Artificial Intelligence and its Usage in Cybersecurity

A General Overview of Artificial Intelligence and Machine Learning Applications in Cybersecurity Defenses.

Justin Linder
Network and Computer Security
SUNY Polytechnic Institute
Utica, NY
linderjm@sunypoly.edu

Abstract

The purpose of this paper is to introduce the key fundamentals of Artificial Intelligence (AI), including Machine Learning (ML) and Deep Learning (DL) as core components. The advantages of using neural network architectures in DL are discussed throughout the paper. The paper highlights key ML/DL techniques such as clustering and classification, explaining their roles in detecting vulnerabilities, automating repetitive tasks, and enhancing threat detection through pattern recognition and anomaly detection. This research paper demonstrates the growing importance of AI in defending against evolving cyber threats due to the increasing complexity of IT infrastructures, expanding attack surfaces, and significant amounts of data in circulation today. This paper also examines Generative AI, Large Language Models (LLMs), and chatbots, then discusses their use in log analysis, vulnerability scanning, and intrusion detection. I will explain the advantages of AI-driven cybersecurity and will primarily focus on AI-based solutions for cybersecurity defenses while also addressing adversarial AI attacks toward the end.

I. INTRODUCTION

As of October 2024, there are currently 5.52 billion internet users, approximately 67.5 percent of the global population [40]. More sophisticated cyber-attacks, such as zero-day attacks, are being deployed at a higher rate than signatures to detect these new cyber-attacks can be created. AI assists in detecting vulnerabilities or events that cybersecurity professionals could have missed by expanding threat detection capabilities in detecting abnormal behaviors and patterns [11]. Techniques like Natural Language Processing (NLP), image recognition, computer vision, and others enable AI systems to detect and respond to high volumes of common and complex threats in real-time [16]. AI-driven cybersecurity solutions aim to significantly reduce human error and automate security tasks that may be time-consuming or repetitive, such as log analysis [33]. This allows companies to conserve resources and money on cybersecurity defenses compared to AI solutions that aren't AI-based. Government agencies are increasingly adopting AI-driven cyber-defense solutions to reduce the likelihood of data breaches involving classified information while simultaneously reducing the use of taxpayer funds. Although AI systems may have vulnerabilities, such as being prone to adversarial AI attacks, their advantages outweigh these risks [11]. AI also plays a vital role in detecting insider threats, which are often complex risks to mitigate, constantly monitoring user activity and analyzing unusual behavioral patterns.

This research paper is suited for individuals with limited knowledge of AI who want to be introduced to its fundamentals and applications in cybersecurity. As a cybersecurity student, I understand that AI is increasingly being integrated into my field of study and potential career. Before starting this research, I had minimal knowledge of how AI and ML function. Therefore, I also did not understand the technical side of AI usage in cybersecurity. This motivated me to learn more about AI and its role in cybersecurity. I was also encouraged to educate classmates, peers, companies, and others interested in being introduced to this topic. I aim to strengthen my background in cybersecurity with this newfound knowledge. I also advocate for individuals and companies to adopt AI-driven cybersecurity solutions to stay ahead of the curve with evolving cyber-attacks, especially phishing. More companies rely on scalable AI solutions for cyber defenses and other business necessities.

The paper starts by giving a brief introduction to cybersecurity in Section 2. Section 3 presents AI and the core subsets of AI that will be discussed in this paper. Section 4 discusses the background in ML, including supervised and unsupervised learning, clustering, clustering algorithms, classification, and classification algorithms. Section 5 provides simple examples of ML techniques used in cybersecurity defenses. Section 6 introduces DL and neural network architectures. Section 7 examines the application of ML and DL techniques used in cybersecurity defenses, particularly in anomaly detection, intrusion detection, pattern recognition, malware analysis, and network traffic monitoring and detection. Section 8 discusses the nature of generative AI, including LLMs and chatbots, and their implications. Section 9 discusses the project, including difficulties faced, lessons learned, and what would have been done differently. Section 10 concludes the end of the paper with a short overview of the research.

II. BACKGROUND: CYBERSECURITY

Cybersecurity defenses are best represented by the CIA triad, which ensures confidentiality, integrity, and availability. Confidentiality means only authorized users can access sensitive information or data, preventing unauthorized access. Integrity is when data remains untampered and in its original, intended form. Availability is when systems and data are readily available to authorized users. Defensive measures are applied to mobile devices, data and information, IoT devices, cloud environments, critical infrastructure, networks and network devices, and applications [26].

Common cybersecurity threats include malware such as viruses, trojans, worms, rootkits, and more. Social engineering attacks, like phishing, gather unethical information about individuals. Ransomware compromises user systems and data, and then attackers demand compensation for restored access. Distributed Denial of Service (DDoS) attacks render systems unavailable by overwhelming them with traffic. Insider threats arise from individuals within an organization who intentionally or unintentionally pose vulnerabilities. Advanced Persistent Threats (APTs), often involving well-funded organizations like government agencies, are sophisticated/targeted attacks designed to maintain access to compromised systems [26].
While cybersecurity includes both offensive and defensive strategies, as mentioned before, this paper focuses on the defensive side. Cybersecurity vulnerabilities include outdated software like firewalls, user-end errors, poor coding techniques, system misconfigurations, validation failures, and other weaknesses that attackers can exploit.

III. BACKGROUND: INTRODUCTION TO ARTIFICIAL INTELLIGENCE AND SUBSETS.

AI is a broad term that describes tasks/predictions performed by machines that usually need human intelligence. AI branches off into several subsets or categories. ML is the primary building block of AI that can function independently or in conjunction with other AI systems. ML uses statistical learning algorithms to analyze data and form models. It uses various algorithms and processes to learn from data and predict future results. While ML is technically a subset of AI, the two terms are often used interchangeably. This paper will primarily focus on the ML subset of AI. ML itself has subsets such as Deep Learning (DL), a newer and more advanced subset of ML. DL uses models with multiple layers and neural networks to analyze and learn from data. DL and neural networks are terms sometimes used interchangeably with AI and ML [2], [3]. They both are technically part of the ML subset. Since ML is a broad category, references to ML may be referring to its subsets, like DL. To differentiate between traditional ML and DL, some authors refer to traditional ML as Shallow Learning (SL) [18]. Figure 1 illustrates the primary subsets of AI discussed in this paper. Generative AI and Large Language Models (LLMs), which are subsets of DL, include popular applications/chatbots like ChatGPT and Gemini.

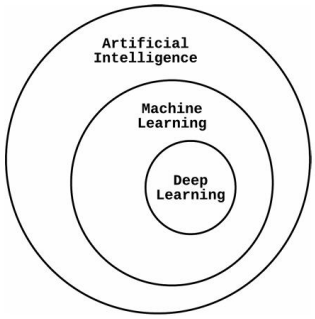

Figure 1: Visualization of AI and subsets [2].

IV. BACKGROUND: MACHINE LEARNING.

A. Supervised and unsupervised learning

There are two main fields in ML, supervised learning and unsupervised learning. Supervised learning means that an algorithm has access to both the samples and the labels within a dataset. The labels represent values or answers to the outcomes that ML models are trained to predict or determine. Classification is a type of supervised learning. Unsupervised learning is when algorithms only have access to samples and don't have any labels. Clustering is an example of unsupervised learning. Semi-supervised learning is when supervised and unsupervised learning are combined so that unsupervised learning of the data can occur first to assume how the data is distributed which is then applied to the supervised learning process. Classification algorithms will use labeled samples in vectors to build a model that will predict labels for new unlabeled samples [5], [7].

B. What is clustering?

Clustering is the process of organizing data so that samples are grouped into clusters based on similar features and characteristics. This is a way to measure how close these samples are when thinking about it graphically [2]. An example of visualizing two clusters graphically is shown in Figure 2. Data sets of any size can be grouped into clusters to display anomalies/outliers, particularly useful to cybersecurity professionals. A simple example of clustering is using the "group by" SQL statement when grouping similar features like the same IP address. [1]. Clustering plays a crucial role when data is unlabeled or ungrouped. Hard clustering is when samples specifically are part of a certain cluster, while soft clustering is based on the probability that a sample is part of a certain cluster [22].

Samples in ML are also commonly referred to as data points, items, instances, examples, and cases. I have seen all these names used in ML when conducting research, which can be confusing at first glance. I will primarily use

the terms samples or data points in this paper. These terms mean the same thing, but data points are helpful when describing graphing scenarios since they can be visualized on a graph.

Clustering is an important part of data mining since it groups similar data for databases. There are a variety of ML algorithms for clustering. Some clustering algorithms that exist today are DBSCAN, k-means, Gaussian Mixture, C-Means fuzzy, Deterministic Annealing EM, Hierarchical, Reconstructive, Partitional, and many more [5], [6]. The top three most used clustering algorithms in ML are k-means, hierarchical, and DBSCAN [22].

