UNIVERSITY OF TARTU
Faculty of Science and Technology
Institute of Computer Science
Computer Science Curriculum

Diana Šramova

A Survey of Machine Learning Methods and their Applicability for Security Analysis

Bachelor's Thesis (9 ECTS)

Supervisor: Raimundas Matulevičius, PhD

Tartu 2024

A Survey of Machine Learning Methods and their Applicability for Security Analysis

Abstract:

The problem highlighted in this thesis is to determine which Machine Learning (ML) and Deep Learning (DL) methods should be applied for detecting information technology (IT) security threats. As IT security attacks are becoming difficult to detect with current technology and resources, today's detection systems require solutions that utilize artificial intelligence (AI) subsets for robustness and automation. The solution to solve this problem is an analysis of ML and DL methods, estimating their applicability across 3 security cases: User and Entity Behavior Analytics (UEBA), vulnerability detection, and phishing detection. This analysis covers both supervised and unsupervised methods, including random forest, support vector machines, logistic regression, k-nearest neighbor, clustering algorithms, association rules, recurrent neural networks, convolutional neural networks, stacked autoencoders, and generative adversarial networks. These methods are considered based on their inputs, outputs, strengths, and weaknesses for specific security cases. The study approach ensures classification, patterns recognition, anomaly identification, and penetration testing, enhancing the robustness and automation of security systems. This solution provides security professionals with guidance on selecting the ML or DL techniques that should be applied to specific IT security tasks, thereby reducing risks and mitigating security threats.

Keywords:
Artificial Intelligence, Machine Learning, Deep Learning, Threat Analysis, Threat Detection

CERCS:P170 Computer science, numerical analysis, systems, control

Masinõppemeetodite ja Nende Turvaanalüüsi Kohaldatavuse Uuring

Lühikokkuvõte:

Selles lõputöös esile tõstetud probleem on määrata, milliseid Masinõppe ja Süvaõppe meetodeid tuleks infotehnoloogia (IT) turvaohtude tuvastamiseks rakendada. Kuna IT-turberünnakuid on praeguse tehnoloogia ja ressurssidega üha raskem tuvastada, vajavad tänapäevased tuvastussüsteemid lahendusi, mis kasutavad tehisintellekti alamhulki töökindluse ja automatiseerimise tagamiseks. Lahendus selle probleemi lahendamiseks on Masinõppe ja Süvaõppe meetodite analüüs, mis hindab nende rakendatavust kolmel turbejuhtumil: kasutajate ja üksuste käitumise analüütika tuvastamine, haavatavuse tuvastamine, ja andmepüügi tuvastamine. See analüüs hõlmab nii järelevalvega kui ka järelevalveta meetodeid, sealhulgas otsustusmetsa, tugivektormasinaid, logistilist regressiooni, K-lähimaid naabreid, klasterdamist, assotsiatsioonireegleid, korduvaid neurovõrke, konvolutsioonilisi neurovõrke, virnastatud autokodeerijaid ja vastandlikke generatiivseid võrke. Neid meetodeid käsitletakse konkreetsete turvajuhtumite sisendite,

väljundite, tugevate ja nõrkade külgede põhjal. See õppemeetod tagab klassifitseerimise, mustrite äratundmise, anomaaliate tuvastamise ja läbitungimise testimise, suurendades turvasüsteemide töökindlust ja automatiseerimist. See lahendus annab turbespetsialistidele juhiseid Masinõppe ja Süvaõppe tehnikate valimiseks, mida mida tuleks rakendada konkreetsete IT-turbeülesannete puhul, vähendades seeläbi riske ja leevendama turbeohtusid.

Võtmesõnad:
Tehisintellekt, Masinõpe, Süvaõpe, Ohuanalüüs, Ohu Tuvastamine

CERCS:P170 Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria)

Contents

1 Introduction **6**
- 1.1 Scope Of Thesis . . . 6
- 1.2 Problem and Research Question . . . 6
- 1.3 Contribution . . . 6
- 1.4 Structure Of Thesis . . . 7

2 Review of Machine Learning Methods **8**
- 2.1 Machine Learning Paradigms . . . 8
- 2.2 Random Forest . . . 10
- 2.3 Support Vector Machines . . . 11
- 2.4 Logistic Regression . . . 11
- 2.5 K-Nearest Neighbor . . . 12
- 2.6 Clustering Algorithms . . . 13
- 2.7 Association Rules . . . 14
- 2.8 Recurrent Neural Network . . . 15
- 2.9 Convolutional Neural Network . . . 16
- 2.10 Stacked Autoencoder . . . 17
- 2.11 Generative Adversarial Network . . . 18
- 2.12 Discussion . . . 19

3 Security Threats Detection **21**
- 3.1 Security Threats Detection Without Machine Learning . . . 21
- 3.2 Security Threats Detection With Machine Learning . . . 21

4 User and Entity Behavior Analytics **23**
- 4.1 Application of Supervised Machine Learning Methods . . . 23
- 4.2 Application of Unsupervised Machine Learning Methods . . . 27
- 4.3 Application of Supervised Deep Learning Methods . . . 29
- 4.4 Application of Unsupervised Deep Learning methods . . . 30
- 4.5 Summary . . . 30

5 Vulnerability Detection **32**
- 5.1 Application of Supervised Machine Learning Methods . . . 32
- 5.2 Application of Unsupervised Machine Learning Methods . . . 35
- 5.3 Application of Supervised Deep Learning Methods . . . 38
- 5.4 Application of Unsupervised Deep Learning methods . . . 39
- 5.5 Summary . . . 40

6 Phishing Detection **42**
6.1 Application of Supervised Machine Learning Methods 42
6.2 Application of Unsupervised Machine Learning Methods 45
6.3 Application of Supervised Deep Learning Methods 48
6.4 Application of Unsupervised Deep Learning methods 49
6.5 Summary 51

7 Concluding Remarks **53**
7.1 Limitations 53
7.2 Answer to Research Question 53
7.3 Future Work 54

Appendix **65**
I. Glossary 65
II. Licence 66

1 Introduction

In today's digital era, threat actors developing new ways to bypass detection and protection systems. Therefore, security detection systems require constant development to protect organizations and individuals. Implementing ML and DL methods provides automation and robustness to security operations.

