why we require K-means++. In contrast, the IDS Model is pre-trained offline, so the initial setup of centroids is not as critical, allowing us to use the traditional K-means clustering algorithm.

Moreover, the input data types for the IDS and CTI Transfer Models are different, and therefore, there is no strict requirement that both models use the same algorithm. The CTI Transfer Model handles unlabelled CTI data, while the IDS Model processes network traffic flows. These two AI models solve different problems independently, and the choice of algorithms is based on their specific learning environments and data characteristics. Optimizing the hyperparameters for both K-means++ and K-means is a crucial step to ensure the efficiency and accuracy of both models.

TABLE 6. Hyperparameter settings.

Model	Classifier	Hyperparameter	Value
IDS Models	SVM	alpha	0.1
		penalty	l1
		loss	hinge
		random_state	456
	Kmeans	n_clusters	2
		random state	42
		max iter	100
CTI Transfer Model	Kmeans++	n_clusters	2
		random state	42

To this end, we embarked on exhaustive grid searches, systematically exploring various hyperparameter combinations. This meticulous process allowed us to pinpoint the configuration that yields the most favorable performance metrics. The culmination of these efforts is showcased in Table 6, where we present the meticulously tuned hyperparameter settings pertinent to our models.

Furthermore, the experiments were conducted on a system equipped with an AMD Ryzen 7 PRO 7840U processor with Radeon 780M Graphics and 64GB of RAM. Additionally, to assess the effectiveness of IDS with CTI integration, we use the F1 score, which balances precision and recall. This metric is widely used in IDS research to measure how well a system can correctly identify attacks while minimizing false positives and false negatives. By using the F1 score, we ensure a comprehensive evaluation of detection performance, demonstrating the impact of CTI on improving threat intelligence processing and overall IDS effectiveness.

B. IDS WITH CTI VS. IDS WITHOUT CTI

The empirical findings regarding the integration of CTI into the IDS Model for online learning reveal significant performance improvements. As illustrated in Figure 6, the blue line represents the ML-based IDS with CTI integration, which dynamically updates using real-time threat intelligence, while the red and green lines correspond to the ML-based IDS and DL-based IDS without CTI, operating as an offline model without continuous updates.

Online learning enables the model to update its parameters based on newly received data points iteratively. With each iteration, the model adapts in real-time, improving its detection capabilities. The figure clearly demonstrates that as online learning progresses, the F1 score of the IDS with the CTI Transfer Model steadily improves. In contrast, the IDS without CTI does not exhibit the same level of enhancement. After ten iterations, the IDS with CTI achieved an F1 score of 89.52%, whereas the offline IDS without CTI stagnated at 80.22%. This 9.29% increase in performance underscores the effectiveness of incorporating CTI into the IDS learning process, allowing for a more adaptive and responsive cybersecurity defense.

This experiment shows that integrating CTI into an IDS significantly enhances its efficiency and adaptability. Continuous CTI updates keep the IDS current with emerging threats, improving detection accuracy. The F1 score increase highlights its enhanced threat identification, making detection more reliable. Results confirm that real-time CTI is crucial for maintaining high IDS performance in dynamic environments and strengthening resilience against sophisticated cyber attacks.

Further analysis of detection performance highlights how the hybrid IDS model handles attack identification. The combination of supervised (SVM) and unsupervised (KMeans) models ensures balanced detection, with KMeans capable of flagging previously unseen threats, albeit at the cost of increased false positives. In our evaluation, SVM achieved a false positive rate of 7.70% and a false negative rate of 4.95%, while KMeans exhibited a significantly higher false positive rate of 42.78% and a false negative rate of 34.69%. While KMeans has a higher false positive rate, its broad anomaly detection scope helps identify novel attack patterns that SVM might miss. However, the primary cause of missed detections stems from KMeans failing to classify certain malicious traffic as anomalous rather than incomplete feedback to the SVM.

To address this, our hybrid IDS model introduces an outlier category, ensuring that uncertain traffic flagged by KMeans does not go unexamined. Instead of directly classifying such traffic as benign, it undergoes further analysis using the CTI Transfer Model, which queries real-time threat intelligence to validate its threat status. This process helps refine detection accuracy by allowing CTI to verify ambiguous cases, providing a mechanism for re-labeling previously misclassified flows based on new intelligence. By continuously updating training data and incorporating feedback from CTI, our system enhances its ability to detect evolving threats while minimizing the limitations of individual models.

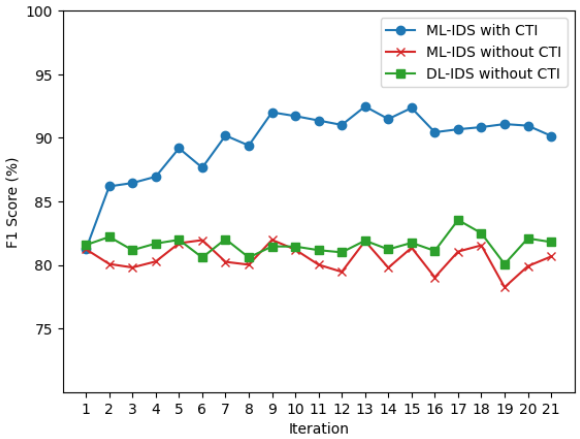


FIGURE 6. Comparison of F1 scores over iterations for IDS model with CTI transfer model.