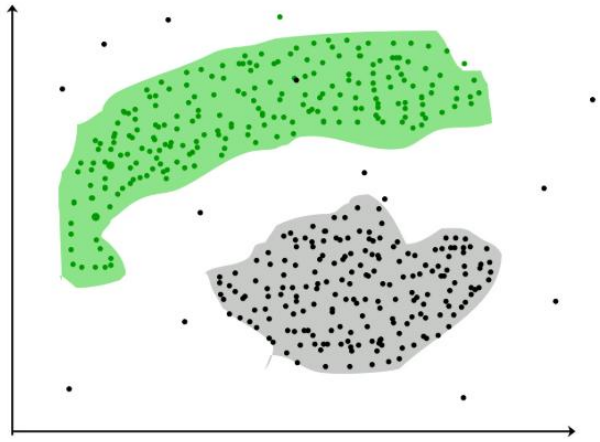

Figure 2: Two different shaped clusters [22].

C. Basic clustering steps:

There are generally four main steps involved in the clustering process.

- First, provide data to collect samples from, and these samples should portray characteristics of the data set as a whole to get accurate results [1].
- Then, extract the most relevant data features from the samples [1].
- Format every sample in a vector so its features are in rows and columns. When data features hold any value within a range, they are "continuous," and in a limited range, are known as "categorical." However, they are usually used in a combination of continuous and categorical. A process known as normalization is required when data features are formatted in a continuous manner so the values of the data features are distributed equally [1].
- The final step in clustering is picking a clustering algorithm within an application or library [1].

D. k-means

The k-means is the most popular type of clustering algorithm. It should be used when dealing with a smaller amount of data since the larger the data sets, the longer it takes to cluster [23]. This is a centroid-based clustering algorithm, meaning samples are grouped based on proximity. Euclidean distance is one method of calculating the proximity of two different samples. The quantity of clusters that k-means will use is decided ahead of time [22].

First, the sample vectors are sorted within the "feature space" via certain coordinates or locations based on the sample's features. A shorter distance between the points indicates a higher probability that the vectors are part of the same cluster, and a longer distance means the opposite after using the first three main clustering steps mentioned above. A hyperparameter is required, and the algorithm will make several clusters. K-means then choose random vectors and give each vector a specific coordinate in the feature space known as "centroids." The goal is to ensure that every sample is part of a cluster that makes the shortest distance from that sample to the cluster's centroid [1], [2].

E. DBSCAN

DBSCAN is a type of density-based clustering that distinguishes groups based on the density of the samples which means how compressed they are together in a tight space. The clusters normally have abnormal shapes and vary in size [22]. It solves the problems of clusters with different densities, but the algorithm requires a cluster size. Clusters are created in the areas of the feature space that have the most vector density. DBSCAN does not need an input for the number of clusters to create as k-means does. Instead, it requires two hyperparameters. One hyperparameter is called the epsilon. The epsilon provides the "epsilon neighborhood", which is how big the radius is. This epsilon neighborhood contains the samples used to decide which cluster they are part of. The second hyperparameter is a value that represents each epsilon neighborhood's minimum amount of data points. Every data point is given a point category of either noise, border, or core point. A cluster ID is assigned for each cluster to distinguish them from one another [1], [2], [6].

F. Hierarchical

Hierarchical clustering algorithms are a group of clustering algorithms that create clusters within bigger clusters by either combining clusters or separating them. It is called "hierarchical" because there is a tree-based hierarchy. It starts at the "root," which is a cluster that gathers samples, while the leaves are clusters that each hold one sample [8]. The structure's height can be adjusted to get a certain amount of clusters. The leaves with one cluster that are closer together are more similar to one another. Samples either start as one large cluster and then break up into

smaller clusters, which is known as "divisive". The other option is that each sample starts as its own cluster and then combines into one large cluster, which is known as "Agglomerative" [22]. A linkage criteria determines how to calculate the distance between each cluster. Examples of linkage criteria include average linkage, complete linkage, single linkage, centroid linkage, and Ward's method [5], [8].

G. What is classification?

Classification is when ML models use algorithms, also known as classifiers, to predict labels for each sample. Labels are how data samples get organized or sorted into different classes [25]. A binary classification means a sample can only belong to one of two classes. A multinomial or multiclass classification refers to a sample belonging to multiple classes, but cannot exist in those classes simultaneously [1], [25]. A multilabel classification is when a sample can be part of multiple classes at the same time. A classification model can produce a discrete prediction when each sample gets assigned a class label. It can produce a continuous prediction when its samples get a class label based on probability scores [25]. Classification algorithms such as decision trees, random forests, and logistic regression are some of the most popular ones used today [1]. Other popular ones are Naïve Bayes, Support Vector Machine (SVM), and k-nearest neighbors [25].

H. Steps for Classification

Enough data must be collected to create accurate models. After the data is collected and labeled, it needs to be trained, validated, and tested within separate sets or subsets. Classification usually has four steps or phases [1], but some combine them into two steps for learning and classification [25].

- The first is the training process, which is the most important subset of classification as it uses a majority of the samples. A model is created using a classifying algorithm for training that uses a vector that is filled with the labels and a matrix that is filled with the samples. Multiple models can be trained with various classification algorithms and hyperparameters to see which models have better accuracy and performance [1].

- The next phase is to use a validation dataset to test the model's accuracy [1].

- Then, test any data that wasn't part of the training and validation phases. This phase will determine the model's accuracy [1].

- If the model is accurate, it is used to predict the class of the unlabeled data [1].

I. Logistic Regression

Logistic regression is a probability algorithm and type of supervised learning that was created based on linear regression [25]. It is commonly used for classification because it's very efficient in dealing with plenty of features and sample sizes. Feature space is used in logistic regression algorithms to separate vectors in different classes with straight lines. A model using binary classification will use a straight line to divide the feature space into two areas that represent two separate classes. That line, also known as a decision boundary, is shown in Figure 3, which is created by the "solver functions" [1], [2].

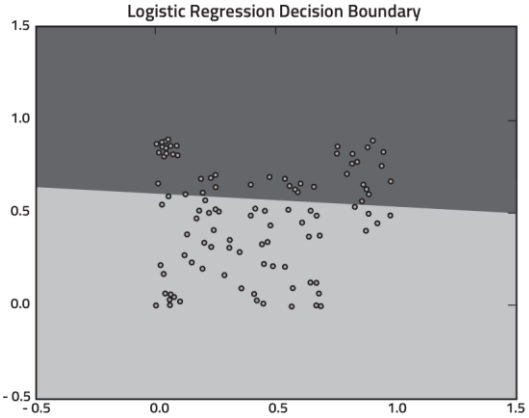

Figure 3: Decision Boundary is the line in the middle [1].

Regression weights are multiplied by the feature values of each feature to decide their contribution to the class that the vector belongs to. This means that the higher the regression weight is, the larger the contribution. Logistic regression utilizes a process called "regularization" to prevent distortion from large values, similar to the concept of applying normalization when clustering. Regression weights predict the vector's class, and the outcomes are validated for accuracy [1]. Figure 4 shows a visual example of what two different classes look like for labeled vectors.

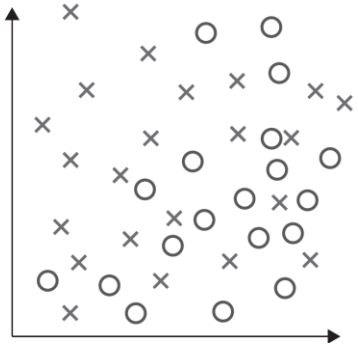
Figure 4: Each point depicts a vector; two classes are represented by X's and O's [1].

J. Decision Tree

Decision Trees are another commonly used classification algorithm that is a form of supervised learning. They are simple to understand but susceptible to overfitting, which can be overcome using random forests, as discussed in the next section. Another weakness is they are susceptible to biases. They predict values by using decision rules [10]. For instance, if-then-else statements are used when programming a binary decision tree. These algorithms vary because if a class label contains continuous values, then decision tree regression will be used, and if it contains categorical values then decision tree classification will be used.

Decision trees predict classes by using branches, leaf nodes, and root nodes. Hence the name decision tree. In this case, the root node is at the top of the tree and consists of the samples that will be trained. The root node then splits the samples into subsets, called branches, based on the samples' feature values. These branches will then split into leaf nodes, and then into terminal nodes which are where the training samples are assigned to a class. More branches can be used before splitting into leaf nodes if needed, and they will continue to split into different subsets until they all are part of one class [1].

Another term used with a decision tree is a parent node, which refers to a decision point, and the child nodes refer to nodes that form after a subset split. Underneath a decision point, the left child node normally represents true and the right child represents false. Decision trees may be simple to understand, but they are not as efficient and accurate as other classification algorithms like logistic regression since they tend to be more complex than they need to be when dealing with certain relationships [2].

K. Random Forests

Random Forests consist of multiple groups of decision trees. Each tree in the random forest is an individual classifier or classification algorithm that starts at the root and goes down the tree to check for each condition [18]. A random forest creates various decision trees during its training, as shown in Figure 5.