1.1 Scope Of Thesis

This thesis observes ML and DL methods and their applicability in detecting security threats across 3 cases: UEBA, vulnerability detection, and phishing detection. Also, it determines the criteria these methods are applicable for classification, pattern recognition, anomaly identification, and penetration testing. The study analyzes the methods' inputs, outputs, strengths, and weaknesses. Based on this analysis, the approach provides insights into which of these observed methods should be applied to specific security tasks. During the writing, chatGPT's text robot assistance and Grammarly web application[1] were used to receive feedback on the content of the thesis, the structuring of chapter outlines and the correctness of language usage in the text of the chapters. Based on the received feedback, the text of the thesis has been refined, and language errors have been corrected [Ope24]. Also, the University of Tartu course [MIA24] was used for reference formatting.

1.2 Problem and Research Question

As security professionals need to enhance detection systems with automation and reliability using AI, they need to comprehend which AI subset methods can be applied for security threat detection. This leads to the research question: "What machine learning and deep learning methods are applicable for detecting different security threats?"

1.3 Contribution

To answer research question, the study provides a literature review based on various types of reviews, including scoping reviews, systematic reviews, and traditional (narrative) literature reviews. Information for the literature review was collected from databases like SpringerLink[2], arXiv[3], ResearchGate[4], and IEEEXplore[5]. For searching in these databases, various queries, including ML and DL method names, were used to find

[1] https://www.grammarly.com/grammar-check
[2] https://link.springer.com/
[3] https://arxiv.org/
[4] https://www.researchgate.net/
[5] https://ieeexplore.ieee.org/Xplore/home.jsp

information for the survey. Additionally, for the analytical part, queries, including ML and DL names with keywords related to threat detection cases, were used to find research proving the methods' applicability. The study first reviews ML and DL methods and determines their output criteria. Afterward, to answer our research question, 3 example cases from various threat detection cases across the world were selected. The thesis provides the methods' strengths, weaknesses, practical examples, and outputs for each threat detection case in the analytical sections, with research proof for the applicability of some methods for these security cases. This approach identifies each method's pros and cons and real-world performance, determining the applicable methods and their criteria for each of the 3 security cases to answer the research question.

1.4 Structure Of Thesis

The rest of this document is organized as follows: chapter 2 reviews ML and DL methods, including their inputs, training processes, outputs, and real-world applications in the information technology security field. It also identifies the methods' output results across criteria: classification, pattern recognition, anomaly identification, and penetration testing. Chapter 3 discusses threat detection, how it's done without ML and DL, and why these methods are needed. The next 3 chapters analyze the application of these methods in 3 specific cases: UEBA, vulnerability detection, and phishing detection, identifying the methods' applicability for specific criteria. These chapters also provide output examples of the methods that are recommended to be applied for specific cases. The final chapter concludes by summarizing the findings, discussing work limitations, addressing the research question, and providing directions for future investigation.

2 Review of Machine Learning Methods

This chapter discusses the methods within the core subsets of AI — ML and DL (see Figure 1). AI is a technology for the simulation of human intelligence by a system or a machine [MB22]. Machine Learning is a subset of AI, that involves algorithms that enable computers to learn from data and perform tasks without explicit programming [WDCP22]. Deep Learning, a subset of ML, models the human brain's learning process through artificial neural networks [MB22]. The goal of this chapter is to observe various ML and DL methods and determine the criteria for which they are suitable. The review includes input data, training processes, outputs, and real-world applications of each method within the IT security context. Based on this review, the aim of this chapter is to identify the criteria for which each method is applicable: classification, pattern recognition, anomaly detection, and penetration testing.

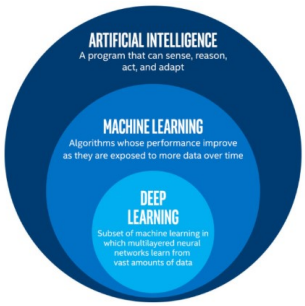

Figure 1. The Subsets of Artificial Intelligence [Ahm19]

2.1 Machine Learning Paradigms

The machine learning field has 4 paradigms: supervised learning, unsupervised learning, reinforcement learning, semi-supervised learning. Supervised learning is a machine learning paradigm that trains systems to understand the input-output relationship using labeled training data [LW12]. The goal of this paradigm is to make accurate predictions for new, previously unseen inputs. Unsupervised learning is a machine learning paradigm that deals with unlabeled data [Ed20]. The goal of this paradigm is to uncover hidden patterns without predefined output values. Reinforcement learning is a machine learning paradigm that involves learning to make decisions through trial-and-error experience [Bar97]. The goal of this paradigm is to develop a dynamic approach for sequential decision-making aimed at maximizing rewards over time [Li22]. Semi-supervised learning is a machine learning paradigm that "increases the effectiveness of supervised or unsupervised learning by leveraging a mix of both labeled and unlabeled data" [vEH20].

The chapter gives a review of the ML and DL methods that belong to supervised and unsupervised learning paradigms. Reinforcement learning and semi-supervised learning are excluded. Reinforcement learning, despite its power in simulated environments, is not suitable for real-world applications due to a pronounced difference between controlled experimental conditions and poorly defined realities of real-life systems [DALM+21]. Despite its promise, semi-supervised learning meets performance challenges when including unlabeled data [vEH20]. While progress has been made, its real-world applications are still limited compared to the established paradigm of supervised learning.

Table 1. Machine Learning and Deep Learning Algorithms

Paradigms/ AI subsets	**Machine Learning**	**Deep Learning**
Supervised Learning	Random Forest Support Vector Machines Logistic Regression K-Nearest Neighbor Decision Tree	Convolutional Neural Networks Recurrent Neural Networks
Unsupervised Learning	Clustering Algorithm Association Rules Generative Adversarial Networks	Deep Boltzmann Machines Stacked Autoencoder Generative Adversarial Networks

Table 1 presents the categorizations of the methods observed in the chapter. The supervised machine learning methods include random forest, support vector machines (SVM), logistic regression, k-nearest neighbor (KNN), and decision tree. The unsupervised machine learning methods include clustering algorithms, association rules, and generative adversarial network (GAN). Here is a need to mention that GAN is classified as both a machine learning method and a deep learning method. The GAN method itself belongs to machine learning, but because the GAN is based on 2 deep learning methods: a generator network and a discriminator network, it also, can be categorized as a deep learning method [GZ19]. Based on its components, in the study, GAN is considered as a deep learning method. The supervised deep learning includes convolutional neural network (CNN), and recurrent neural networks (RNN) [Zai18]. And unsupervised deep learning methods include deep boltzmann machines, stacked autoencoder (SAE), GAN [Gee]. The chapter does not cover deep boltzmann machines, due to the method's slow processing speed, which hampers its functionality and performance [LMM+21]. Additionally, the decision tree is excluded because, compared to random forest, which integrate multiple decision trees, random forest provide more accurate predictions than a single decision tree [IBMc].