Additionally, Figure 6 also presents a comparative evaluation between ML-based IDS and DL-based IDS, both with and without CTI integration. For DL models, we use an LSTM-based IDS from [29]. The results show that while both ML and DL models benefit from CTI, ML gains significantly more from CTI integration. This is because ML models can effectively learn from small sample sizes, and with the additional intelligence provided by CTI, they can continuously refine their detection capabilities without requiring massive datasets. In contrast, DL models typically require large datasets to generalize well, making them less efficient in scenarios where labeled attack data is limited and costly to collect. The CTI-enhanced ML model exhibits stronger adaptability, leveraging real-time intelligence updates to continuously improve performance, whereas the DL model benefits less from CTI due to its higher dependence on extensive training data. These results highlight that ML-based IDS, when integrated with CTI, is more effective in real-world cybersecurity applications where data availability is constrained.

Figure 7 presents a comparison of training time, memory consumption, and CPU utilization between ML-IDS and DL-IDS. The results clearly demonstrate that ML-IDS is significantly more resource-efficient than DL-IDS. In terms of training time, ML-IDS completes training in 3.22 seconds, whereas DL-IDS requires 59.30 seconds, indicating that deep learning models demand significantly longer computational time due to their complex architectures and iterative optimizations. Similarly, in terms of memory consumption, ML-IDS uses only 17.43 MB, while DL-IDS consumes 65.14 MB, as deep learning models typically require large parameter sets and activations, leading to higher memory overhead. Additionally, CPU utilization for ML-IDS remains at 36.6%, while DL-IDS consumes 79.5%, further emphasizing the lightweight nature of ML-based approaches.

Those findings highlight a key advantage of using ML-IDS over DL-IDS in real-world cybersecurity applications, where computational efficiency is crucial for deployability, scalability, and responsiveness. Given that network intrusion detection systems often operate in resource-constrained environments, ML-IDS presents a more practical solution by maintaining strong detection performance while ensuring minimal resource consumption.

While evaluating computational efficiency is essential, analyzing the complexity trade-offs between an IDS with and without CTI integration presents additional challenges. Complexity can be interpreted in different ways, often referring to time or space complexity in an algorithmic sense (big-O notation). However, comparing an online learning IDS with CTI and a traditional offline IDS is not straightforward due to fundamental differences in their training paradigms.

An offline IDS undergoes periodic large-scale training, incurring a one-time overhead when retrained with new data, whereas an online IDS updates incrementally as new threats emerge. Since these models operate differently, direct time complexity comparisons do not fully capture real-world performance. Additionally, inference complexity remains the same for both systems, as they rely on the same ML models such as SVM and K-Means. The primary computational overhead in an IDS with CTI arises when outlier traffic triggers a CTI lookup and an incremental model update. However, these updates run asynchronously, ensuring that real-time classification remains unaffected. Moreover, CTI updates occur only for uncertain (outlier) traffic, reducing unnecessary processing overhead. Although CTI lookups introduce additional processing, this cost is offset by enhanced detection accuracy and adaptability in handling evolving threats.

The scalability of continuous CTI updates depends on the frequency of outlier detections and the volume of IoCs retrieved. To mitigate potential delays, our implementation leverages caching mechanisms to reduce redundant queries for previously analyzed IoCs. The incremental update mechanism ensures that the IDS remains responsive, as updates are localized to new threat intelligence rather than requiring full dataset reprocessing. Furthermore, our evaluation includes resource usage metrics such as CPU consumption, memory utilization, and update time, demonstrating that the approach maintains efficient real-time operation without excessive computational burden.

Similarly, space complexity varies depending on implementation. An offline IDS stores training datasets on disk and loads them when retraining, while an online IDS processes data incrementally, reducing peak memory usage but requiring multiple updates over time. CTI lookups may introduce additional storage overhead when caching threat intelligence data, but this depends on implementation-specific factors such as database retention policies and memory management strategies.

Furthermore, in order to understand which types of sightings can produce better quality training data for the CTI Transfer Model, we explored the impact of training data

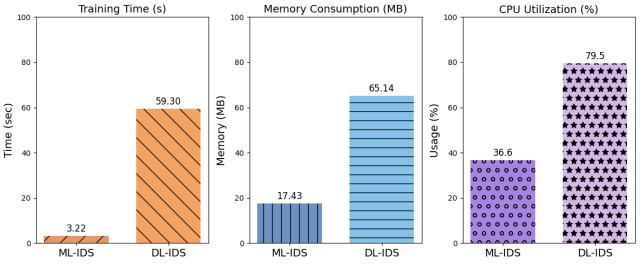


FIGURE 7. Comparison of ML-IDS and DL-IDS in terms of resource utilization.