It is referred to as a "random" forest because it uses completely random samples from the data set to train on for each decision tree. Also, each decision tree chooses random feature conditions at each subset to separate samples. This allows the model to make more accurate predictions since each decision tree makes independent decisions and has unique feature conditions for each subset. Each decision tree's output prediction is combined and either averaged or voted on by each decision tree to form a final prediction [24].

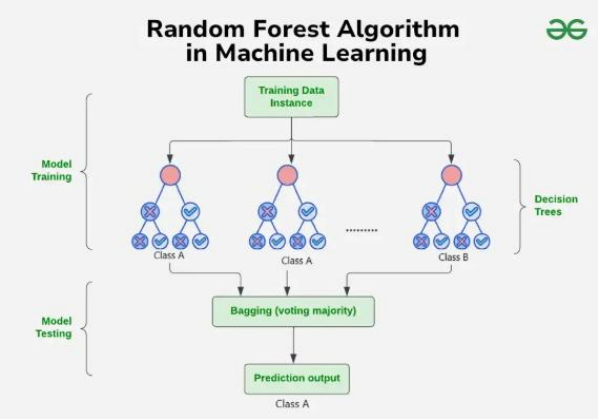

Figure 5: Shows three decision trees in a random forest [24].

V. BASIC EXAMPLES OF MACHINE LEARNING IN CYBERSECURITY DEFENSES.

A. Purpose of clustering in cybersecurity:

Clustering is often used together with different methods like pattern recognition after the features get extracted and normalized. Clustering allows security professionals to observe the relationships between cluster members that are hidden within vast amounts of network and system data because the characteristics or features that define cluster members appear to be similar or distinct [1]. These data features amongst cluster members that are deemed irrelevant or abnormal in the clustering process are also known to be "anomalies". ML detects anomalies as a vital part of the pattern recognition process for unusual traffic or log patterns in security [4]. Clustering is used to detect intrusions, data breaches, malware, and network analysis. This will be discussed in more detail throughout the paper. Pattern recognition is more commonly used in supervised learning tasks, especially when enough unbiased labeled data contains distinguishable samples. These techniques are often used together in semi-supervised learning to perform network analysis when detecting and preventing phishing attacks, spam, DoS/DDoS, botnets, spoofing, data breaches, pivoting, and access controls [2].

B. Monitoring network traffic with clustering

Clustering can group network traffic anomalies like increased traffic volumes. Using cluster analysis on logs for HTTP servers can show unusual user and traffic patterns, which may include abnormal user request information, times, dates, IP addresses, usernames, errors, server responses, authentication patterns such as anomalous locations, and more [1]. This abnormal traffic behavior can be indicated by malicious activity like botnet communications or a DDoS attack.

Specifically, IP addresses can be clustered via response codes and verbs within the HTTP logs using clustering algorithms like k-means, hierarchical, DBSCAN, etc. This will cluster IP addresses based on similar HTTP response

codes and HTTP verbs that are in the HTTP logs. Then, the clusters can be inspected and validated to determine if the anomalous or unusual network activity in one cluster has malicious intent while the expected network activity will be in a separate cluster. It is also common to continue to optimize cluster results by adjusting or increasing the number of clusters to produce more sufficient results. If an IP address is deemed to be malicious and the cluster, it is in has a high "Silhouette" score then that means the other IP addresses in the same cluster are very similar to the malicious IP address and are most likely malicious as well. The network activity for the malicious IP addresses can be further inspected by observing the HTTP server logs for those specific addresses [1].

C. Clustering and classification combined

When clustering and classification are used together, after clustering IP addresses from network traffic logs, the clusters of potentially malicious IP addresses can be inspected by a cybersecurity professional to verify if they are malicious. Then they can be classified by labeling if they are malicious or not for binomial classification, or by using multinomial classification to specify different types of malicious traffic [1]. Using the two together will utilize the strengths of detecting unlabeled anomalies with clustering while focusing on the known threats that are labeled with classification algorithms. This is a basic example of standard ML since it requires some type of human intervention, unlike DL models.

D. Decision Tree example in cybersecurity

A basic example of using a decision tree could involve testing if websites are malicious or not. Malicious and non-malicious websites are the roots. The website will start by spitting into branches based on the date of the website's domain. For instance, websites can split into branches based on whether the domain is at least a specific number of days, weeks, or months. The other half would be if the domain is not that amount of time. Then do another split to test if that category is also part of a list of common websites currently being used. For the subset that doesn't reach the minimum, test if the samples have a certain quantity of IP addresses. If either test has the minimum amount of IP addresses being a common website, then these samples are classified as non-malicious [1]. That completes the first half of the decision tree.

Lastly, test the data samples with the minimum amount of IP addresses automatically generated by the website. If that test is positive, those samples are assigned to the malicious class, and when negative the samples are assigned to the non-malicious class. Test if the list of common websites currently being used has samples with a domain owner flagged with a suspicious report. The samples that test positive for that will be assigned to the malicious class, and the samples that test negative will be assigned to the non-malicious class [1]. This is a general example, and there can be more decision rules for testing malicious websites that will produce more accurate results. A random forest could be used instead to test randomized rulesets.

E. Classifying spam

A simple example of classification in cybersecurity is the binary classification of emails as either spam or not spam. This involves using a data set consisting of pre-labeled spam and non-spam email samples. A classification algorithm is trained on this dataset to learn the features that distinguish spam from non-spam. Once trained, the algorithm can predict whether unlabeled samples should be classified as spam or not spam based on the learned features. In binary classification, the model assigns each sample to one of two classes to determine which labels are assigned. This binary classification example for spam is also an instance where pattern recognition is used [1]. This is a basic example of classification and a simple model like this will most likely result in false positives, which can be lessened by using more complex ML or DL models.

F. Classifying botnet traffic

C2 servers are capable of controlling botnets which can infect multiple PCs that are normally used to launch DDoS attacks. Classification is one of the methods that could be used for detecting various types of botnets. Linear regression and decision trees are specific examples of classification with ML mentioned earlier that would be able to do this. The first thing that needs to happen is the collection of samples. These samples are obtained by sending HTTP requests to all the web servers you want to test to provide us with files and response codes that can be stored in a pre-vector file. These features should be extracted from the pre-vectors file and placed in a vector. Now that this data has been collected, we can train a model, assess its accuracy, test, and validate it [1]. Honeypots or honeynets are commonly used when analyzing botnet traffic. Some other traffic samples that are normally collected for feature extraction include destination and source IP addresses, ports, protocols, time patterns, sizes of packets, etc.

VI. BACKGROUND: DEEP LEARNING.

A. What is Deep Learning?

Deep Learning (DL) is an advanced subset of ML that takes more time and computational resources to train models since more data is used in the training process. It requires more processing power, which normally requires Graphics Processing Units (GPUs) [30]. It is more effective than ML when dealing with complex tasks such as machine translation, image recognition, speech recognition, Natural Language Processing (NLP), and computer vision [9], [16].

NLP allows for the creation and understanding of human languages so they can create texts and recognize speech. Computer vision allows machines to detect and recognize data visually, classify images, and understand certain features in images [30]. DL was derived from Artificial Neural Networks (ANNs) in the year 2006 [27]. DL uses supervised learning, unsupervised learning, and reinforcement learning techniques. Backpropagation in supervised learning allows a model to calculate a loss function based on the discrepancies between the actual and predicted output [28], [30]. This improves the model's accuracy by reducing the probability of errors in the future. DL is also used by Generative AI and Large Language Models (LLMs), which will be discussed in a later section.

B. *Process*

DL uses neural networks in the training process. Neural networks are algorithms created to represent how the human brain works. DL can process significantly more data than ML because it has multiple layers for distributing learning and processing data at each layer. A simple neural network, such as an ANN, consists of three layers. Each layer consists of multiple nodes connected to each other, and these nodes are referred to as "neurons," which is how the human brain thinks. The first layer is the input layer responsible for receiving and sending data inputs to the hidden layer [2], [16]. Neural networks may consist of multiple "hidden" layers called Deep Neural Networks (DNNs). They are called shallow neural networks when they consist of only one hidden layer [32]. Complex problems use more hidden layers to solve them. A hidden layer receives the data inputs, gives each data value a weight, assigns a bias value, and uses the specific hidden layer's activation function which performs a calculation unique to that hidden layer. The last layer is the output layer. This layer has activation functions as well, which help determine the model's final prediction and provide an output of the results. The results in the output layer are used in a loss function to decide whether those results can be improved in the training process before they become the final output [2], [11], [16]. A visual of a Neural Network and its layers is shown in Figure 6.

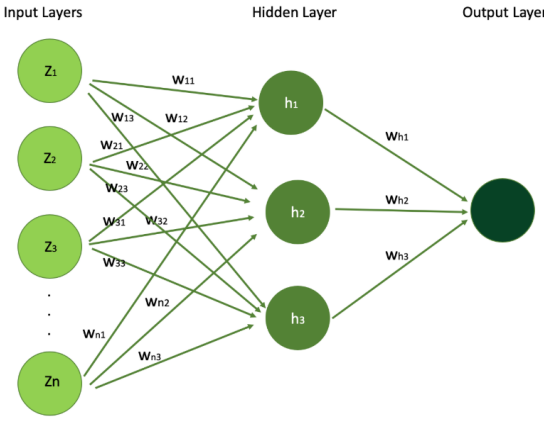

Figure 6: Neural Network Architecture [28].