2.2 Random Forest

Description Random forest, also known as random decision forest, is a set of decision trees where each tree is constructed using a randomly selected subset of data [Bre01]. This selection is independent for each tree but follows the same distribution across the forest. The effectiveness of a random forest is determined by the strength of the individual trees and their degree of correlation. A diverse collection of uncorrelated trees ensures the robustness of the model's predictions.

Algorithm Input For a random forest model, the input comprises a dataset with various features and their respective labels, with each instance characterized by its feature values [IBMf] [SKK23] [MW12]. During training, a subset of features is randomly chosen to construct each tree at every node, utilizing bagging and bootstrapping techniques to mitigate the overfitting common in individual decision trees. This method trains multiple trees on different subsets of the data, with replacement, enhancing the model's robustness by reducing variance without significant bias.

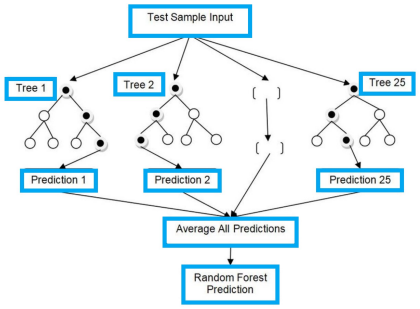

Figure 2. Example of Random Forest Regression Tree [PSSS20]

Algorithm Output The output of an random forest algorithm is the class prediction that presents the majority vote across all the decision trees in the random forest (Figure 2) [IBMf] [SKK23] [MW12]. This collective decision-making process leverages the 'wisdom of the crowd' with the collection of various uncorrelated tree predictions, producing a more accurate result than any individual tree within the model.

Examples Random forest methods are useful for the robustness and accuracy of machine learning detectors [ALMdO+23]. The study "Hardening Random Forest Cyber Detectors Against Adversarial Attacks" [AACM20] provides examples of random forest methods usage for different threat detections, such as PDF malware detections and flow-based botnet detections, producing detection rates close to 0.99.

2.3 Support Vector Machines

Description Support Vector Machines are a supervised machine learning algorithm that operates by identifying an optimal hyperplane that maximizes the margin between different classes in an N-dimensional space [CV95] [IBM23a]. This method is applicable in both linear and nonlinear classification tasks, making it versatile across various data types. The SVM algorithm was developed within the framework of statistical learning theory by Cortes and Vapnik in 1995 [CV95].

Algorithm Input SVM transforms input vectors into a higher-dimensional feature space to create a linear decision surface [CV95] [IBM23a]. Initially, data is divided into a training set and a testing set, often preceded by exploratory data analysis to identify any outliers or missing data. Afterward, the SVM model is trained on the training set where it learns to classify data by finding the optimal hyperplane that maximizes the margin between the classes. Performance is evaluated using metrics like accuracy, f1-score, precision, and recall. Additionally, hyperparameters such as the kernel type, regularization, and gamma are tuned through methods like grid search and cross-validation to improve the model's performance.

Algorithm Output The output of an SVM is a model with high generalization capability, classifying new data based on the learned hyperplane [CV95] [IBM23a]. SVM is applied in classification tasks across various fields due to its ability to handle high-dimensional data, choose kernel functions, and generate complex decision boundaries.

Examples SVM could be used for classification, and prediction in the IT security field, achieving an accuracy rate of 96.99% as provided by article "Cyber Security Approach In Web Application Using SVM" [CSD12]. Also, SVM is applicable in managing high-dimensional, heterogeneous, and unbalanced datasets, making it useful in intrusion detection systems (IDS) [SBZH21].

2.4 Logistic Regression

Description Logistic regression is a statistical method used to investigate the relationship between independent variables (either categorical or continuous) and a dichotomous dependent variable [Leo98] [IBMd]. This method calculates the log odds of the probability that an event will occur, which are then transformed by the logistic function to convert these log odds into a probability that lies within the 0 to 1 range.

Algorithm Input The input to logistic regression consists of independent variables, which can be categorical or continuous, paired with a dichotomous dependent variable

representing two classes (e.g., success/failure) [Leo98] [IBMd]. During training, the model constructs a relationship between the independent variables and the dependent variable's log odds, which is the natural logarithm of the probability ratio of success to failure. This relationship is captured using the sigmoid function, which maps any real-valued number into the (0, 1) interval, estimating the probability that the dependent variable belongs to a particular class. The model parameters are optimized through maximum likelihood estimation (MLE) to fit the model best to the observed data.

Algorithm Output Logistic regression output is a probability estimate, ranging between 0 and 1, generated using a sigmoid function [IBMd]. This probability represents the likelihood of a specific event occurring based on the input variables provided to the model. This method is applicable in classification and predictive analytics.

Examples Logistic regression can be applied in IT security for classification and attack detection problems [EM17]. It is helpful in threat detection systems, such as detection systems in security operations centers (SOC), where it is needed to reduce the amount of false positives. For example, the method could identify and categorize potentially suspicious URLs, as presented in the article "Machine Learning Algorithms for Detection of Cyber Threats Using Logistic Regression" [Gon23].

2.5 K-Nearest Neighbor

Description K-nearest neighbors is a non-parametric, supervised learning method. The method used for classification and regression tasks [IBMg]. This algorithm classifies individual data points based on the majority vote of their k-nearest neighbors, with the assumption that similar data points are close. KNN is applied for analytical tasks due to its simplicity and applicability in capturing the essence of the data's underlying structure.