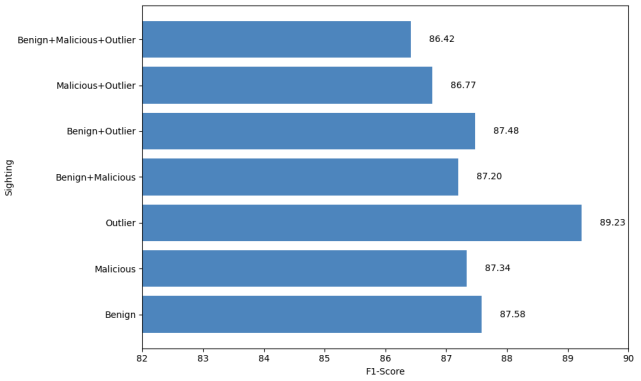


FIGURE 8. Comparison of F1 score for different types of sighting.

generated from different sightings. Figure 8 shows the F1 score performance for six different sighting categories: Benign, Malicious, Outlier, Benign and Malicious, Benign and Outlier, Malicious and Outlier, and all categories combined (Benign, Malicious, Outlier).

From the figure, it is evident that the outlier-only category scores the highest with an F1 score of 89.52%. In contrast, although other categories and combinations also enable the CTI Transfer Model to generate training data that can enhance the IDS Model, the training data generated from outlier sightings most significantly improves its performance. This is because the IDS Model has a certain level of recognition capability for benign and malicious types of sightings. Therefore, while the training data generated by the CTI Transfer Model can enhance the model's capabilities, the effect is not as strong as that of outliers.

Outliers are initially difficult for the IDS Model to accurately classify as benign or malicious, indicating a weaker recognition ability for these types. However, through the enhancement provided by the CTI Transfer Model, the previously uncertain outlier sightings can be classified more accurately. The combination of all categories (Benign, Malicious, and Outlier) performs the worst because it includes too many types of sightings, resulting in excessive diversity in the training data. This reduces the model's performance in identifying specific types of events. Due to the inclusion of too many types, the model struggles to learn and recognize all types effectively, which instead decreases the

FIGURE 9. Case of training data generation through CTI Transfer Model.

time_start	time_end	duration	src_ip	dest_ip	src_port	dest_port	proto	flags	router_ip_address
2/8/2023 20:02	2/8/2023 20:02	0	185.225.	140.	50948	80	TCP	.....RS.	140.123.103.250
2/8/2023 20:02	2/8/2023 20:02	0	185.225.	140.	39166	80	TCP	...AP.SF	140.123.103.250
2/8/2023 20:02	2/8/2023 20:02	1	185.225.	140.	46068	80	TCP	...AP.SF	140.123.103.250
2/8/2023 20:02	2/8/2023 20:02	0	185.225.	140.	43970	80	TCP	...AP.SF	140.123.103.250
2/8/2023 20:02	2/8/2023 20:02	1	185.225.	140.	41870	80	TCP	...AP.SF	140.123.103.250
2/8/2023 20:02	2/8/2023 20:02	1	185.225.	140.	51686	80	TCP	...AP.SF	140.123.103.250
2/8/2023 20:02	2/8/2023 20:02	0	185.225.	140.	49812	80	TCP	...AP.SF	140.123.103.250
2/8/2023 20:02	2/8/2023 20:02	0	185.225.	140.	59044	80	TCP	...AP.SF	140.123.103.250
2/8/2023 20:02	2/8/2023 20:02	1	185.225.	140.	56734	80	TCP	...AP.SF	140.123.103.250
2/8/2023 20:02	2/8/2023 20:02	1	185.225.	140.	48006	80	TCP	...AP.SF	140.123.103.250
2/8/2023 20:02	2/8/2023 20:02	0	185.225.	140.	54454	80	TCP	...AP.SF	140.123.103.250
2/8/2023 20:02	2/8/2023 20:02	1	185.225.	140.	60872	80	TCP	...AP.SF	140.123.103.250

FIGURE 10. Case of port obfuscation attack.

time_start	time_end	duration	src_ip	dest_ip	src_port	dest_port	proto	flags	router_ip_address
2/3/2023 8:56	2/3/2023 8:56	0	204.11	140.	80	1883	TCP	....A....	140.123.103.250
2/3/2023 8:56	2/3/2023 8:56	0	204.11	140.	80	1883	TCP	....A.R..	140.123.103.250
2/3/2023 12:53	2/3/2023 12:53	4	204.11	140.	80	12129	TCP	...AP.S.	140.123.103.250
2/3/2023 12:54	2/3/2023 12:54	0	204.11	140.	80	12129	TCP	....A....	140.123.103.250
2/3/2023 16:56	2/3/2023 16:56	0	204.11	140.	80	13307	TCP	....A....	140.123.103.250
2/3/2023 16:56	2/3/2023 16:56	0	204.11	140.	80	13307	TCP	....A....	140.123.103.250
2/3/2023 20:56	2/3/2023 20:56	4	204.11	140.	80	8149	TCP	...AP.S.	140.123.103.250
2/3/2023 20:57	2/3/2023 20:57	0	204.11	140.	80	8149	TCP	....A....	140.123.103.250
2/4/2023 1:02	2/4/2023 1:02	0	204.11	140.	80	4046	TCP	....A....	140.123.103.250
2/4/2023 1:02	2/4/2023 1:02	0	204.11	140.	80	4046	TCP	....A...F	140.123.103.250
2/4/2023 4:59	2/4/2023 4:59	4	204.11	140.	80	14942	TCP	...AP.S.	140.123.103.250
2/4/2023 4:59	2/4/2023 4:59	0	204.11	140.	80	14942	TCP	....A....	140.123.103.250

FIGURE 11. Case of port hopping attack.

overall performance of the model. These results suggest that the model's detection effectiveness seems to improve when incorporating the outlier category, possibly due to the more distinct characteristics of outlier instances.