C. *Types of Neural Networks*

Common Neural Networks used in DL are ANNs, Recurrent Neural Networks (RNNs), Convolution Neural Networks (CNNs), and Autoencoders (AE). ANNs were the first type of DL architecture that inspired the first design based on the human brain. Feedforward Neural Networks (FNNs) can only pass data from nodes in the input layer to the output layer [16]. FNNs are commonly used for NLP, speech recognition, and classifying images [30].

RNNs use a feedback loop structure to pass data backward from nodes in the output layer to the input layer [32], [34]. These are different from FNNs because they hold memory for relationships between the current input and previous inputs by sending copies of the values for each sample in the output layer back to the input layer [45]. RNNs assist in sequential data and time-series data [9], [27], [31]. They are applied to NLP, speech recognition, captioning images, and language translation [30], [31]. Specific examples of RNN applications are Apple's Siri, Google Translate, and voice features used to search the web [31]. Long Short-Term Memory (LSTM) is a type of RNN that has memory cells for storing data that are used when dealing with problems that require long-term data storage [19].

CNNs are FNNs that automatically learn features, like colors or edges, and find patterns in images. They have convolutional, pooling, and fully connected layers, and the number of layers can range into the thousands. Therefore, they require a lot of processing power and are highly effective when dealing with audio and imaging data [30], [31]. The convolutional layer extracts features from the input via a convolution filter. A pooling layer has feature maps made by the convolution layer and reduces its dimensions, distributing the information into regions representing each feature. The fully connected layer maps each feature to an output prediction [34]. CNNs are mainly used for image classification, image recognition, NLP, and computer vision [9], [27], [31]. This provides image and video recognition that could be used for facial recognition, analyzing medical images, image segmentation, detecting objects, and more [27], [30], [31].

AE uses unsupervised learning to reduce multidimensional data. Using the encoder, they use dimension reduction by compressing multidimensional data inputs into data with low-dimensional feature space. Samples from the data are then reconstructed with the decoder to retrieve the data and compare it with the original data to calculate reconstruction errors using the reconstructive loss function [27], [29].

VII. MACHINE LEARNING AND DEEP LEARNING IN CYBERSECURITY DEFENSES.

A. *Deep Learning in Cybersecurity*

DL is used in Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS). IDS will be discussed in more detail in the Anomaly Detection section, but it is important to note that DL decreases the number of false positives encountered when using ML. CNNs and RRNs are specific neural networks used in DL when dealing with an IDS or IPS. DL can detect more complex and sophisticated attacks via Malware analysis compared to ML or certain types of firewalls. There are many types of social engineering attacks, such as spam. DL uses NLP to detect more sophisticated versions of this based on patterns involving types of human languages. DL is being used by many companies for detecting internal traffic and behaviors by its employees for potential threats from inside. Analyzing network traffic with an ANN detects various network-based attacks such as Denial of Service or SQL injection [17].

It is being implemented in cybersecurity defenses because it has demonstrated advanced techniques compared to conventional rule-based, signature-based, and ML ones. RNNs are popular DL models because cybersecurity problems tend to be with time-series data. Restricted Boltzmann Machines (RBMs) are commonly used in cyber-defenses, mostly with IDSs and malware detection and classification. DL is also better at identifying file types, network traffic, spam and Domain Generation Algorithms (DGA), insider threats, Border Gateway Protocol (BGP) anomaly detection, verifying keystrokes, and injections with false data. DNNs are used for IDSs in Software-Defined Networking (SDN) [19]. DL offers better performance than traditional ML because it can use several billion parameters, has several layers for processing, and can process multidimensional data that includes many data types [45]. Another advantage of DL methods, such as LSTM, is its ability to capture dependencies of temporal and spatial data [38].

AE is used in cybersecurity for anomaly detection by using anomaly scores provided by the reconstructive loss function, which will produce significant quantities of reconstruction errors when reconstructing anomalies. When AE is used in hybrid models LSTM-AE or CNN-AE detects anomalies in additional data types such as graphical, sequential, video, and image [29]. Autoencoders can classify vast quantities of data [31]. Multiple hidden layers in AE provide more accurate email spam detection predictions compared to other DL models trained on a similar data set [19]. Other common neural network architectures examined for cybersecurity tasks included CNN-RNN, RBM-AE, RBM-RNN, and Generative Adversarial Networks (GANs) [19]. LSTM is often used instead of RNN since it is a far superior architecture.

GANs will be discussed in more detail in the Generative AI and LLMs section. DL was studied using certain data sets for cybersecurity-based models. These include NSL-KDD, KDDCUP 1999, Internal Microsoft dataset, and many more [19]. Data sets are custom-made as well.

Source	Topic	Key findings
[18]	ML and DL applications in cyber-defenses	Advantages of intrusion detection with DL for DGAs and botnets include malware detection for polymorphic malware.
[21]	Detecting advanced malware using DL	ScaleMalNet for zero-day attacks, CNN-LSTM performed best with 96.3 percent accuracy.
[29]	Anomaly detection with DL	End-to-end anomaly score learning for anomaly detection.
[38]	Network traffic analysis and detection with DL	Compares the accuracy of various DL models when classifying and predicting traffic patterns. LSTM-CNN showed the best performance, with botnet detection having 99 percent accuracy.

Figure 7: This table identifies the main topic and key findings for some of the resources used in the research for ML/DL methods used in cybersecurity defenses.

B. Anomaly detection and IDS

In cybersecurity, it is essential to recognize anomalies, which are identified because they are either unexpected or out of the ordinary, hence why an anomaly is commonly referred to as an “outlier. An anomaly can be identified and later determined as a false positive, meaning it was anomalous but not malicious. However, false positives are better than false negatives regarding cybersecurity, allowing human intervention to assist in the detection, prevention, and training processes when necessary. Anomalies can be categorized into three groups. The first group consists of “point anomalies” which is a specific data sample or data point flagged for being abnormal compared to a set of normal data samples, such as a significant increase in traffic. Another group is called “collective anomalies”. This is a category of anomalies where multiple data samples in a cluster are all identified as anomalies. “Contextual anomalies” are specific data samples identified as anomalies based on a certain feature or “context” rather than the data sample entirely. Anomalies are useful for early and real-time detection of data breaches, malicious activity, data integrity, Advanced Persistent Threats, and more [13].

Anomalies can be detected within multiple layers of the OSI model. There are several types of anomalies such as geo-location, services, specific times or time intervals, volumes of data, CPU usage, or traffic patterns that can be detected if they are deemed to be outliers. IP addresses, protocols, ports, and other network traffic features can be anomalous based on the ML/DL model. Models tend to be more accurate when detecting anomalies when they are trained with data consisting of many features with

unfamiliar parameters. A “null hypothesis” refers to the baseline of network behavior that is normally expected, and it is also used to calculate anomaly scores. Higher anomaly scores portray a higher probability that a data point or feature is anomalous. Intrusion detection Systems can detect anomalies that may go undetected with signature-based processes [12]. Supervised Learning techniques like semi-supervised anomaly detection are better for learning normality in data sets. Deep anomaly detection uses feature mapping to calculate these anomaly scores for each sample. This allows for anomaly detection of non-linear relationships and large quantities of multidimensional and complex data types. Multidimensional data includes multiple data types, such as sequential, temporal, audio, video, graphing, and image data [29]. DL models can automatically extract and learn features. End-to-end optimization is a process that allows a model to detect anomalies while simultaneously learning. This only requires some samples to be labeled no matter the type of data, which is another benefit of deep anomaly detection over standard ML techniques [29].

Anomaly detection is the core of most Intrusion Detection Systems. There are host-based, network-based, and application-based versions of Intrusion Detection Systems that can detect various anomalies. Some examples of potentially malicious anomalies include certain accounts that have been created or are being used, currently running, processes initiated to run startup and background processes, kernel modules, DNS lookups, temp files or directories, registry changes, and much more. Intrusion Detection Systems normally use Deep Packet Inspection to inspect all the data encapsulated in a packet from multiple layers. This includes both the header and payload. Metadata is additional information about the data that is included within the header. Deep Packet Inspection can detect anomalies that indicate data extraction, cross-site scripting, SQL injections, spam, adware, ransomware, and other types of malware detection [2]. Machine Learning can perform anomaly detection by using a supervised learning technique called “forecasting” to predict data values based on the time series and previous data. A forecast model can be trained to learn past data to predict future data. Neural networks can take forecasting a step further by learning automatically, dealing with larger data sets, and utilizing more complex pattern recognition [2]. DL is also better at detecting DGAs and botnets with IDS. A DGA is used to create several domain and host names automatically for command-and-control servers by implementing NLP [18].