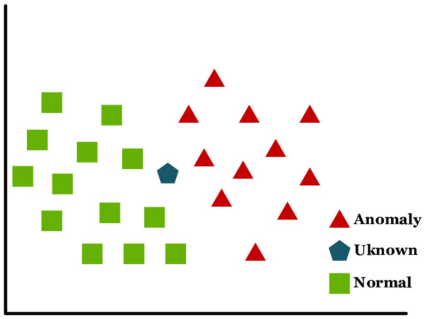

Figure 3. Example of K-Nearest Neighbor Classification [AMK+20]

Algorithm Input For KNN model training the input data should consist of a labeled dataset with examples, which is characterized by a set of variables [Nad16] [IBMg]. During training, KNN calculates the distance between the new sample and all training samples to find the nearest k neighbors (Figure 3). After, it classifies the new sample based on the majority category among these neighbors. If there is no agreement, further rules are applied to score each possible category and determine the most likely category for the new sample [Wan19].

Algorithm Output The output of an KNN is a classification or a prediction for previously unknown data points, determined by the most common class among the nearest k neighbors [IBMg]. This algorithm is utilized across various applications, especially for classification tasks.

Examples KNN could be useful in intrusion detection. The article "Use of K-Nearest Neighbor classifier for intrusion detection" [LV02] utilized KNN to classify program behaviors as normal or intrusive by analyzing the frequencies of system calls, which are treated as words in documents. This method detects intrusive attacks with a low false positive rate. Additionally, the study "On the Robustness of Deep K-Nearest Neighbors" [SW19] demonstrated that despite vulnerabilities, the deep k-nearest neighbor (DKNN) enhances the robustness against adversarial attacks.

2.6 Clustering Algorithms

Description Clustering algorithms or cluster analysis are a methods used to divide data objects into distinct groups known as clusters or subclasses [YLL23]. This methods aims to ensure that the objects within each cluster are as similar as possible to one another (maximizing homogeneity), and at the same time, as different as possible from the objects in other clusters (maximizing heterogeneity).

Algorithm Input Clustering algorithms input involves a array of data types, both structured and unstructured, including the large and various datasets [OIO+19]. The clustering have different methods: partitional clustering, hierarchical clustering, soft clustering, ensemble clustering, grid based clustering, density-based clustering and model based clustering. For example, partitional clustering divides a dataset into distinct clusters ensuring each data point belongs only to one cluster. Hierarchical clustering organizes data into a tree-like structure called a dendrogram, that enables data analysis at multiple granularity levels, and making it suitable for understanding complex structures. Soft clustering allows data points to belong to multiple clusters with varying degrees of membership, thus increasing the flexibility in cluster assignment.

Algorithm Output The outputs of clustering algorithms depending on the method employed [OIO+19]. As example, partitional clustering results in distinct, non-overlapping groups of data points. Hierarchical clustering, on the other hand, provides a dendrogram that represents hierarchical relationships among the data, which is useful for analyzing information at different levels of detail. Soft clustering allows data points to have memberships across multiple clusters, thereby offering flexibility in data categorization. Grid-Based clustering organizes data into a structured grid of cells, each representing a cluster, simplifying the processing of spatial data.

Examples Clustering techniques are used in IT security to detect and analyze threats, as provided in the research article "The Role of Machine Learning in Cybersecurity" [ALMdO+23]. For example, clustering employed in Android malware detection helps to achieve success rates over 95% and enables the grouping of malware to focus efforts on unknown clusters. Another example from the same study is that clustering can help identify protocols and network ports frequently targeted by attackers.

2.7 Association Rules

Description Association rules are a method in data mining for uncovering recurring patterns, correlations, and associations within large datasets [YMA+23]. The goal of association rules is to detect connections between disparate sets of items by indicating implicit or previously unknown, potentially valuable information. The structure of these rules is "X $\rightarrow$ Y," suggesting the likelihood of Y occurring when X is present, where X and Y represent the antecedent (condition) and consequent (result).

Algorithm Input Association rules mining is compatible with a wide range of data types, encompassing binary, discrete, and numerical attributes [YMA+23]. This flexibility makes association rules applicable in various scenarios where discovering patterns and connections within data sets is essential for informed decision-making. The generation of association rules involves sifting through all possible combinations of frequent item sets and then refining the rules based on the confidence criterion[6]. This iterative process ensures that only relevant and reliable rules are retained.

Algorithm Output The output of the association rules algorithm is a set of association rules that represent relationships between different sets of items [YMA+23]. These rules assist in identifying correlations between attribute values and offering information about the relationships within the data.

[6]https://athena.ecs.csus.edu/ mei/associationcw/RuleGeneration.html

Examples Association Rules could be used in Process Activity Monitoring, to flag unusual patterns in process activity, summarizing features like event types, executable names, IP addresses, ports accessed [BBCV21] .

2.8 Recurrent Neural Network

Description Recurrent neural network is a class of neural networks that are designed for processing sequential data [IBMb] [Pra23]. The RNN architecture includes loops that allow information to persist, simulating a form of memory by allowing outputs to be impacted by previous calculations (Figure 4). Their internal state (memory), which captures information about previously analyzed data points, makes them applicable for tasks that require consideration of individual data points within the context of a sequence, such as language translation, music generation, or speech recognition.

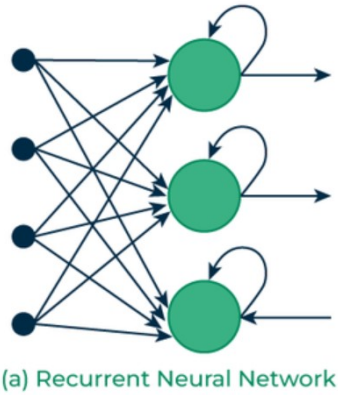

Figure 4. The Structure of Recurrent Neural Network [Gee23]

Algorithm Input RNN is designed for sequential input data [IBMb] [Pra23]. RNN do not train independently but utilize the backpropagation through time (BPTT) technique for training. BPTT adjusts the network's weights by propagating errors back through each timestep, optimizing the model for sequential data. The use of shared weights across recurrent connections helps to minimize error throughout the sequences and reduces the model's complexity and its computational requirements.