Lastly, we perform the experiment with a case study that involves generating training data for the IDS Model. This case study showcases the CTI data we obtained, and the training data produced using our solution, DICI.

As shown in Figure 9, a structured CTI report for an IoC of outlier sighting is obtained via CTI Lookup. Subsequently, by utilizing the CTI Transfer Model to analyze the features extracted by the structured CTI report, we can confirm whether this IoC in the structured CTI report is benign or malicious. In other words, the IoC originally identified as an outlier by the IDS Model would be classified as benign or malicious by the DICI solution. Next, we use the IoC classified as malicious to perform a lookup in sighting records, identifying all sightings associated with this IoC. These sightings are then labeled as malicious, and the data is used to train the IDS model. Through this process, the model learns attack patterns associated with malicious IoCs.

Adversarial attacks pose a significant challenge to ML-based IDS, particularly gradient-based adversarial attacks, which manipulate extracted features to evade detection [30], [31]. While it is theoretically possible for attackers to manipulate packet data to reshape packet-level features, modifying packet content while maintaining the attack's effectiveness is highly challenging, especially in limited-knowledge attack scenarios. Most adversarial research assumes that attackers modify extracted network features rather than raw packet data. If an attacker alters only the packet data, the IDS may process a modified version of the traffic, but the network attack itself could fail due to protocol inconsistencies or unintended disruptions. This makes feature-based adversarial attacks more viable, as they allow attackers to evade IDS detection while still successfully executing their attacks–a strategy known as a double attack.

However, modifying extracted features post-capture requires privileged access to IDS devices, routers, or network probes, making such attacks impractical unless the attacker controls the IDS itself [32]. While an attacker can send crafted traffic, they cannot alter extracted features post-capture without compromising the IDS or its processing pipeline. Since ML-based IDS extracts features before analysis, adversarial research often assumes that attacks occur at the feature level, which is unrealistic without direct control over the IDS.

Instead, real-world adversarial tactics are more likely to involve host-based perturbations, where attackers modify their behavior at the command level rather than tampering with extracted network features. These host-based perturbations can alter network behavior, making the ML-based IDS fail to detect attacks because the behavioral shift does not match its training data. However, by integrating real-time CTI updates, DICI can detect attacks even if the initial packet or flow was classified as benign. CTI provides contextual intelligence beyond raw traffic analysis, allowing the system to recognize attack patterns linked to known adversarial tactics.

In this work, we evaluate DICI's performance against host-based perturbations, using port obfuscation and port hopping attacks as representative techniques. Port obfuscation involves dynamically altering source ports to disguise malicious traffic, making it difficult for traditional IDS models to recognize attack patterns. Similarly, port hopping allows attackers to frequently switch communication ports, bypassing firewall rules and predefined detection thresholds. Since ML-based IDS relies on static training data, these behavioral shifts cause the IDS to misclassify attacks as the new patterns deviate from its training data. By leveraging real-time CTI updates, DICI continuously adapts to evolving threats, mitigating the risk of adversarial evasion.

Figures 10 and 11 demonstrate the effectiveness of our proposed solution, DICI, in detecting these attack strategies. These results were achieved through the integration of CTI with the IDS Model, which enabled the generation of enhanced training data. In conventional IDS models, these techniques often go undetected due to the lack of real-time updates. However, by integrating CTI, DICI can identify

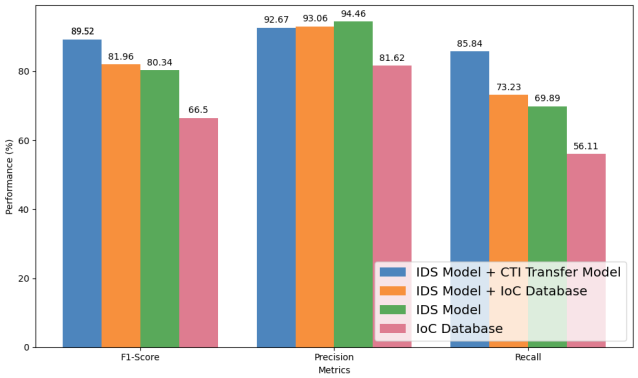


FIGURE 12. Performance metrics for IDS Model with CTI approaches.

IoCs associated with changing port numbers, continuously learning from new intelligence to maintain updated detection capabilities. This approach allows DICI to effectively detect port obfuscation and port hopping attacks, ensuring that adversaries cannot exploit static detection rules to evade classification. The dynamic learning mechanism enhances the IDS's resilience against evolving attack techniques, preventing adversarial tactics from bypassing detection. Sensitive IP information in the figures has been masked in response to privacy concerns.