RNNs and CNNs combined with LSTM in a hybrid model have been shown to have higher detection rates with network anomalies and more accurate classifications. Hybrid algorithms have higher accuracy rates, as demonstrated in studies. An RNN model only had an 87 percent accuracy rate compared to a hybrid model of RNN, LSTM, and a Gated Recurrent Unit (GRU) which had a 99.1 percent accuracy rate in IDS classifications [35]. DL-based methods for anomaly detection offer better performance than ML methods when applied in the real world. For instance, an NSLKDD data set was used to train ML models using decision trees, SVM, and random forest classifiers, which were compared to DL methods using AE, CNN, and RNN [38]. The DL methods displayed higher accuracy percentages in classifications than the ML methods. This was also tested with a CNN trained on KDDCUP1999, which produced 98.4 percent accuracy, and DARPA1998, which had 97.9 percent accuracy [38].

C. Pattern Recognition

Pattern recognition is used to detect similar patterns in data based on characteristics or features. This is useful when these data features may appear to be hidden. Pattern recognition relies on a model that is trained on unbiased and large data sets to make predictions. While anomaly detection can detect an unlimited number of outliers. Even when the outliers aren’t contained in the data sets used to train the model. In cybersecurity practices, pattern recognition is a technique involved in botnet, malware, spam, and fraud detection. A brief example of fraud detection would be using pattern recognition to recognize certain locations and match a client’s spending history, as well as patterns of high amounts of money spent [2].

The data in Common Vulnerabilities and Exposures (CVEs) and Common Weakness Enumeration (CWE) documents can be used to train ML/DL models to classify common vulnerabilities in source code. Static Application Security Testing (SAST) is a technology that scans through each line of code to detect vulnerabilities in Java and .NET applications and other common frameworks and programming languages. Control and data flow vectors are used by ML algorithms to recognize patterns in CWE [36]. Dynamic Application Security Testing (DAST) uses ML to automate code scanning to detect vulnerabilities and attacks in website applications. It can also be used in a Next-Generation Firewall (NGFW). Burp Scanner is a DAST commonly used today. HTTP requests are sent to web servers to collect information about potential vulnerabilities. ML extracts temporal features from multiple layers in the OSI model and combines multiple samples over a specific time window called “tumbling windows" [37].

When Pattern recognition is used together with anomaly detection, AI/ML models can continue to be retrained to decrease the number of false positives in detection systems and learn from past data to establish more accurate results in the future. For example, a cloud-based platform called “CrowdStrike Falcon” uses AI/ML pattern recognition and anomaly detection techniques to find what they call “indicators of compromise”. They refer to these techniques as “behavioral analysis”. This platform is capable of detecting data breaches and other behaviors or activities that appear to be malicious. It can detect identity-based attacks like stolen user credentials or unauthorized users who are using someone else’s credentials. CrowdStrike Falcon can implement a behavior baseline within an authentication

system like Active Directory by using pattern recognition and anomaly detection to mitigate identity threats or attacks. This is important because identity-related attacks are extremely common. Up to 82 percent of the United States population has experienced an identity-related attack [15].

D. Malware

Malware can be analyzed statistically without executing the file or be analyzed using an ML or DL model [2]. Potential indicators of malicious content in a file or URL can be included in file headers, IP addresses, domain or file names, metadata, hashes, and other strings [20]. It is common for malware to be encoded in binary, which requires reverse engineering to analyze its behavior [2]. Therefore, static analysis can fail to detect advanced types of malware since the code is not being run. It is more effective to use dynamic analysis for potential malware because the file is being executed and the behavior can also be subject to analysis. This usually happens in a sandbox environment to isolate your host machine [20]. Honeypots and honeynets are also used to lure attackers to collect malware for analysis. Once the malware has been collected, the features can be extracted and ML/DL can be used to classify different malicious and non-malicious samples. For instance, these samples could be classified so it ranks them from most serious to the least serious so a company would know which ones pose the biggest threat [2].

Behaviors and attributes of malware can be used to detect known malicious signatures, but these signatures are not always effective. For example, types of malware like polymorphic or metamorphic viruses will change or edit their code, so behaviors and attributes generate undetectable signatures, and rule-based and signature-based detections will not work [2],[18]. Dynamic analysis with ML is more effective when dealing with obfuscated data features [19]. ML classifies malware based on techniques like fuzzy matching which uses a confidence score, and "automated property selection" to determine the importance of each feature [2]. In DL an autoencoder is a type of neural network that creates extra copies of data to decrease the loss within the input and output layers [2]. DL models like CNNs have been shown to outperform and produce more accurate results compared to standard ML models. For example, a study compared how an RNN and CNN model classified behavior, with API calls as the features, against the ML models using Hidden Markov Model and SVM, which showed that the DL models produced more accurate results [21]. RNNs are useful in malware detection for extracting features within time-series data like API calls. A CNN would be better for detecting malware hidden inside an image or video file [19]. Dynamic analysis outperforms static analysis, while DL models outperform ML models. Static analysis is quicker but less accurate than dynamic analysis, and zero-day malware cannot be detected. A hybrid approach uses both static and dynamic analysis. This allows for less computational power, while still providing some of the benefits of dynamic analysis like image processing [21]. ScaleMalNet is a DL architecture that takes a two-stage approach to Malware analysis. Stage one uses both static and dynamic analysis to detect malware, and stage two uses image processing to classify the detected malware [21].

E. Network Traffic

In cybersecurity defenses, ML and DL models are trained to classify network traffic, subject to monitoring and analysis. Telecommunication networks and the Internet of Things (IoT) transmit large amounts of heterogeneous traffic with complex and hidden patterns. DL is adequate when handling time-series data and sequential data, such as network traffic [38]. ML uses pattern mining techniques to detect correlations in data, which is used to detect repetitive network scans and volume-heavy DoS attacks. There are passive and active techniques for monitoring and analyzing network traffic, as well as passive and active cyber-attacks [2], [38]. Passive monitoring doesn't add data to network flow; it monitors more network data to identify patterns. An example of passive network monitoring is using a packet sniffer or IDS. Active monitoring is also called "synthetic" monitoring. Packets are created to simulate network traffic to learn about real-time network performance metrics [39]. Active cyber-attacks include DoS, spoofing, security breaches, and pivoting-based attacks. Man-in-the-middle and port scanning are passive network attacks [2].

A characteristic of network traffic is any value of a feature. For instance, a specific protocol such as HTTP is a characteristic of the protocol feature [19]. Packet sniffers and network logs provide relevant network features for the training process. DL automatically extracts features from .pcap or .pcapng files using a packet sniffer like Wireshark. A private key allows TLS/SSL data encapsulated in TCP packets to be decrypted so the features can be extracted from the plaintext. This extracts features such as protocols being used, bytes transmitted from source to destination and vice versa, numbers of packets in a session, time duration of the session, networking services or protocols used, status of TCP handshake, and the ACK packets. HTTP, SMTP, FTP, and Telnet are application layer protocols that already show data in plaintext within a packet sniffer. These are examples of vulnerable protocols that can be classified in network traffic. Attacks on applications can be caused by SQL injections, cross-site scripting, and buffer overflows, which are detectable through server-side traffic inspection. Fuzzy matching finds string patterns instead of signature-based detection methods for attacks using polymorphism to obfuscate code [2].

A labeled NSL-KDD data set contains simulated LAN traffic collected using tcpdump. This data set is used in training an ML model on 24 different types of brute force attacks, privilege escalation attacks, DoS attacks, and unauthorized access. Each sample has its specific attack type labeled, and the network traffic features, examples of

these features are listed in Figure 8. After the model is trained on this data set and features are standardized or normalized, it can classify new network traffic data into attack and non-attack classes for further analysis. "Ensembling" ML models combine multiple models to increase the accuracy of classifying network attacks or traffic [2].

1	duration	9	urgent
2	protocol_type	10	hot
3	service	11	num_failed_logins
4	flag	12	logged_in
5	src_bytes	13	num_compromised
6	dst_bytes	14	root_shell
7	land	15	su_attempted
8	wrong_fragment	16	num_root
17	num_file_creations	30	diff_srv_rate
18	num_shells	31	srv_diff_host_rate
19	num_access_files	32	dst_host_count
20	num_outbound_cmds	33	dst_host_srv_count
21	is_host_login	34	dst_host_same_srv_rate
22	is_guest_login	35	dst_host_diff_srv_rate
23	count	36	dst_host_same_src_port_rate
24	srv_count	37	dst_host_srv_diff_host_rate
25	serror_rate	38	dst_host_serror_rate
26	srv_serror_rate	39	dst_host_srv_serror_rate
27	rerror_rate	40	dst_host_rerror_rate
28	srv_rerror_rate	41	dst_host_srv_rerror_rate
29	same_srv_rate		

Figure 8: Network traffic features for classifiers [2].

