Machine Learning Enabled Intrusion Detection: DistilBERT and Tree Ensemble in Elastic Cloud

Yujin Lee
Purdue University
West Lafayette, IN, 48907
yujinlee@purdue.edu

Eason Yi Zheng
Purdue University
West Lafayette, IN, 48907
zheng719@purdue.edu

Abstract – From DDoS to TCP-SYN attack, the rapid evolution of cyber threats increased tremendously due to the adaptation of IoT devices. Conventional method of relying on analysts to process log files and network traffic puts excessive pressure on the security integrity of the Incident Response team due to time sensitivity and resource constraints. The advent of machine learning generated detection prevention systems is one of the promising directions that can be employed to reduce the workload of analysts by automating away less sophisticated common cyberattacks. This research investigates the training of three popular datasets-UNR-IDD, NSL-KDD, and CIC-IDS2018-using DistilBERT and Tree Ensemble. DistilBERT is a popular method due to its lightweight and fast deployment capabilities. Tree Ensemble is known to generate high precision and recall algorithms in cyberspace malicious activity detection. The machine learning models were trained using clusters offered by AWS and GCP through the interface of Elastic Cloud; their performances were determined to be extremely similar while AWS offers more pricing flexibility. The results revealed the precision and recall percentage of six total experiments. In addition, the binary classification of Normal versus Attack in all six experiments showed phenomenal precision scores despite the less impressive recall percentage across the board. By analyzing the experimental results generated, the study has shown possible vectors to discover limitations and best use cases of the studied algorithms in mitigating evolving cyber threats. The findings from the experiments showcase the potential of applying machine learning methods to enhance cyber threats detection.

Index Terms – Cybersecurity, Threat Hunting, Machine Learning, Elastic Kibana, DistilBERT, Anomaly Detection

I. Introduction

The conventional method of conducting threat hunting and incident response typically involves analyzing the artifacts generated by an exploration tool, such as Elastic Kibana, Grafana, and Arkime, capable of providing visualization for log analysis and application monitoring. This is especially true in the military cyberspace defense setting. For example, one of the highly coveted malicious accesses was the AFNET in recent years according to the Air Force Materiel Command. Within the AFNET, there are three subdomains: NiPR, SIPR, and JWICS. For everyday users, NiPR is a network where all the unclassified materials and traffic are processed and resided. The SIPR network is where classified materials up to "secret" category reside. The highest level of classification network within the Department of Defense is the JWICS network where "top-secret" materials and traffic are handled.

Due to the increasing threats in cyberspace and rising popularity in IoT devices, the amount of data involving security logs and malicious network traffic that cybersecurity professionals in the Depart of Defense must process has increased tremendously. This is especially true when one of the most common threats comes from insiders who unknowingly carelessly handle malicious emails causing the network to be exposed from within. According to Tessian's report in 2024, roughly there was a 47% increase in the insider threats frequency between 2018 and 2020 globally. And according to Teramind and NOAA, some of the common indicators of an insider threat are 1) seeking access to classified information above their clearance level, 2) purposely mishandling classified or sensitive material, 3) excessive data downloads, 4) unauthorized access attempts. The complex situation is one major contributor as to why intrusion detection and prevention within one of the most important networks in the United State can be difficult to protect completely due to the lack of manpower and necessary training.

To counter the ever-increasing cyber threats, machine learning is one of the up-and-coming solutions. Machine

learning can be applied to threat hunting, IDS, and IPS on a platform such as Elastic Kibana to reduce the workload of the analysts needing to process the exchanges of countless logs and files in the network. To construct a well-established machine learning tool in Elastic Kibana, dataset feature selection is important. This research paper focuses on the reference paper of UNR-IDD as the benchmark in addition to exploring NSL-KDD and CIC-IDS2018.

While feature extraction and selection are two major factors in constructing an effective machine learning model, choosing the appropriate training model is just as important. Depending on the size of the training model and available spaces on the Elastic Cloud cluster, only models that fit the computational and storage limits should be considered. When comparing the results of the datasets, the focus is precision, accuracy, and f1 score. The intent of this paper is to provide a machine learning solution in aiding Elastic Kibana to reduce human workload through comparing the results to the benchmark. The finished product can then be deployed on the cloud 24/7 and streamlined with data ingestion capabilities to import network traffic; this will enable anomalies detection and act as an intrusion detection and prevention system that can self-perpetuate its improvement on precision and recall for years to come.