Algorithm Output The output of RNN depends on the specific task but generally includes predictions or classifications based on learned sequences in the data [IBMb] [Pra23]. From real-word applications RNN could be applied in tasks that require the analysis of sequential data, such as sentiment analysis, machine translation, and speech recognition, and for generating predictions in time-series analysis.

Examples RNNs could be used in IDS. For instance, the study "Design and Development of RNN Anomaly Detection Model for IoT Networks" [UM22] developed a deep learning model based on RNNs for anomaly detection within the Internet of Things (IoT) networks. Another study, "RNNSecureNet: Recurrent neural networks for Cybersecurity use-cases" [RVS18], explores RNN application in scenarios including incident detection, fraud detection, and android malware classification.

2.9 Convolutional Neural Network

Description CNN is a class of deep, supervised learning algorithms utilized for processing grid-like data [IBMa] [YND+18]. The model contains three types of layers (Figure 5): convolutional, pooling, and fully connected layers. The convolutional layers apply filters to the input data to generate feature maps, which highlight important attributes in the data. Pooling layers decrease the dimensionality of the feature maps by retaining only the most significant features. The fully connected layer then integrates these features to make predictions or classifications.

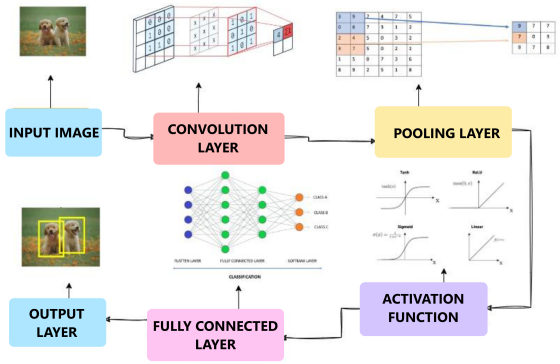

Figure 5. The Structure of Convolutional Neural Network [Tay23]

Algorithm Input The input data for an convolutional neural network is grid pattern type data [YND+18]. The training of a CNN involves optimizing the kernels in the convolutional layers and the weights in the fully connected layers to minimize differences between the predicted outputs and the actual labels. This optimization is achieved through a backpropagation combined with a gradient descent algorithm, which modifies the parameters iteratively by calculating the loss function and updating parameters in the direction that reduces the loss.

Algorithm Output The output of a CNN depends on the given task, such as prediction or classification, performed by a fully connected layer based on features extracted from

previous layers [Tay23] [IBMa]. CNN are applied for tasks such as image classification, object detection, tracking, pose estimation, text detection, visual saliency detection, action recognition, scene labeling, speech, and natural language processing.

Examples CNNs are applied in IDS. They could be used for classifying malicious images potentially used in attacks as proofed by "The Application of Convolutional Neural Network in Malware Images Classification" [WLZ22]. Another application provided by the study "Phishing Email Detection Using Improved RCNN Model With Multilevel Vectors and Attention Mechanism" [FZH+19] is phishing email detection, where CNNs analyze various components of emails. The proposed in this study Themis model, which utilizes a recurrent convolutional neural network (RCNN), has achieved an accuracy exceeding 99%, demonstrating the method's robustness in identifying phishing attempts [FZH+19].

2.10 Stacked Autoencoder

Description Stacked autoencoder is an unsupervised deep learning method designed to minimize reconstruction errors and enhance feature extraction capabilities without supervision [BCR+23] [KAH21]. Constructed as a deep neural network, the SAE consists of multiple layers of autoencoders (AEs) arranged vertically, where each layer is trained independently (Figure 7). Each autoencoder compresses the input data into a smaller representation in the encoding phase and expands it back to its original size in the decoding phase. This configuration allows the SAE to reduce data dimensionality effectively than a single-layer autoencoder, enabling it to retain information with enhanced feature compression.

Algorithm Input The input to a single autoencoder is a vector of n dimensions [KAH21]. The training of an AE involves two principal stages: encoding and decoding (Figure 6). During the encoding stage, the AE compresses the input data into a new, smaller set of features. In the decoding stage, it attempts to reconstruct the input data accurately using these learned features. SAE enhances this process by training multiple AEs in a layered sequence, where each layer's output serves as the input for the next. This methodical training progresses through all hidden layers to minimize the reproduction error to restore the original input data as closely as possible [BCR+23].

Algorithm Output The output of a stacked autoencoder is the reconstructed version of the input data, aimed at preserving essential features while minimizing the reconstruction error [KAH21]. This capability makes SAEs suitable for solving nonlinear problems across various applications. For example, they are used in image classification, image reconstruction, object detection, and various natural language processing tasks

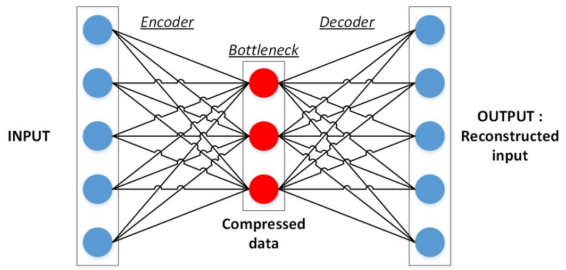

Figure 6. The Single Layer Autoencoder Structure Example [SK19]

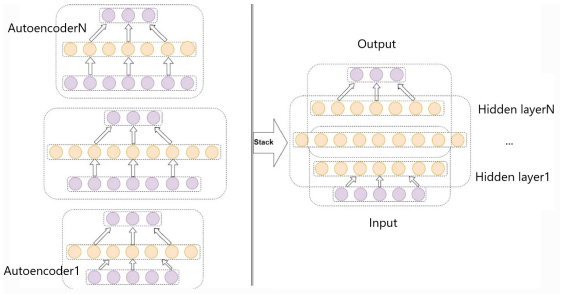

Figure 7. The Structure of Stacked Autoencoder [KAH21]

such as sentiment analysis, text classification, speech recognition, and recommendation systems [AK19] [LPL23].

Examples Stacked autoencoders are used in the network intrusion detection field [YQW+20] [TMG23]. As an example, the method used to detect anomalies and unknown types of attacks is presented in the study "Practical autoencoder based anomaly detection by using vector reconstruction error" [TMG23].