A study on an AE-CNN was used with a classification layer. AE performed dimension reduction and feature extraction. In the classification layer, CNN used the features extracted from AE to perform pattern recognition for classifying applications generating traffic and encrypted traffic types, either from a Virtual Private Network (VPN) or not from a VPN. The classification of applications generating traffic was 98 percent accurate, while network traffic classification was 93 percent accurate [19].

Port-based, flow-based, and payload-based categories are used when classifying network traffic. A study on classifying mobile network traffic with CNN, LSTM, Multilayer Perceptron (MLP), and Spare Autoencoder (SAE) using the Android and FB-FBM data sets showed CNN as the most accurate classifier [38]. CNNs performed better automatically detecting and extracting features. A CNN-RNN had the highest detection score in a study classifying network traffic. It used ISCX2012 and DARPA1998 data sets that had spatial-temporal network features for a HAST-IDS with 97 percent accuracy. A CNN-LSTM was trained using CTU-13 and ISOT data sets to classify and detect botnet traffic, producing an overall accuracy of 99 percent [38].

VIII. THE BASICS OF GENERATIVE AI, LARGE LANGUAGE MODELS, AND CHATBOTS

A. What is Generative AI?

Generative AI, commonly referred to as "Gen AI," uses DL models to generate text, audio, image, video, and code based on a user's prompt [42], [43]. Types of Generative AI are Generative Adversarial Networks (GANs), Variational Autoencoders (VAEs), Transformer-based Models (TRMs), and Diffusion models (DMs) [41]. GANs have a generator to create new "synthetic" or fake data based on random noise and a discriminator network that decides if data is synthetic/fake or real. The generator and discriminator compete, and the loser of the "zero-sum game" has to change its parameters, which continues up to the point that the discriminator cannot tell if the data the generator outputs are synthetic/fake or real [41], [42], [43]. VAEs originated in 2013. These were the first DL models for creating practical audio/speech and images [44]. A VAE has an encoder and decoder like a traditional autoencoder (AE), but it uses a probabilistic approach with a Gaussian distribution to capture data patterns and generate similar samples [41], [42]. TRMs are used for sequence-to-sequence related problems, and they use "multi-head attenuation" to be able to process all the sequences in a user's input simultaneously instead of individual steps like in an RNN [41], [42]. They use "self-attenuation" to learn relationships/dependencies amongst objects, such as words, without regard to distance. Position encoding is used when dealing with NLP, so TRMs know the order of each word in a sequence [41]. DMs enhance a GAN's performance by adding random noise to data in the training set and training the model to remove the noise in iterations until the desired output is obtained [41], [43]. These are often used when creating high-resolution images, and DMs take longer to train than GANs and VAEs [43].

Large Language Models (LLMs) are based on transformer architecture. LLMs are trained using enormous quantities of data. They are sometimes referred to as "transformer LLMs." An encoder-decoder is used for self-attenuation, and GPU power is used for self-training to decrease training times [47]. Some forms of training include reinforcement learning with human feedback (RLHF), self-supervised training, fine-tuning, and zero-shot learning [47], [48]. LLMs also handle sequential data using NLP and Natural Language Understanding (NLU), especially with chatbots, to understand the meaning of a user's input/requests and process and generate human text using natural language [48]. LLMs use billions of parameters. OpenAI's older model, GPT-3, has roughly 175 billion parameters, and some LLM models are capable of processing over 100 natural languages [47].

ChatGPT is an example of a "chatbot' since it has a chatbot built onto an LLM and interacts in conversations with humans using natural language. A chatbot consists of a client module, network module, response generation module, and database module [50]. Today's popular applications with chatbots are OpenAI ChatGPT, Anthropic Claude, Google Gemini, Microsoft Copilot, and GitHub Copilot. Chatbots are built on top of different models. For instance, ChatGPT has models including GPT-4, GPT-4o, GPT-4o mini, o1-mini, and more. Some chatbots integrate other models. GitHub Copilot uses OpenAI's Codex model, and Microsoft Copilot uses OpenAI's DALLE and GPT models [49].

B. How are chatbots used in cybersecurity defenses?

Retrieval-augment generation allows companies to connect chatbots to external resources such as repositories. It's estimated that around 67 percent of companies will integrate Retrieval-augment generation and Gen AI to improve chatbot decision-making in a specific domain, like cybersecurity defenses. Instead of researching solutions to resolve an issue or responding to an incident, a chatbot can provide quick and accurate feedback on complex problems in real-time [51]. Chatbots have become a powerful tool for cybersecurity professionals. It is widely used for vulnerability testing/scanning, inspecting and generating code, analyzing logs, detecting malware and phishing, and more [50]. A chatbot can assist with system and network configurations, whether for troubleshooting current configurations or setting up new ones. Using a chatbot to issue commands for a specific task or command syntax saves cybersecurity professionals time from checking manuals, especially when there are a series of tasks or configurations. Table 9 shows an example of ChatGPT's response to a prompt regarding firewall configurations to block/secure vulnerable ports on a Linux system.

Log analysis with SIEM logs and other logs can be automated with chatbots to reduce incident response times and provide analysts feedback on security improvements to reduce the likelihood of these events occurring [53]. An attacker could be attempting a brute-force attack against Active Directory. A chatbot like ChatGPT can analyze events, firewalls, IDS, and other logs to see increased traffic patterns from a specific IP address and multiple failed logins. Based on these patterns, the chatbot can identify the type of attack, actions to take, and future preventative measures such as new security policies. One can simulate different types of attacks with chatbots so they can familiarize themselves with each attack pattern they haven't yet experienced and know how to mitigate or respond to each attack [51]. Chatbots perform code scanning and analysis on a multitude of programming languages to discover flaws or potential vulnerabilities, such as buffer overflow. This process would take a long time to do manually with significant lines of code, and a chatbot saves the analyst time while potentially detecting bugs they've missed [53]. OpenAI's Codex is a fine-tuned model specifically for interpreting and generating code, which makes it an excellent vulnerability scanner [52]. A cybersecurity analyst can prompt the chatbot to regenerate secure code and ask for feedback on particular vulnerabilities instead of conducting research. Chatbots have DL-based malware detection advantages, as discussed earlier in the paper since they are trained on updated data sets with well-known malicious code fragments, malicious signatures, and behavioral patterns [53].

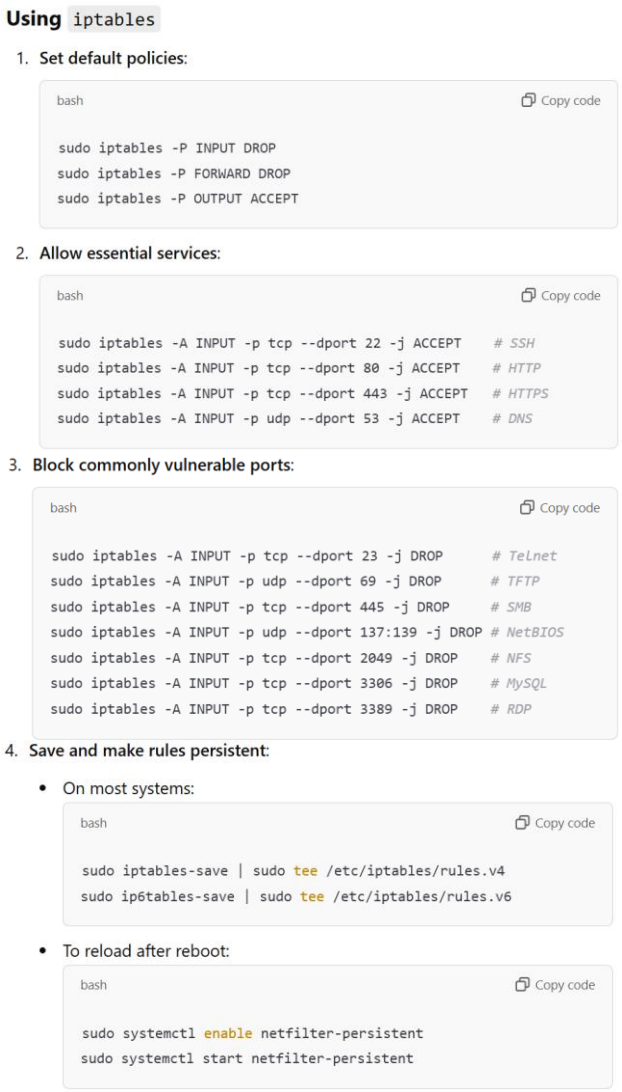

Figure 9: Output from a ChatGPT chatbot after prompting it to generate firewall commands/rules to secure all common vulnerable ports on a Linux machine.