II. Literature Review

In this section, we review existing literature on cases of anomaly detection visualized using Elasticsearch or Kibana, APT detection cases, and intrusion detection using deep learning, with a focus on leveraging Elastic Kibana for AI-assisted anomaly detection.

H. Almohannadi et al. [1] collected cyber-attack log data using honeypots on the AWS cloud and analyzed it using Elasticsearch, Logstash, and Kibana (ELK). This research identified attacker behaviors and patterns, providing actionable insights to prevent future attacks.

K. A. N. Madhuvantha et al. [2] used Kibana to visualize the output data while securing the information using blockchain. This research highlighted the effectiveness of Kibana in processing real-time HTTP and network traffic logs for AI-based anomaly detection.

Several studies have explored the use of AI and machine learning for detecting Advanced Persistent Threats (APTs).

Aloseel et al. [3] introduced social engineering-based techniques to collect intelligence about APTs focusing on the early stages of attacks.

Siddiqui et al. [4] proposed a mechanism for detecting anomalous network flows by comparing normal and malicious traffic using fractal dimension-based machine learning classification.

Bodström and Hämäläinen [5] developed a novel deep learning stack that combines various methods to improve APT behavior detection. There are several studies on deep learning approaches for intrusion detection.

Yin et al. [6] adopted recurrent neural networks (RNNs) to develop an intrusion detection system with high accuracy using the NSL-KDD dataset.

Shone et al. [7] integrated autoencoders and random forests for detecting anomalies, achieving significant improvements in performance metrics.

During the initial literature research stage, aside from the two machine learning models that were chosen- DistilBERT and Tree Ensemble- five other models were also examined for their applicability. These were Gradient Long Short-Term Memory (LSTM), Boosting Classifier (GBC), Random Forest (RF), Support Vector Machine (SVM), and Multi-Layer Perceptron (MLP).

Gradient Boosting Classifier (GBC) is an ensemble method that builds models sequentially, optimizing a differentiable loss function by correcting errors made by previous models [13]. Each new model, $h_m(x)$ is added to minimize the loss function $L\big(y, F_m(x)\big)$, where $F_m(x)$ is the ensemble model at iteration m:

$$F_m(x) = F_{m-1}(x) + \nu \cdot h_m(x)$$

where $v \in (0,1]$ is the learning rate. The weak learner $h_m(x)$ is trained on the gradient of the loss function with respect to the predictions:

$$h_m(x) = \arg\min_h \sum_{i=1}^{N} \Big(\nabla_{F_{m-1}(x_i)} L\big(y_i, F_{m-1}(x_i)\big) - h(x_i)\Big)^2$$

By iteratively fitting the negative gradient of the loss function, GBC creates a strong classifier.

Random Forest (RF) is an ensemble learning algorithm based on decision trees, where each tree $T_i(x)$ is trained on a random subset of the data (bagging) and a random selection of features [14]. The final classification result is determined by majority voting:

$$y = mode\big(T1(x), T2(x), \dots, Tn(x)\big).$$

Mathematically, for a given feature space, each tree partitions the space into regions $R_{i,j}$ and predicts class probabilities as:

$$P(c \mid x) = N1j \sum 1x \in Ri, jP(c \mid Ri, j),$$

where $P(c \mid R_{i,j})$ is the probability of class c within region $R_{i,j}$. Aggregating predictions reduces overfitting and improves generalization.

Support Vector Machine (SVM) identifies a hyperplane that maximally separates two classes of data [15]. For linearly separable data, it solves:

$$\min_{\boldsymbol{w},b} \frac{1}{2}||\boldsymbol{w}||^2, \quad \text{subject to } y_i(\boldsymbol{w}^\top \boldsymbol{x_i} + b) \geq 1,$$

where $\boldsymbol{w}$ is the weight vector, b is the bias, and $y_i \in \{-1,1\}$ are class labels.
For non-linear data, the kernel trick maps data to a higher-dimensional space where it becomes linearly separable. The optimization problem becomes:

$$\max_{\alpha} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j K(\boldsymbol{x_i}, \boldsymbol{x_j}) - \sum_{i=1}^{N}\alpha_i,$$

subject to $\sum_{i=1}^{N}\alpha_i y_i = 0 \; and \; 0 \leq \alpha_i \leq C$, where $K(\boldsymbol{x_i}, \boldsymbol{x_j})$ is the kernel function.