C. CTI TRANSFER MODEL Vs. IoC DATABASE

In this section, we evaluate the defensive effectiveness of integrating the IDS Model with either the CTI Transfer Model or the IoC database. Figure 13 presents the performance metrics of four different configurations: IDS Model with CTI Transfer Model, IDS Model with IoC Database, standalone IDS Model, and IoC Database alone, with the vertical axis representing percentages and the horizontal axis representing various performance metrics.

The IDS Model integrated with the CTI Transfer Model achieved an F1 score of 89.52%, outperforming the IDS Model with the IoC database, standalone IDS Model, and IoC database. In terms of precision, the standalone IDS Model achieved a score of 94.46%. However, integrating the IDS Model with the IoC database reduced the precision to 93.06%, and further integration with the CTI Transfer Model resulted in a slight decrease to 92.67%. This approximate 1% reduction in precision can be attributed to the IoC database's partial inaccuracies in capturing real-world attacks, as reflected by its own precision of 81.26%.

Despite the minor decline in precision, the recall metrics showed notable improvement. The integration of the IDS Model with the CTI Transfer Model led to a substantial 15.95% increase in recall. This significant enhancement in recall indicates a greater ability to detect real threats.

In summary, while the IDS Model with the CTI Transfer Model experienced a slight 0.39% decrease in precision compared to the ML-based IDS with the IoC database, it demonstrated a 7.16% improvement in F1 score and a substantial 12.61% increase in recall. These results highlight that the IDS Model with the CTI Transfer Model not only enhances the detection of genuine network attacks but also significantly outperforms the IDS Model with the IoC database, underscoring its effectiveness in bolstering IDS capabilities.

D. ML ON CTI TRANSFER MODEL

In our CTI Transfer Model, we combine ML and heuristic methods to evaluate the analytical capabilities of CTI. We use the rule-based approach as the baseline because, to the best of our knowledge, this is the first study to process and analyze structured CTI reports using an ML model. The rule-based classifier follows predefined heuristics to determine whether an IoC is malicious or benign based on static rules derived from CTI reports. This approach is commonly used in traditional CTI processing, where security policies or thresholds define how intelligence is applied for detection. However, it lacks adaptability to evolving threats since it cannot generalize beyond predefined conditions. Since no prior ML-based methods exist for structured CTI processing, comparing our model to a rule-based classifier provides a meaningful evaluation of its effectiveness in dynamically adapting to real-time threat intelligence.

Furthermore, we extract all available information from structured CTI reports to determine the most effective automated approach for their analysis. Since CTI evaluation lacks labels, we perform sighting lookups on IoCs within the CTI and use labels derived from these sightings as external indicators to assess the model's performance.

Figure 13 compares the performance of KMeans++ and KMeans against a rule-based classifier for analyzing CTI reports. The x-axis represents the number of features extracted from structured CTI reports. The primary intention of this figure is to illustrate the impact of feature selection on model performance and to emphasize that increasing the number of features does not necessarily improve the F1 score.

Notably, we observe a performance degradation when fewer than five features are used, suggesting that the model requires a sufficient number of features to learn meaningful patterns effectively. Furthermore, while the dataset contains 110 features, using all of them does not provide additional benefits. These results highlight that simply increasing the number of features does not inherently enhance performance. Instead, it underscores the importance of feature quality over quantity in CTI reports, as high-quality features contribute more significantly to meaningful threat intelligence analysis.

Therefore, CTI platforms should focus on enhancing the quality of key features in CTI reports. Additionally, the rule-based classifier employs a consistent and transparent set of predefined rules applied uniformly across all CTI reports. This approach ensures that evaluation criteria remain constant, reducing the potential for subjective bias or algorithmic inconsistencies, thus providing robust and reliable analysis that can be systematically replicated.

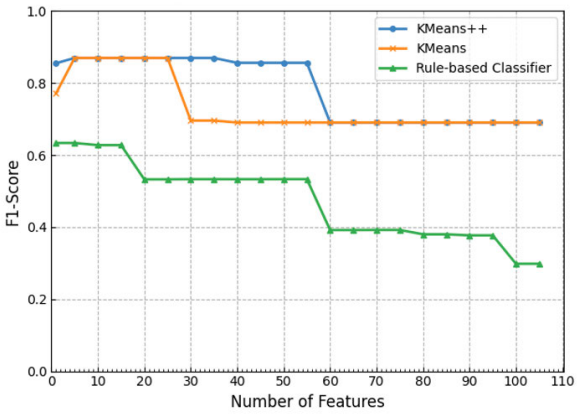


FIGURE 13. F1 score of ML on CTI transfer model.

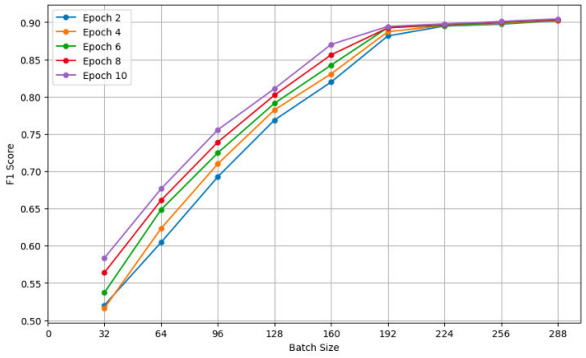


FIGURE 14. F1 score variation with batch size across different epochs.