C. Adversarial AI/ML attacks

Algorithms used in AI/ML can be vulnerable to adversarial AI attacks due to exploits in the training phase [13]. When user input and feedback are part of the learning process, an attacker can purposely manipulate the data used

in a model's training phase, such as a classification model, leading to incorrectly labeled samples. This is called "poisoning" the model. This can cause the model to make inaccurate decisions or predictions [46]. These inaccuracies pose a security risk if the model is used to derive a security solution. For example, CrowdStrike has a generative AI cybersecurity chatbot called "Charlotte AI," which many cybersecurity professionals use to ask questions and automate tasks. If Charlotte AI were poisoned and produced inaccurate responses, then the systems managed by the cybersecurity professionals using Charlotte would have a heightened security risk because of these newfound vulnerabilities caused by the poisoned model [14]. Poising a model is also called a "red herring" attack and is done on models that automatically learn online. Evasion attacks aim to manipulate classifiers after the training phase by generating adversarial inputs, so a model misclassifies and produces inaccurate predictions [46].

IX. DISCUSSION

This research project provided me with much newfound knowledge of AI since I originally had minimal knowledge regarding ML, DL, and Generative AI. Integrating AI into cybersecurity demonstrates significant advantages in mitigating current and future cyber threats. I also learned that unsupervised learning techniques, such as clustering, and supervised learning techniques, like classification, are the building blocks of ML. These techniques are also used in neural networks and their applications in cybersecurity. DL displayed enhanced capabilities in anomaly detection and IDSs, malware classification, and network traffic analysis, making it superior to traditional ML in cybersecurity defenses.

One challenge I faced during this project was reading through hundreds of research materials that often used highly technical and sophisticated language. I had to find a balance between simplifying these concepts and providing enough technical details to make the research meaningful to myself and the audience, who may also have limited AI knowledge. Additionally, my research on DL and Generative AI relied primarily on online articles and research papers, as the books I used mainly focused on ML. This created a learning curve in my understanding of DL, which is newer and more complex. If I could redo this project, I would prioritize my research using books on DL, Generative AI, LLMs, and their applications in cybersecurity.

Since this project was my first time researching AI in-depth, I started with limited prior knowledge of the topic; I learned how deep down the rabbit hole I could go on either one specific AI topic, such as ML classifiers, as well as a specific AI-driven cybersecurity topic, such as intrusion detection. Reflecting on this project has made me realize that covering multiple subsets of AI in a limited time lessened the strength of my research. For instance, I spent a month reading traditional ML books when I started the research. I realized I could deepen my research further if I chose traditional ML in cybersecurity. I learned that conventional ML is starting to become outdated, but this would have been an excellent topic for a beginner in AI. If I could redo this project, I would focus on that single subset to allow for more thorough research. Additionally, including a hands-on lab component would have provided technical details and enhanced the overall paper. I would have liked to include another section on Security Orchestration, Automation, and Response (SOAR) and the rising problems with deepfakes. Lastly, I would've added more on the current and future challenges with AI or AI in cybersecurity.

Despite the challenges, I believe the research project was a success because it allowed me to deepen my understanding of AI and focus on specific research areas in the future. I originally had minimal knowledge of AI and its applications in cybersecurity, and now I feel like I learned enough fundamentals to get into more technical research to start hands-on projects.

X. CONCLUSION

This research paper highlights the importance of AI, with a strong focus on ML and DL, and a brief overview of Generative AI, LLMs, and chatbots in cybersecurity defenses. It discusses basic implementations of traditional ML and how DL typically offers higher detection rates, greater precision, and fewer false positives, handles large quantities of data, and can detect complex cyber-attacks like zero-day attacks. Automating security tasks and reducing the constant dependence on human intervention are some of the key benefits of integrating AI-driven technologies. Chatbots have risen in popularity and are frequently used in automated vulnerability scans and secure code generation. These address many of the challenges posed by the rapidly evolving cyber-attacks and IT infrastructure. Vulnerabilities to adversarial attacks and ethical concerns regarding data privacy are key issues mentioned when discussing AI. With AI integration on the rise, insider threats are becoming more detectable. Companies worldwide are starting to use AI, not just for cybersecurity but for everyday business tasks. The findings of this research project display the necessity of AI in advancing cybersecurity defenses. Despite its challenges, the new insights I have gained from conducting this research have increased my support for AI adoption in cybersecurity practices. I believe that further research into real-world AI applications, along with the development of more sophisticated AI models, will be inevitable, and this will not only change the cybersecurity industry.

ABBREVIATIONS

AE – Autoencoders
AI - Artificial Intelligence
ANN - Artificial Neural Networks
API – Application Programming Interface
CNN - Convolution Neural Networks

CVEs - Common Vulnerabilities and Exposures
CWE - Common Weakness Enumeration
DAST – Dynamic Application Security Testing
DGA - Domain Generation Algorithm
DL – Deep Learning
DM - Diffusion Model
GAN - Generative Adversarial Network
GPU - Graphics Processing Unit
IDS – Intrusion Detection System
IPS - Intrusion Prevention System
LLM – Large Language Model
LSTM - Long Short-Term Memory
ML - Machine Learning
MLP - Multilayer Perceptron
NLP – Natural Language Processing
NLU - Natural Language Understanding
NGFW - Next-Generation Firewall
RBM - Restricted Boltzmann Machine
RLHF - Reinforcement Learning with Human Feedback
RNN - Recurrent Neural Networks
SAE - Sparse Autoencoder
SAST - Static Application Security Testing
SDN - Software-Defined Networking
SOAR - Security Orchestration, Automation, and Response
SVM – Support Machine Vector
TRM - Transformer-based Model
VAE - Variational Autoencoder
VPN – Virtual Private Network

REFERENCES

[1] B. Wallace, S. Akhavan-Masouleh, A. Davis, M. Wojnowicz, and J. H. Brock, *Introduction to Artificial Intelligence for Security Professionals*. Irvine, CA: The Cylance Press, 2017, pp. 1-78.

[2] C. Chio and D. Freeman, *Machine Learning and Security: Protecting Systems with Data and Algorithms*. Sebastopol, CA: O'Reilly Media, 2018, pp. 1-233.

[3] E. Keserer, "The six main subsets of AI: (Machine Learning, NLP, and more)," Akkio, 2024. https://www.akkio.com/post/the-five-main-subsets-of-ai-machine-learning-nlp-and-more (accessed Sep. 10, 2024).

[4] O. Watkins, "4 use cases for AI in cyber security," Red Hat, 2024. https://www.redhat.com/en/blog/4-use-cases-ai-cyber-security (accessed Sep. 10, 2024).

[5] G. Fung, "A Comprehensive Overview of Basic Clustering Algorithms," CiteSeer, 2001. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=b7cd0a4e0a129b55b76b7a0faf8ec73a5ebd1645 (accessed Sep. 20, 2024).

[6] S. Pitafi, T. Anwar, and Z. Sharif, "A taxonomy of machine learning clustering algorithms, challenges, and future realms," MDPI, 2023. https://www.mdpi.com/2076-3417/13/6/3529 (accessed Sep. 28, 2024).

[7] Google, "Supervised learning," Google, https://developers.google.com/machine-learning/intro-to-ml/supervised (accessed Sep. 28, 2024).

[8] scikit-learn, "Unsupervised Learning," scikit-learn, https://scikit-learn.org/stable/unsupervised_learning.html (accessed Oct. 10, 2024).

[9] L. K. Choudhury, "Study on logic and artificial intelligence subsets of Artificial Intelligence," Innovative Research Thoughts, https://irt.shodhsagar.com/index.php/j/article/view/1114 (accessed Sep. 28, 2024).

[10] scikit-learn, "Supervised Learning," scikit-learn, https://scikit-learn.org/stable/supervised_learning.html (accessed Oct. 15, 2024).

[11] R. Das and R. Sandhane, "Artificial Intelligence in Cyber Security," iopscience, 2021. https://iopscience.iop.org/article/10.1088/1742-6596/1964/4/042072/meta (accessed Sep. 15, 2024).

[12] P. Rubin-Delanchy, D. J. Lawson, and N. A. Heard, "Anomaly detection for cyber security applications | Dynamic Networks and Cyber-Security," World Scientific, 2016. https://www.worldscientific.com/doi/abs/10.1142/9781786340757_0006 (accessed Oct. 9, 2024).

[13] K. Cross, "Anomaly Detection in Cybersecurity," CrowdStrike, https://www.crowdstrike.com/en-us/cybersecurity-101/next-gen-siem/anomaly-detection/#:~:text=In%20the%20context%20of%20cybersecurity,breach%2C%20cyberattack%20or%20system%20failure (accessed Oct. 9, 2024).

[14] L. Stanham, "AI-powered behavioral analysis in cybersecurity," CrowdStrike, 2023. https://www.crowdstrike.com/en-us/cybersecurity-101/artificial-intelligence/ai-powered-behavioral-analysis/?srsltid=AfmBOorh-ruCZR1Yt4SF11l7JXxuzQm1ao1Kly44zoeSpgvfiuAEZ7O4 (accessed Oct. 9, 2024).

[15] V. Shastri, "The impact of machine learning and AI in identity security," CrowdStrike, 2024. https://www.crowdstrike.com/en-us/cybersecurity-101/next-gen-siem/anomaly-detection/ (accessed Oct. 9, 2024).