Multi-layer Perceptron (MLP) is a type of artificial neural network that maps input data $\boldsymbol{x}$ through multiple layers of nonlinear transformations to predict an output $\boldsymbol{y}$ [16]. The hidden layers are computed as:

$$\boldsymbol{h^l} = \sigma(\boldsymbol{W^l h^{l-1}} + \boldsymbol{b^l})$$

where $\boldsymbol{W^l}$ and $\boldsymbol{b^l}$ are the weights and biases of layer l, σ is the activation function (e.g., ReLU, Sigmoid), and $\boldsymbol{h^0} = \boldsymbol{x}$. The output layer computes:

$$\hat{\boldsymbol{y}} = \text{softmax}(\boldsymbol{W^L h^{L-1}} + \boldsymbol{b^L})$$

for classification tasks. MLP is trained using backpropagation and gradient descent to minimize a loss function, such as cross-entropy:

$$L = -\sum_{i} y_i \log(\hat{y}_i)$$

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) designed to model sequential data by addressing the vanishing gradient problem [17]. LSTM uses memory cells with input, forget, and output gates to regulate information flow. The gates are computed as:

$$\begin{gathered}\boldsymbol{f_t} = \sigma(\boldsymbol{W_f x_t} + \boldsymbol{U_f h_{t-1}} + \boldsymbol{b_f}),\\ \boldsymbol{i_t} = \sigma(\boldsymbol{W_i x_t} + \boldsymbol{U_i h_{t-1}} + \boldsymbol{b_i}),\\ \boldsymbol{o_t} = \sigma(\boldsymbol{W_o x_t} + \boldsymbol{U_o h_{t-1}} + \boldsymbol{b_o}),\\ \boldsymbol{c_t} = \boldsymbol{f_t} \odot \boldsymbol{c_{t-1}} + \boldsymbol{i_t} \odot \tanh(\boldsymbol{W_c x_t} + \boldsymbol{U_c h_{t-1}} + \boldsymbol{b_c}),\\ \boldsymbol{h_t} = \boldsymbol{o_t} \odot \tanh(\boldsymbol{c_t}),\end{gathered}$$

where $\odot$ denotes element-wise multiplication, σ is the sigmoid activation, and $tanh$ is the hyperbolic tangent. LSTM's gating mechanism allows it to retain long-term dependencies effectively.

III. Research Methodology

The initial steps in this research involved brainstorming a research project that can be incorporated into the author's PhD research venues. In short, the author's PhD research motivation is in reducing the rare earth elements mining through recycling via advanced HPLC separation and recovery processes. One of the essential steps in the research is configuring the experimental parameters through simulation before confirming via actual lab experimentations. In this process, all chemical simulations rely on an outdated simulation toolkit called VERSE, which was developed in the 1980's by Dr. Wang's group at Purdue University. Due to the benefits of autoscaling, user-friendly development environment, security enhancement, cost-effectiveness, and versatility of the cloud storage, the plausibility to place VERSE on the cloud was assessed.

In the preliminary step, pricing information of a long-term in-house storage and computational units-Geddes (Purdue University)- was assessed. However, this paper focuses on the security aspect of cloud computing and storage; therefore, experimentations conducted using a free trial version of Elastic Cloud on AWS and GCP remained the focal point. From initial experimentation, Elastic Cloud hosted using AWS and GCP offered similar speed and performance when multiple test samples were tried to determine the winner. In the later stage, GCP was chosen over AWS due to the already setup GCP account offering faster logins and user authentication streamline.