2.11 Generative Adversarial Network

Description Generative adversarial network is a deep learning model that learns the distribution of data classes [GZ19] [YXXZ20]. This model is structured around a two-person zero-sum game from game theory, where the loss of the other offsets the gain of one player. It consists of two components, the generator and the discriminator, which compete against each other. The generator aims to produce fake samples that imitate real data, attempting to fool the discriminator. Conversely, the discriminator acts as a binary classifier to differentiate between genuine data from the real dataset and fake data produced by the generator.

Algorithm Input GAN framework begins by taking real data samples as input [YXXZ20] [RRM20]. The generator uses a probabilistic latent space to produce data samples that resemble the target dataset. During training, the discriminator evaluates both real and synthetic samples from the generator and classifies them using a binary classification method. Throughout this process, both sub-networks—the generator and the discriminator—are optimized; the generator aims to produce increasingly realistic samples, while the discriminator enhances its ability to distinguish between real and generated data accurately (Figure 8).

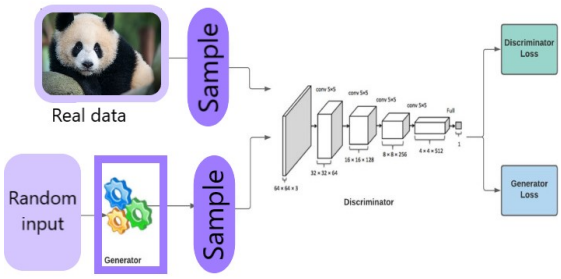

Figure 8. The Structure of the Generative Adversarial Network [BBC23] [AMB21]

Algorithm Output GAN model produces as an output synthetic samples that imitate the distribution of the original input dataset [YXXZ20]. Generative adversarial networks are applied for synthetic data generation, image reconstruction, video generation, segmentation, classification, and penetration testing tasks [GZ19].

Examples GAN model is applied in IT security, such as in IDS and Red Team activities [GZ19]. For example, GAN utilized for the detection of DDoS, insider, phishing, and adversarial attacks, as well as deep fakes, by generating synthetic adversarial examples that enrich training datasets [TGRD23]. In Red Teaming, GANs facilitate the creation of data for disinformation campaigns.

2.12 Discussion

Based on the observed methods, they can be grouped into 4 criteria useful for threat analysis and detection (Table 2): classification, pattern recognition, anomaly detection, and penetration testing.

Classification is a process of assigning previously unknown data to categories [Bat23]. This process is suitable for all supervised machine learning methods analyzed in the study, such as random forest, support vector machines, logistic regression, and k-nearest neighbor [LW12]. Supervised deep learning methods, including RNN and CNN, are also useful in classification. Although SAE are unsupervised methods, they can be adapted for classification tasks [AK19] [LPL23].

Pattern recognition is a process that analyzes data to identify patterns and regularities [Bah22]. The method is capable of detecting patterns, even if they are partially hidden or unfamiliar. Of the methods discussed in this chapter, the unsupervised machine learning techniques of clustering and association rules are applicable to this criterion [Ed20]. Additionally, deep learning methods like RNN and CNN can also be useful for this purpose [Imr23]. Although SAE and GAN have utility here, they are more suitable for

Table 2. Methods' Suitability for Criteria

ML/DL methods	**Classification**	**Patterns Recognition**	**Anomaly Detection**	**Penetration Testing**
Random Forest	X			
Support Vector Machines	X			
Logistic Regression	X			
K-Nearest Neighbor	X			
Clustering Algorithms		X		
Association Rules		X		
Recurrent Neural Network	X	X	X	
Convolutional Neural Network	X	X	X	
Stacked Autoencoder	X		X	
Generative Adversarial Network				X

other criteria [GZ19] [KAH21].

Anomaly detection is a process of identifying observations, events, or data points that differ from the norm [IBM23b]. From the methods observed, SAEs are suitable for this criterion due to their ability to detect reconstruction errors that deviate from normal behavior patterns [KAH21]. Additionally, RNN and CNN are also applicable here, due to neutral networks' multi-functionality depending on specific tasks [UM22] [AC20]. GAN, although neural networks are suited for different criteria, however, this method more applicable for other criteria [GZ19].

Penetration testing is a process of security assessment involving simulated attacks to identify vulnerabilities in a computer system [IBMe]. Among the observed methods, GANs are should be applied for this task [GZ19].

3 Security Threats Detection

Threat detection is defined as the process of "identifying and responding to security incidents to prevent or mitigate attacks and security breaches" [McC23]. The thesis describes 3 specific threat detection cases: UEBA, vulnerability detection, and phishing detection. For threat detection, 3 methodologies can be highlighted: signature-based detection, behavior-based detection, and ML-based detection [Hol22]. These methods form the basis of modern threat identification and response frameworks. In the described scenarios, threats were detected and categorized as suspicious by detection systems, including QRadar User Behavior Analytics (QRadar UBA), Trend Micro Vision One system, and Microsoft Defender XDR. These cases were further analyzed using ML and DL techniques to classify and estimate their severity. The next chapters of this thesis provide examples of these detected cases, including the ML and DL applicability analysis and output examples of the applicable methods.

3.1 Security Threats Detection Without Machine Learning

Without machine learning, the primary methods for threat detection could be signature-based threat detection and behavior-based threat detection [Hol22]. The signature-based detection method identifies security threats by looking for specific indicators, such as file hashes, file names, or registry keys already known and associated with malicious activity. This method is applicable for identifying known attacks, but it has limitations with new attack techniques, making it easier to manage with automation and additional context. The behavior-based detection method identifies security threats by comparing a user's behavior to their established baseline patterns. This method is applicable for identifying anomalous activities in the user's behavior. Despite its usability in identifying anomalies, the baseline of behavior-based detection methods must be regularly updated to reflect the changing nature of user behavior.

3.2 Security Threats Detection With Machine Learning

In today's information technology security landscape, threat detection methodologies, including signature-based and behavior-based detection, face challenges and limitations [Hol22] [Sab23]. The signature-based method struggles to identify unknown attack techniques due to their dependence on pre-existing knowledge bases. This limitation makes them less adaptable to new threats that still need to establish signatures. The behavior-based method, while dynamic, depends on holding behavioral baselines to detect anomalies. The need for regular updates and the generation of false positives from un-updated baselines make this method resource-intensive and less trustworthy in environments where user behavior frequently changes. Additionally, the volume of data

from network traffic, logs, and alerts disturbs both these detection methods to identify threats.