[16] AWS, "What is a neural network? ," AWS, https://aws.amazon.com/what-is/neural-network/ (accessed Oct. 15, 2024).

[17] E. Fichtner, "5 Amazing Applications of Deep Learning in Cybersecurity," Datto, 2022. https://www.datto.com/blog/5-amazing-applications-of-deep-learning-in-cybersecurity/ (accessed Oct. 15, 2024).

[18] G. Apruzzese, M. Colajanni, L. Ferretti, A. Guido, and M. Marchetti, On the Effectiveness of Machine and Deep Learning for Cyber Security, 2018. https://ccdcoe.org/uploads/2018/10/Art-19-On-the-Effectiveness-of-Machine-and-Deep-Learning-for-Cyber-Security.pdf (accessed Oct. 15, 2024).

[19] D. S. Berman, A. L. Buczak, J. S. Chavis, and C. L. Corbett, "A Survey of Deep Learning Methods for Cyber Security," mdpi, 2019. https://www.mdpi.com/2078-2489/10/4/122 (accessed Oct. 15, 2024).

[20] K. Baker, "Malware Analysis," CrowdStrike, 2023. https://www.crowdstrike.com/en-us/cybersecurity-101/malware/malware-analysis/?srsltid=AfmBOoogqWFL6gCi5edQV0NoeODSgOlQdzEUSZjZQrBGzf9rEqTQwMhR (accessed Oct. 15, 2024).

[21] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran and S. Venkatraman, "Robust Intelligent Malware Detection Using Deep Learning," in *IEEE Access*, vol. 7, pp. 46717-46738. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8681127 (accessed Oct. 26, 2024).

[22] GeeksforGeeks, "Clustering in Machine Learning," GeeksforGeeks.org, 2024. https://www.geeksforgeeks.org/clustering-in-machine-learning/ (accessed Oct. 26, 2024).

[23] freeCodeCamp, "8 clustering algorithms in machine learning that all data scientists should know," freeCodeCamp.org, 2021. https://www.freecodecamp.org/news/8-clustering-algorithms-in-machine-learning-that-all-data-scientists-should-know/ (accessed Oct. 26, 2024).

[24] GeeksforGeeks, "Random Forest algorithm in machine learning," GeeksforGeeks.org, 2024. https://www.geeksforgeeks.org/random-forest-algorithm-in-machine-learning/ (accessed Oct. 26, 2024).

[25] I. Belcic, "What is classification in machine learning?" IBM 2024. https://www.ibm.com/think/topics/classification-machine-learning (accessed Oct. 26, 2024).

[26] CompTIA, "What is cybersecurity: Types and threats defined: Cybersecurity," CompTIA, https://www.comptia.org/content/articles/what-is-cybersecurity (accessed Nov. 2, 2024).

[27] I. H. Sarker, "Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions," Springer, 2021. https://link.springer.com/content/pdf/10.1007/s42979-021-00815-1.pdf (accessed Nov. 2, 2024).

[28] T. Aslanyan, "Deep Learning Fundamentals Handbook – what you need to know to start your career in AI," freeCodeCamp.org, 2020. https://www.freecodecamp.org/news/deep-learning-fundamentals-handbook-start-a-career-in-ai/ (accessed Nov. 2, 2024).

[29] G. Pang, C. Shen, L. Cao, and A. V. D. Hengel, "Deep Learning for Anomaly Detection: A Review," arxiv, 2020. https://arxiv.org/pdf/2007.02500 (accessed Nov. 2, 2024).

[30] GeeksforGeeks, "Introduction to deep learning," GeeksforGeeks, 2024. https://www.geeksforgeeks.org/introduction-deep-learning/ (accessed Nov. 2, 2024).

[31] J. Holdsworth and M. Scapicchio, "What is deep learning?," IBM, 2024. https://www.ibm.com/topics/deep-learning (accessed Nov. 2, 2024).

[32] cloudflare, "What is a neural network?," cloudflare, https://www.cloudflare.com/learning/ai/what-is-neural-network/ (accessed Nov. 2, 2024).

[33] K. Michael, R. Abbas and G. Roussos, "AI in Cybersecurity: The Paradox," in IEEE Transactions on Technology and Society, June 2023, vol. 4, no. 2, pp. 104-109.

[34] Z. Chen, "Deep Learning for Cybersecurity: A Review," *2020 International Conference on Computing and Data Science (CDS)*, 2020, pp. 7-18. https://ieeexplore.ieee.org/document/9275959 (accessed Nov. 3, 2024).

[35] R. Kimanzi, P. Kimanga, D. Cherori, and P. K. Gikunda, "Deep learning algorithms used in intrusion detection systems - a review," arXiv.org, 2024. https://arxiv.org/abs/2402.17020 (accessed Nov. 3, 2024).

[36] S. Zaharia, T. Rebedea, and S. Trausan-Matu, "Machine learning-based security pattern recognition techniques for code developers," MDPI, 2022. https://www.mdpi.com/2076-3417/12/23/12463 (accessed Nov. 3, 2024).

[37] P. Shahrivar, S. Millar, and E. Shereen, "Detecting web application DAST attacks with machine learning," rapid7, https://www.rapid7.com/globalassets/_pdfs/research/detecting-web-application-dast-attacks.pdf (accessed Nov. 13, 2024).

[38] M. Abbasi, A. Shahraki, and A. Taherkordi, “Deep Learning for Network Traffic Monitoring and Analysis (NTMA): A survey,” science direct, 2021. https://www.sciencedirect.com/science/article/pii/S0140366421000426 (accessed Nov. 13, 2024).

[39] S. Wickramasinghe, “Active vs. passive monitoring: What’s the difference?,” Splunk, 2023. https://www.splunk.com/en_us/blog/learn/active-vs-passive-monitoring.html (accessed Nov. 13, 2024).

[40] Statista, “Number of internet and social media users worldwide as of October 2024,” Statista, 2024. https://www.statista.com/statistics/617136/digital-population-worldwide/ (accessed Nov. 13, 2024).

[41] S. S. Sengar, A. B. Hasan, S. Kumar, and F. Carroll, “Generative Artificial Intelligence: A systematic review and applications,” arXiv.org, 2024. https://arxiv.org/abs/2405.11029 (accessed Nov. 13, 2024).

[42] S. Feuerriegel, J. Hartmann, C. Janiesch, and P. Zschech, “Generative AI ,” SpringerLink, 2023. https://link.springer.com/article/10.1007/s12599-023-00834-7 (accessed Nov. 14, 2024).

[43] C. Stryker and M. Scapicchio, “What is Generative AI?,” IBM, 2024. https://www.ibm.com/topics/generative-ai (accessed Nov. 14, 2024).

[44] K. Martineau, “What is Generative AI?,” IBM Research, 2023. https://research.ibm.com/blog/what-is-generative-AI (accessed Nov. 14, 2024).

[45] B. Wallace, S. Akhavan-Masouleh, A. Davis, M. Wojnowicz, and J. H. Brock, *Introduction to Artificial Intelligence for Security Professionals*. Irvine, CA: The Cylance Press, 2017, pp. 115-155.

[46] C. Chio and D. Freeman, *Machine Learning and Security: Protecting Systems with Data and Algorithms*. Sebastopol, CA: O’Reilly Media, 2018, pp. 315-332.

[47] AWS, “What is LLM (Large Language Model explained)? ,” AWS, https://aws.amazon.com/what-is/large-language-model/ (accessed Nov. 15, 2024).

[48] IBM, “What are Large Language Models (LLMs)?,” IBM, https://www.ibm.com/topics/large-language-models (accessed Nov. 15, 2024).

[49] M. Rebelo, “The best ai chatbots in 2024,” Zapier, 2024. https://zapier.com/blog/best-ai-chatbot/ (accessed Nov. 16, 2024).

[50] A. Qammar et al., “Chatbots to ChatGPT in a Cybersecurity Space: Evolution, Vulnerabilities, Attacks, Challenges, and Future Recommendations,” arXiv.org, 2023. https://arxiv.org/abs/2306.09255 (accessed Nov. 16, 2024).

[51] N. Sessions, “Bolstering cybersecurity: How large language models and Generative AI are Transforming Digital Security,” NVIDIA Developer, 2023. https://developer.nvidia.com/blog/bolstering-cybersecurity-how-large-language-models-and-generative-ai-are-transforming-digital-security/ (accessed Nov. 16, 2024).

[52] M. Hill, “6 ways generative AI Chatbots and LLMS can enhance cybersecurity,” CSO Online, 2023. https://www.csoonline.com/article/575377/6-ways-generative-ai-chatbots-and-llms-can-enhance-cybersecurity.html (accessed Nov. 17, 2024).

[53] M. Gupta, C. Akiri, K. Aryal, E. Parker and L. Praharaj, "From ChatGPT to ThreatGPT: Impact of Generative AI in Cybersecurity and Privacy," in *IEEE Access*, 2023. vol. 11, pp. 80218-80245. https://ieeexplore.ieee.org/abstract/document/10198233 (accessed Nov. 16, 2024).