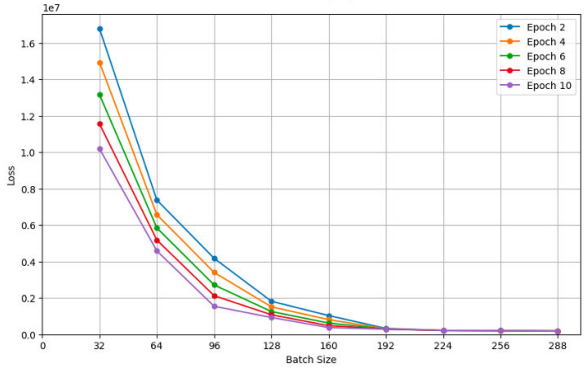


FIGURE 15. Loss variation with batch size across different epochs.

Furthermore, KMeans++ improves the average F1 score by 30.92% compared to the rule-based classifier, indicating that KMeans++ offers a more effective approach for analyzing CTI reports and identifying potential threats compared to traditional heuristic methods. As the features and characteristics of structured CTI reports evolve or new ML methods emerge, future research can build on this work to further enhance CTI report analysis.

Lastly, we employed a rigorous grid search technique for both KMeans++ and KMeans models, carefully configuring relevant parameters such as model initialization to ensure a thorough exploration of the parameter space. For the rule-based classifier, we developed specific rules based on key features within the CTI reports to automate the classification process. These rules were derived from expert insights, allowing for efficient analysis of CTI data and the identification of potential threats.

E. BATCH SIZE AND MODEL PERFORMANCE

The online learning capability of the IDS Model means that training data is continuously updated and processed in batches, rather than all at once, as in traditional offline learning. This incremental learning process enables the model to adapt to new data in real-time. However, it also introduces challenges in determining the optimal batch size and number of epochs required for training. These configurations directly affect the trade-off between model performance, computational resource usage, and the dynamic nature of the incoming data. Balancing these parameters is essential to ensuring both the effectiveness and efficiency of the IDS in real-world applications.

Smaller batch sizes can limit the model's overall performance, as they may not provide sufficient data for effective learning in each iteration. Conversely, larger batch sizes initially improve performance but demand significantly more computational resources. However, after a certain point, further increases in batch size yield diminishing returns, where additional resources do not result in proportional performance gains. As a result, selecting a moderate batch size combined with a higher number of training iterations often proves to be the most effective strategy for achieving a higher F1 score.

Figure 14 illustrates the relationship between batch size and the F1 score. The horizontal axis represents the batch size, and the vertical axis represents the F1 score, with each line corresponding to a different number of epochs. The analysis reveals that for batch sizes below 224, increasing the number of epochs consistently improves the F1 score. Similarly, larger batch sizes also contribute to higher F1 scores. However, when the batch size exceeds 224, further increases in both batch size and the number of epochs fail to yield improvements. This suggests that for practical applications, identifying an optimal batch size is essential to avoid unnecessary computational costs while still ensuring sufficient training iterations to enhance model performance.

Additionally, Figure 15 explores the relationship between batch size and model loss. As batch size increases, the loss function steadily decreases, reflecting the model's improving ability to minimize errors. The loss function reaches its lowest point at a batch size of 224, indicating optimal performance at this size. Beyond this threshold, further increases in batch size and epochs do not lead to significant reductions in loss, suggesting a saturation point where further fine-tuning results in minimal gains. This observation highlights the importance

of careful parameter selection during training, as the appropriate batch size can significantly impact the model's overall performance and efficiency.

VII. CONCLUSION AND FUTURE WORKS

In this study, we proposed an approach that integrates CTI to enhance ML-based IDS, enabling dynamic model updates and improved threat detection. Our results revealed that the integration of the IDS Model with the CTI Transfer Model achieved a 9.29% higher F1 score compared to the IDS Model solely. We also observed that increasing the number of features in CTI reports did not enhance the F1 score, emphasizing the importance of feature quality. Notably, KMeans++ clustering improved the F1 score by 30.92% compared to rule-based methods, underscoring the effectiveness of machine learning for analyzing complex data.

These findings underscore the value of integrating CTI into IDS, especially in enabling the capability of dynamic model updates through online learning environments, where both security and efficiency are critical. The results also emphasize the need for careful selection of machine learning algorithms to optimize threat detection.

While our results demonstrate the effectiveness of using CTI to enhance ML-based IDS, there are several directions for further exploration. First, developing algorithms that enable ML-based IDS to evolve autonomously, minimizing manual intervention. This includes enhancing the unsupervised component of the IDS model with advanced anomaly detection techniques or deep clustering models to improve its ability to identify subtle or highly sophisticated attack patterns that traditional clustering methods may overlook. Second, future research should explore the integration of multiple CTI platforms dynamically to enhance threat intelligence coverage, improve detection accuracy, and reduce reliance on a single source. By leveraging diverse CTI data, IDS models can achieve a more comprehensive understanding of evolving attack patterns and improve adaptability to emerging threats. Finally, privacy-preserving methods for data sharing are crucial for enabling the correlation between sightings and CTI without compromising sensitive information. Developing these methods will facilitate more effective analysis while ensuring user privacy is maintained.