ML and DL methods address these challenges by implementing the automation, adaptability, and analytical depth of threat detection systems [Sab23]. ML and DL are adaptable for analyzing datasets of varying sizes, making them applicable to reduce the workload and improve the detection accuracy of traditional detection models. ML and DL learn from historical and real-time data, allowing them to identify patterns and anomalies that might detect known and new threats. Based on this, the integration of ML and DL can develop signature and behavior-based systems capabilities. The hybrid approach provides more accurate detections, where ML can add context to the data flagged by traditional methods with more robust solutions.

4 User and Entity Behavior Analytics

User and entity behavior analytics is a security process that analyzes user and entity behaviors to monitor and detect unusual activities [YJ21]. Its benefits include faster response times, automated threat detection, and reduced false positives through machine learning, statistical models, and rule-based systems. However, the human element is still need for investigating and verifying these anomalies.

Use Case The SOC team receives an alert from the QRadar User Behavior Analytics (QRadar UBA) due to suspicious non-privileged user behavior based on information from the log source, which is responsible for Office365. The user 7 times attempted to log in to Azure Active Directory from unusual for user geolocation - Kenya. The user's usual geolocation is Estonia. The first 6 attempts failed due to the wrong password (InvalidUserNameOrPassword, error code: AADSTS50126), but the last attempt was successful; the user successfully logged in and attempted access to confidential information documents, but the attempt was unsuccessful due to restricted permissions for the user. This is the first user activity in 1.5 months. The responsible log source collects this information and provides UBA QRadar model logs with the following information: time (23:59), username, source IP, destination IP (Azure Active Directory legitimate IP), action, error code, and error description. Based on detected events in QRadar UBA Moodle, the user risk score was decreased, and the alert was generated. After alert analysis by already trained AI, the SOC team received an alert with the tag - suspicious login from unusual geolocation following an attempt to access sensitive information. This categorization made by Artificial Intelligence help the SOC team prioritize the alert as high priority and immediately take response actions, which will help avoid data loss for the company.

4.1 Application of Supervised Machine Learning Methods

Supervised machine learning algorithms are applicable for tasks with labeled data, such as detecting suspicious user behavior based on historical habits. From supervised machine learning methods, one of the applicable methods is random forest. The random forest, compared to the decision tree, contains several single decision trees, which makes the random forest more quick and robust to overfitting and can handle complex high-dimensional datasets [IBMf]. Based on the advantages of random forest, this method can provide a robust solution for classification and prediction in this case. For instance, random forest can classify login attempts as suspicious by utilizing multiple decision trees to analyze various features such as time, IP addresses, geolocation, and records of successful and failed logins. Also, the detection accuracy for user and entity behavior anomaly by Random Forest provided in thesis "User and Entity Behavior Anomaly Detection using Network Traffic" [CN17]. Based on the same thesis, SVM also show the

detection accuracy in user and entity behavior data. The advantage of the SVM method is the ability to handle unstructured or limited nonlinear and high-dimensional data (such as login attempts, geolocations, and error codes) [CN17] [IBM23a]. In this case, SVM can classify logs based on binary classification task as suspicious and non-suspicious user behaviour. Logistic regression is used for classification tasks, which could help to classify user behavior as an anomaly or normal [IBMd]. The advantage of logistic regression is its low variance, indicating less variability in the data and resulting in more accurate classification [A9]. In this security case, the logistic regression algorithm can be applied to predict the probability of suspicious behavior from factors such as user multiple login attempts or source IP geolocation. The KNN has accuracy for data user and entity behavior anomaly detection, as provided in thesis "User and Entity Behavior Anomaly Detection using Network Traffic" [CN17]. The advantage is the ability to easily implement and adapt the KNN for different tasks [IBMg]. As example for particular security case, the algorithm can be applied for predictions based on known activities, or classification of user activity as suspicious or non-suspicious.

Based on the investigation, the recommended supervised machine learning algorithms for this case are random forest, SVM, logistic regression, and KNN. Random forest applicable for its accuracy and resistance to overfitting [IBMf] [CN17]. SVM applicable for accurately classifying anomalies in user and entity behavior [CN17] [IBM23a]. Logistic regression applicable for its low data variability and following accuracy [A9] [IBMd] and KNN applicable due to its accuracy in user and entity behavior anomaly detection [CN17]. The recommended methods' output examples for this case are presented after this analysis.

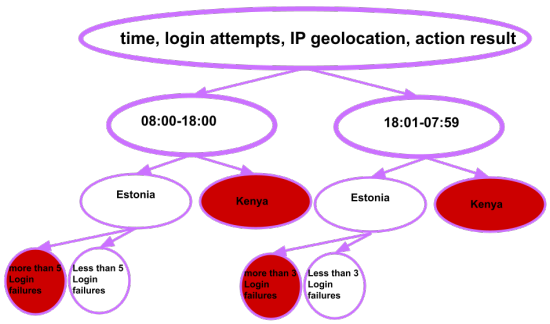

Figure 9. Single Decision Tree Example

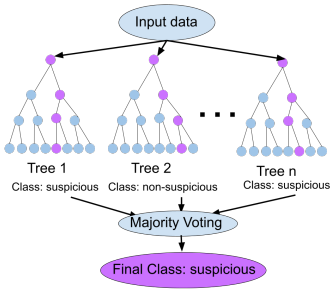

Figure 10. Random Forest Example

Random Forest Random forest is applied for classification tasks involving high-dimensional datasets [IBMf]. The random forest input data consists of current events data extracted from the Office365 log source, which includes time, login attempts, geolocation, IP addresses, and the success or failure of these attempts with corresponding error codes. Subsequently, various subsets of these features are utilized to predict outcomes using

multiple single decision trees within a random forest framework. Figure 9 illustrates an example of one such decision tree. The root node categorizes the dataset into two groups based on the event time: working hours (08:00-18:00) and non-working hours (18:01-07:59). The second layer of nodes further divides the data based on the source IP's geolocation - if the geolocation is Estonia, the process moves to the next layer; otherwise, the event is immediately categorized as suspicious. The final level divides the data based on the success or failure of login attempts. If the number of failures exceeds 3 during non-working hours or 5 during working hours, the event is categorized as suspicious; otherwise, it is deemed legitimate. Using this decision tree, all 7 current login attempts are predicted as suspicious user behavior, indicated by red circles in Figure 9. Figure 10 shows an example of a random forest with an unspecified number of decision trees, where each tree's prediction path is marked with purple circles from the beginning to the final node. The final prediction of the random forest will be presented as the most commonly predicted class by the decision trees — suspicious user behavior.