A. Project Process

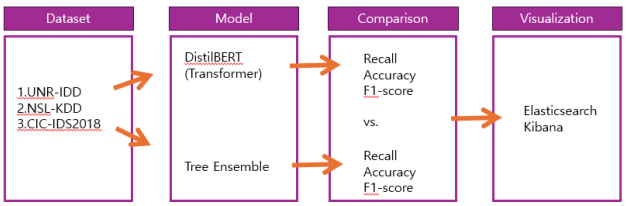

Fig. 1. Project Process

Our research follows a structured process to address the challenge of excessive network logs in Elastic Kibana. The process begins with identifying the issues and focuses on reducing the manual effort required to review large volumes of logs in Elasticsearch and Kibana. To achieve this, we selected three datasets: UNR-IDD, NSL-KDD, and CIC-IDS2018, considering their relevance and diversity in intrusion detection scenarios. We evaluated their effectiveness in handling structured and unstructured data using machine learning models, including DistilBERT, a

transformer-based deep learning architecture, and Tree Ensemble, a traditional machine learning approach. The classification results were visualized in Elastic Kibana to provide an intuitive interface for further analysis and decision making.

B. Datasets

The datasets used in this research were chosen to ensure a comprehensive and diverse data coverage for intrusion detection. The UNR-IDD dataset [8] focuses on network port statistics including activity records and delta statistics and addresses the limitations of previous datasets by incorporating the latest cyber-attack types. The NSL-KDD dataset [9] simulates various intrusion scenarios in a military network environment and mitigates the redundancy and bias issues present in previous versions. Finally, the CIC-IDS2018 dataset [10] provides realistic network traffic data with 50 attacker machines and 420 victims, covering 7 types of attacks. Together, these datasets provide a strong foundation for training and evaluating machine learning models for anomaly detection.

C. Model Selection

As stated on the Elastic Cloud official website, a list of 11 pre-trained machine learning models were compatible with the interface. These were BERT, BART, DPR bi-encoders, DistilBERT, ELECTRA-Google, MobileBERT, RoBERTa, etc. Due to the limited storage space and RAM space available on the free-trail Elastic Cloud cluster, the size of the dataset and ML model were taken into consideration.

Two machine learning models were chosen to leverage their unique strengths in handling different types of data. DistilBERT [11] is a transformer-based deep learning model. It uses self-attention mechanisms for efficient processing of sequential and textual data, which makes it particularly suitable for identifying patterns in network traffic logs. Its lightweight architecture facilitates faster computation, although it comes with a slight performance penalty compared to larger transformer models. Since DistilBERT was not an existing selection option already available on Elastic Cloud, it had to be coded and then imported to the cluster. In contrast, the Tree Ensemble model [12] trained and provided by Elasticsearch excels at capturing non-linear relationships in structured data using decision trees, making it effective for analyzing network traffic features.

D. Literature Multi-Class Classification Performance

In the research work done by the University of Nevada, the precision, recall and f-measure of the attack vectors were composed after observing the performance of the Random Forest machine learning algorithm. [8] Similarly, the performance of the other algorithms such as SVM, MLP, BC, KNC, and ABC were also observed. The two dabbles are presented in this paper for the purpose of referencing the performance comparison only.

TABLE I: Multi-Class Classification-Random Forest (UNR-IDD)

Label	P	R	F
Blackhole	0.98	0.98	0.98
Diversion	0.99	0.97	0.98
Overflow	0.98	0.76	0.86
PortScan	0.91	0.94	0.92
TCP-SYN	0.91	0.92	0.92
Normal	1.0	1.0	1.0

TABLE II: Multi-Class Classification Models (UNR-IDD)

Algorithm	P_u	R_u	F_u
SVM	0.89	0.79	0.81
MLP	0.59	0.54	0.54
BC	0.95	0.93	0.94
KNC	0.79	0.75	0.77
ABC	0.69	0.59	0.55
RF	0.96	0.92	0.94

IV. Experiments and Results

A. Network Detection using DistilBERT model

First, experiments were conducted to evaluate the performance of the DistilBERT model in detecting anomalies on three datasets (UNR-IDD, NSL-KDD, CIC-IDS2018). The results were measured in terms of key performance metrics including accuracy, recall, and f1-score. Elasticsearch and Kibana were used to visualize the results.