REFERENCES

[1] A. Halbouni, T. S. Gunawan, M. H. Habaebi, M. Halbouni, M. Kartiwi, and R. Ahmad, "Machine learning and deep learning approaches for cybersecurity: A review," *IEEE Access*, vol. 10, pp. 19572–19585, 2022.

[2] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 1, pp. 41–50, Feb. 2018.

[3] P.-C. Lin et al., "Correlation of cyber threat intelligence with sightings for intelligence assessment and augmentation," *Comput. Netw.*, vol. 228, Jun. 2023, Art. no. 109736.

[4] T. D. Wagner, K. Mahbub, E. Palomar, and A. E. Abdallah, "Cyber threat intelligence sharing: Survey and research directions," *Comput. Secur.*, vol. 87, Nov. 2019, Art. no. 101589.

[5] R. Mills, A. K. Marnerides, M. Broadbent, and N. Race, "Practical intrusion detection of emerging threats," *IEEE Trans. Netw. Service Manage.*, vol. 19, no. 1, pp. 582–600, Mar. 2022.

[6] Y. Kurogome et al., "EIGER: Automated IOC generation for accurate and interpretable endpoint malware detection," in *Proc. 35th Annu. Comput. Secur. Appl. Conf.*, Dec. 2019, pp. 687–701.

[7] A. Niakanlahiji, L. Safarnejad, R. Harper, and B.-T. Chu, "IoCMiner: Automatic extraction of indicators of compromise from Twitter," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2019, pp. 4747–4754.

[8] S. Fujii, N. Kawaguchi, T. Shigemoto, and T. Yamauchi, "CyNER: Information extraction from unstructured text of CTI sources with noncontextual IOCs," in *Proc. Int. Workshop Secur.*, Jan. 2022, pp. 85–104.

[9] R. Basheer and B. Alkhatib, "Threats from the dark: A review over dark web investigation research for cyber threat intelligence," *J. Comput. Netw. Commun.*, vol. 2021, pp. 1–21, Dec. 2021.

[10] J. Zhao, Q. Yan, J. Li, M. Shao, Z. He, and B. Li, "TIMiner: Automatically extracting and analyzing categorized cyber threat intelligence from social data," *Comput. Secur.*, vol. 95, Aug. 2020, Art. no. 101867.

[11] A. Berady, M. Jaume, V. V. T. Tong, and G. Guette, "From TTP to IoC: Advanced persistent graphs for threat hunting," *IEEE Trans. Netw. Service Manag.*, vol. 18, no. 2, pp. 1321–1333, Jun. 2021.

[12] N. Rastogi, S. Dutta, A. Gittens, M. J. Zaki, and C. Aggarwal, "TINKER: A framework for open source cyberthreat intelligence," in *Proc. IEEE Int. Conf. Trust, Secur. Privacy Comput. Commun. (TrustCom)*, Dec. 2022, pp. 1569–1574.

[13] A. H. Nursidiq and C. Lim, "Cyber threat hunting to detect unknown threats in the enterprise network," in *Proc. IEEE Int. Conf. Cryptogr., Informat., Cybersecur. (ICoCICs)*, Aug. 2023, pp. 303–308.

[14] R. Panigrahi et al., "A consolidated decision tree-based intrusion detection system for binary and multiclass imbalanced datasets," *Mathematics*, vol. 9, no. 7, p. 751, Mar. 2021. [Online]. Available: https://www.mdpi.com/2227-7390/9/7/751

[15] T.-T.-H. Le, H. Kim, H. Kang, and H. Kim, "Classification and explanation for intrusion detection system based on ensemble trees and SHAP method," *Sensors*, vol. 22, no. 3, p. 1154, Feb. 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/3/1154

[16] C. Anthony, W. Elgenaidi, and M. Rao, "Intrusion detection system for autonomous vehicles using non-tree based machine learning algorithms," *Electronics*, vol. 13, no. 5, p. 809, Feb. 2024. [Online]. Available: https://www.mdpi.com/2079-9292/13/5/809

[17] P.-S. Lin, Y.-C. Lai, M.-L. Liao, S.-P. Chiu, and J.-L. Chen, "Hybrid clustering mechanisms for high-efficiency intrusion prevention," in *Proc. 26th Int. Conf. Adv. Commun. Technol. (ICACT)*, Feb. 2024, pp. 1–6.

[18] B. Xie, X. Dong, and C. Wang, "An improved K-means clustering intrusion detection algorithm for wireless networks based on federated learning," *Wireless Commun. Mobile Comput.*, vol. 2021, no. 1, pp. 1–15, Jan. 2021.

[19] Shodan API. (2024). *Shodan*. [Online]. Available: https://developer.shodan.io

[20] MalwareBazaar API. (2024). *MalwareBazaar*. [Online]. Available: https://bazaar.abuse.ch/api/

[21] ThreatMiner API. (2024). *ThreatMiner*. [Online]. Available: https://www.threatminer.org/api.php

[22] Shadon API. (2024). *Shadon*. [Online]. Available: https://www.greynoise.io/features/product-feature-api

[23] Virustotal API. (2023). *Virustotal*. [Online]. Available: https://docs.virustotal.com/reference/overview

[24] phaag. (2023). *Phaag/nfdump*. [Online]. Available: https://github.com/phaag/nfdump

[25] G. Lemaître, F. Nogueira, and C. K. Aridas, "Imbalanced-learn: A Python toolbox to tackle the curse of imbalanced datasets in machine learning," *J. Mach. Learn. Res.*, vol. 18, no. 17, pp. 1–5, 2017.