TABLE III: DistilBERT Performance on the UNR-IDD dataset

Label	***P***	***R***	***F***
Blackhole	0.97	0.99	0.98
Diversion	1.00	0.97	0.98
Normal	1.00	1.00	1.00
Overflow	0.89	0.77	0.82
PortScan	0.94	0.94	0.94

The results from the UNR-IDD dataset showed a high level of accuracy in distinguishing between normal and

malicious network traffic. In particular, the performance on the Normal data was 100%. The predicted classifications were almost identical to the ground truth, indicating the robustness of the model in handling structured data.

TABLE IV: DistilBERT Performance on the NSL-KDD dataset

Label	***P***	***R***	***F***
back	0.0	0.0	0.0
guess_passwd	0.0	0.0	0.0
ipsweep	1.0	1.0	1.0
neptune	1.0	0.9877	0.9938
nmap	0.0	0.0	0.0
normal	0.7969	0.9811	0.8795
pod	0.0	0.0	0.0
portsweep	0.6667	0.5	0.5714
rootkit	0.0	0.0	0.0
satan	0.5789	1.0	0.7333
smurf	0.6667	1.0	0.8

On the NSL-KDD dataset, which contains various intrusion scenarios, the DistilBERT model showed several aspects. It showed high detection accuracy for attacks such as ipsweep and neptune but failed to detect attacks such as nmap and rootkit.

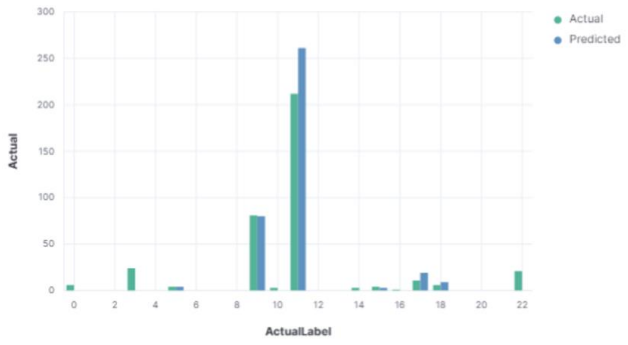

Fig. 2. Actual labels and predicted labels of the DistilBERT model on the NSL-KDD dataset

The green bars are the actual labels, and the blue bars are the predicted labels. Some attacks have bars of equal length, while others are exceeded or not detected.

TABLE V: DistilBERT Performance on the CIC-IDS2018 dataset

Label	**P**	**R**	**F**
Benign	0.6772	0.2629	0.3787
Infiltration	0.6564	0.9183	0.7656

For the CIC-IDS2018 dataset, which provides realistic network traffic data, the DistilBERT model showed low accuracy.

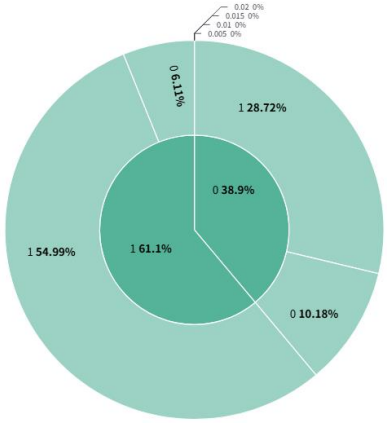

Fig. 3. Actual labels (inner circle) and predicted labels (outer circle) of the DistilBERT model on the CIC-IDS2018 dataset

The inner circle is the true label, and the outer circle is the predicted label. 1 is Infiltration and 0 is Benign, indicating that there is a distribution of mismatched labels.

B. Network Detection using Tree-ensemble model

The platform used in the research was Elastic Cloud, and a free-trial version was acquired through educational purposes. Due to the limitations imposed on the free access, the specifications of the clusters were substantially reduced. In Table VI, the specifications of the instances on Elastic Cloud using Google Cloud Platform are shown. On Elastic Cloud, there are pre-trained machine learning models such as lang_ident, elser_model_1, and E5 available for rapid deployment.The Tree_Ensemble model is used as the default method to train imported datasets. Due to the limitations of the free version of Elastic Cloud, adjusting the size of the dataset via cleaning was the only way to ensure enough RAM space for processing, even with autoscaling of different instances.

TABLE VI: Elastic Cloud Instances Specification

Name	*Type*	*RAM*	*Disk-Allocate*
gcp.es.master.n2.68x32x45-v2	Master	1GB	45GB
gcp.es.datahot.n2.68x10x45-v1	Data_hot	4GB	180GB
gcp.es.ml.n2.68x32x45-v1	AI-ML	4GB	180GB
gcp.es.datahot.n2.68x10x45-v1	Data_hot	4GB	180GB
gcp.enterprisesearch.n2.68x32x45-v1	Search	2GB	N/A
gcp.integrationsserver.n2.68x32x45-v3	Server	1GB	N/A
gcp.kibana.n2.68x32x45-v1	Stack	1GB	N/A

The UNR-IDD dataset after processing using the tree ensemble model showed similar experimental results compared to the original paper. Blackhole is 0.998 vs 0.98 when comparing the precision of tree ensemble on Elastic Cloud to the reference paper. All other labels have identical precision. Similarly, all recalls are within reasonable ranges within 1% standard deviation. In both cases, the normal label is identical. During the initial steps, the feature importance was determined to be Port Alive Duration and Packets Matched. In subsequent experiments, only these two features were used to train the UNR-IDD dataset. The results showed that only a mere 1.7% dropped in mean precision and the binary classification of Normal Vs. Attacks remained at 100%. The reduced features allowed the model to be trained quickly and lowered the demand for online storage and computational resources.

TABLE VII: Multi-Class Classification-Tree Ensemble (UNR-IDD)

Label	P	R
Blackhole	0.998	0.992
Diversion	0.99	0.97
Overflow	0.98	0.76
PortScan	0.91	0.94
TCP-SYN	0.91	0.92
Normal	1.0	1.0

The NSL-KDD dataset had different features compared to the benchmark-UNR-IDD; there was no overlapping feature found. For example, the important features were Port Alive Duration and Packets Matched; however, in the NSL-KDD dataset, the important features were duration, protocol, and level. Compared to the UNR-IDD dataset, despite the mean recall percentage is 82.22%, the standard deviation has a rather large spread. This is due to the recall percentage for labels such as Nmap and Ipsweep being close to one, and in contrast, Loadmodule and Imap were rather low at 0.727 and 0.444 respectively. The selected features might have contributed to the high precision and recall for a particular attack but not others with different characteristics. Overall, the binary classification achieved a precision and recall of 100% compared to the UNR-IDD dataset.

TABLE VIII: Multi-Class Classification-Tree Ensemble (NSL-KDD)

Label	P	R
Nmap	1	0.994
Perl	1	0.667
Portsweep	1	0.999
Buffer_overflow	1	0.933
Imap	1	0.727
Loadmodule	1	0.444
Ipsweep	1	0.999
Warezclient	1	0.996
Normal	1	1

The third dataset being used to the Tree Ensemble machine learning model was the CIC-IDS2018, which has more than 100,000 samples. Due to the limitations of the Elastic Cloud trial version, the sample size was trimmed to ¾ and ½ to determine the upper limits of the machine learning model. The upper limit was determined to be 50,000 samples, which was approximately 30% of the dataset. Unlike the first two datasets, the CIC-IDS2018 dataset focused on only two attacks: SSH-brute force and FTP-brute force. While the binary classification of Normal Vs. Attacks did not achieve a precision of one, it was nearly 99.999% accurate.

TABLE IX: Multi-Class Classification-Tree Ensemble (CIC-IDS2018)

Label	P	R
SSH-brute force	0.99995	1.0
FTP-brute force	0. 99995	1.0
Normal	0.99995	1.0

V. Conclusions and Future Work

The power of machine learning can effectively reduce the workload of mundane tasks involved in cyber malicious activities detection. In this study, two classic machine learning algorithms, DistilBERT and Tree Ensemble, were examined for their performance in processing three datasets: UNR-IDD, NSL-KDD, and CIC-IDS2018. The overall precision and recall for all three datasets from two machine learning models achieved more than 90% in the majority of cases. Most importantly, the binary classification of Normal Vs. Attacks were nearly 100% on the Tree Ensemble. In the case of DistilBERT on CIC-IDS2018, the rather low precision and recall is possibly due to the downsizing of the training set given the limitations of the Elastic Cloud hot storage. In terms of potential future work, this research has shed light on the use case of Tree Ensemble and DistilBERT to concurrently train all three datasets and streamline the source data ingestion pipeline. The reason is that all three datasets rely on different feature

selections to detect various attack vectors that may not be covered. Future research should expand on moving past the software limitations by investing in more cloud storage and computational power on the cloud in order to reap the benefits of cloud computing.