[26] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Nov. 2011.

[27] M. Waskom, "Seaborn: Statistical data visualization," *J. Open Source Softw.*, vol. 6, no. 60, p. 3021, Apr. 2021.

[28] J. D. Hunter, "Matplotlib: A 2D graphics environment," *Comput. Sci. Eng.*, vol. 9, no. 3, pp. 90–95, 2007.

[29] Y. Imrana, Y. Xiang, L. Ali, and Z. Abdul-Rauf, "A bidirectional LSTM deep learning approach for intrusion detection," *Expert Syst. Appl.*, vol. 185, Dec. 2021, Art. no. 115524, doi: 10.1016/j.eswa.2021.115524.

[30] D. Han et al., "Evaluating and improving adversarial robustness of machine learning-based network intrusion detectors," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 8, pp. 2632–2647, Aug. 2021, doi: 10.1109/JSAC.2021.3087242.

[31] Y.-D. Lin et al., "ELAT: Ensemble learning with adversarial training in defending against evaded intrusions," *J. Inf. Secur. Appl.*, vol. 71, Dec. 2022, Art. no. 103348, doi: 10.1016/j.jisa.2022.103348.
[32] G. Apruzzese, M. Andreolini, L. Ferretti, M. Marchetti, and M. Colajanni, "Modeling realistic adversarial attacks against network intrusion detection systems," *Digit. Threats, Res. Pract.*, vol. 3, no. 3, pp. 1–19, Feb. 2022, doi: 10.1145/3469659.

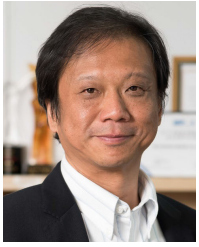
YING-DAR LIN (Fellow, IEEE) received the Ph.D. degree in computer science from the University of California at Los Angeles (UCLA) in 1993. He was a Visiting Scholar at Cisco Systems, San Jose, from 2007 to 2008; CEO at Telecom Technology Center, Taiwan, from 2010 to 2011; and the Vice President of National Applied Research Labs (NARLabs), Taiwan, from 2017 to 2018. He is currently a Chair Professor of computer science at National Yang Ming Chiao Tung University (NYCU), Taiwan. He co-founded L7 Networks Inc., in 2002 and O'Prueba Inc., in 2018. His research interests include cybersecurity, wireless communications, network softwarization, and machine learning for communications. He served or is serving on the editorial boards for several IEEE journals and magazines, including the Editor-in-Chief for IEEE COMMUNICATIONS SURVEYS AND TUTORIALS, from 2017 to 2020.

YI-HSIN LU received the master's degree in cybersecurity from National Yang Ming Chiao Tung University (NYCU) in 2024. His research interests include network security, intrusion detection, and cyber threat intelligence.

REN-HUNG HWANG (Senior Member, IEEE) received the Ph.D. degree in computer science from the University of Massachusetts at Amherst, Amherst. He is currently the Dean of the College of Artificial Intelligence, National Yang Ming Chiao Tung University (NYCU), Taiwan. Before joining NYCU, he was with National Chung Cheng University, Taiwan, from 1993 to 2022. He has published more than 250 international journals and conference papers. He was the Dean of the College of Engineering from 2014 to 2017. His current research interests include deep learning, wireless communications, network security, and cloud/edge/fog computing. He received the IEEE Best Paper Award from IEEE UbiMedia 2018, IEEE SC2 2017, and IEEE IUCC 2014.

YUAN-CHENG LAI received the Ph.D. degree from the Department of Computer and Information Science, National Chiao Tung University, in 1997. He joined a Faculty Member with the Department of Information Management, National Taiwan University of Science and Technology, in August 2001; and has been a Distinguished Professor, since June 2012. His research interests include performance analysis, software-defined networking, wireless networks, and the IoT security.

DIDIK SUDYANA received the Ph.D. degree from the Department of Electrical Engineering and Computer Science (EECS), National Yang Ming Chiao Tung University (NYCU), in 2024. He is currently an Assistant Professor at the Computer and Network Center, National Cheng Kung University (NCKU), Taiwan. His research interests include cybersecurity, machine learning, and network design and optimization.



WEI-BIN LEE (Senior Member, IEEE) received the Ph.D. degree from National Chung Cheng University in 1997. He was a Professor at the Department of Information Engineering and Computer Science, Feng Chia University. He was also a Visiting Professor at Carnegie Mellon University, USA, and The University of British Columbia, Canada. Since 2021, he has been the CEO of the Hon Hai Research Institute and the Director of the Information Security Research Center. Before joining Hon Hai Research Institute, CEO Wei-Bin Lee held important positions in Taipei City Government, Taipei Fubon Bank, Fubon Financial Holdings, and Feng Chia University. His research interests include network security, cryptography, digital rights management, and privacy/security management and governance.