REFERENCES

[1] H. Almohannadi, I. Awan, J. Al Hamar, A. Cullen, J. P. Disso, and L. Armitage, "Cyber Threat Intelligence from Honeypot Data Using Elasticsearch," 2018 IEEE 32nd International Conference on Advanced Information Networking and Applications (AINA), Krakow, Poland, 2018, pp. 900-906, doi: 10.1109/AINA.2018.00132.

[2] K. A. N. Madhuvantha, M. H. Hussain, H. W. D. T. De Silva, U. I. D. Liyanage, L. Rupasinghe, and C. Liyanapathirana, "Autonomous Cyber AI for Anomaly Detection," 2021 3rd International Conference on Advancements in Computing (ICAC), Colombo, Sri Lanka, 2021, pp. 85-90, doi: 10.1109/ICAC54203.2021.9671203.

[3] A. Aloseel, H. He, C. Shaw, and M. A. Khan, "Analytical Review of Cybersecurity for Embedded Systems," IEEE Access, vol. 9, pp. 961-982, 2020.

[4] S. Siddiqui, M. S. Khan, K. Ferens, and W. Kinsner, "Detecting Advanced Persistent Threats Using Fractal Dimension-Based Machine Learning Classification," Proceedings of the 2016 ACM on International Workshop on Security and Privacy Analytics, pp. 64-69, 2016.

[5] T. Bodström and T. Hämäläinen, "A Novel Deep Learning Stack for APT Detection," Applied Sciences, vol. 9, no. 6, pp. 1055, 2019.

[6] C. Yin, Y. Zhu, J. Fei, and X. He, "A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks," IEEE.

[7] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A Deep Learning Approach to Network Intrusion Detection," IEEE Transactions on Emerging Topics in Computational Intelligence, vol. 2, no. 1, pp. 41-50, 2018, doi: 10.1109/TETCI.2017.2772792.

[8] [1] T. Das, O. A. Hamdan, R. M. Shukla, S. Sengupta, and E. Arslan, "UNR-IDD: Intrusion Detection Dataset using Network Port Statistics," 2023 IEEE 20th Consumer Communications & Networking Conference (CCNC), Las Vegas, NV, USA, 2023, pp. 497-500, doi: 10.1109/CCNC51644.2023.10059640.

[9] M. Haggag, M. M. Tantawy and M. M. S. El-Soudani, "Implementing a Deep Learning Model for Intrusion Detection on Apache Spark Platform," in IEEE Access, vol. 8, pp. 163660-163672, 2020, doi: 10.1109/ACCESS.2020.3019931

[10] Sharafaldin, Iman et al. “Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization.” International Conference on Information Systems Security and Privacy (2018).

[11] Sanh, V., 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. arXiv preprint arXiv:1910.01108.

[12]https://www.elastic.co/guide/en/elasticsearch/reference/current/put-trained-models.html

[13] Friedman, J. H (2001). Greedy function approximation: A gradient boosting machine. The Annals of Statistics, 29(5), 1189-1232. https://doi.org/10.1213/aos/1013203451

[14] Breiman, L. Random Forests. Machine Learning 45, 5–32 (2001). https://doi.org/10.1023/A:1010933404324

[15] Cortes, C., Vapnik, V. Support-vector networks. *Mach Learn* **20**, 273–297 (1995). https://doi.org/10.1007/BF00994018

[16] Rumelhart, D., Hinton, G. & Williams, R. Learning representations by back-propagating errors. *Nature* **323**, 533–536 (1986). https://doi.org/10.1038/323533a0

[17] Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780.