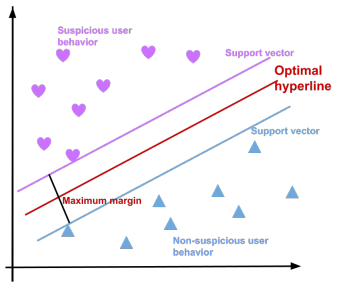

Figure 11. Support Vector Machines Binary Classification Example

Support Vector Machines SVM is applied for classification tasks by finding the optimal hyperplane that maximizes the margin between two classes [IBM23a]. For this unauthorized access case, the example can be a binary classification task, as shown in Figure 11; the output classifies user behavior into two classes: 'Non-suspicious user behavior' and 'Suspicious user behavior.' The input data includes the following features from logs: non-working hours, username, source IP, destination IP, action, error code, and error description. This data is divided by the optimal hyperplane into two classes. The 'Non-suspicious user behavior' class, marked in Figure 11 with blue color, includes data from previous login attempts and user behavior: working hours, source IP addresses from Estonia, legitimate action performed by the user after successful login, and previous error codes followed by successful login from Estonia. The 'Suspicious user behavior' class, marked in Figure 11 with purple color, also includes current case data: non-working hours login time (23:59), source IP addresses from unusual geolocation - Kenya, multiple failed login attempts, error code, and error description.

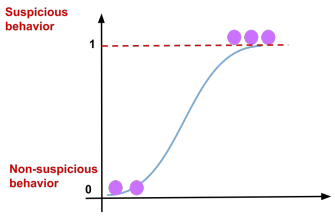

Figure 12. Logistic Regression Classification Example

Logistic Regression Logistic regression is suitable for classification tasks, such as predicting whether user behavior is suspicious or legitimate, by estimating probabilities using a logistic function [IBMd]. Based on current and historical data and the event's characteristics, the output is presented as a probability estimate, varying between 0 and 1, representing the likelihood of the monitored user behavior as suspicious or non-suspicious. Figure 12 illustrates an example of splitting data into suspicious and non-suspicious user behavior. The 'Suspicious behavior' class is shown in Figure 12 as a class with value 1 and includes current anomaly user behavior data: non-working hours activity, multiple failed login attempts followed by a successful login, failed attempts to log in from an unusual geolocation - Kenya, successful login from an unusual geolocation - Kenya, and attempts to access confidential documents. The 'Non-suspicious behavior' class includes normal user activity based on historical logs: working hours activity, successful login attempts from the usual geolocation - Estonia, with several (a maximum of 3) or no login failures, and no attempts to access confidential documents.

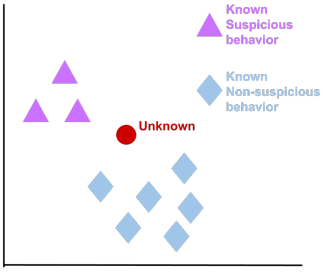

Figure 13. K-Nearest Neighbor Classification Example

K-Nearest Neighbor KNN is applicable for classifying or estimating the probability of a user's behavior being suspicious or legitimate [IBMg]. For instance, the output of KNN classifies user activity as suspicious or non-suspicious based on the events' similarity to known suspicious and normal activities for the user. Figure 13 provides an example of classifying one unknown event as 'Known Suspicious Behavior' or 'Known Non-Suspicious Behavior'. The unknown event is a successful login from an unusual

geolocation—Kenya—with an attempt to access confidential information. For this event, features include time (23:59), source IP (identifying the user geolocation as Kenya), action, and error code, which, in this case, is empty. This event is categorized as 'Suspicious User Behavior' based on already known historical user geolocation, error code, and normal activity time.

4.2 Application of Unsupervised Machine Learning Methods

The following section contains an analysis of the applicability of the unsupervised machine learning algorithms discussed in the thesis. The clustering algorithms are suitable for dividing data into groups based on the similarity of data objects, such as by dividing current and historical events based on geolocation, in this case [YLL23]. The applicability of clustering algorithms is presented in the article "A comprehensive investigation of clustering algorithms for User and Entity Behavior Analytics" [AMM24], showing a analysis of 15 clustering methods for UEBA. Conversely, the association rules are useful for investigating relationships between different sets of items or events [Mat22]. The algorithm is useful in data mining for analyzing and predicting data, such as future user behavior. In this case, the association rules can be applied to detect patterns such as multiple failed login attempts followed by a successful login and attempts to access confidential documents. Despite that, the disadvantage of association rules is that the algorithm could produce complex and misunderstood rules, which perform poorly [Mat22].

Given these insights, both clustering algorithms and association rules have their approaches to analyzing and handling data for detecting hidden patterns in user behavior, each with its own set of advantages and limitations. However, both can uncover dependencies in user behavior data through their approaches, indicating different patterns. This analysis provides examples of outputs from the applicable methods for this case.

Clustering Algorithms Clustering algorithms are applicable for organizing data into groups based on similarity, which helps uncover hidden patterns in the data [OIO+19]. Due to the wide range of clustering methods for output examples, 2 were selected. For partitional clustering, the output can be presented as several disjoint clusters, where each element belongs to only one cluster. As an example (Figure 14), the current and historical data are divided into two groups: 'Suspicious Login Attempts' and 'Normal Login Attempts'. The 'Suspicious Login Attempts' cluster includes IP addresses with previously unseen Kenyan geolocation, multiple login attempts with error codes and descriptions (if the login attempt result was failed), and non-working hours activity (midnight). The 'Normal Login Attempts' cluster includes data from the user's previous login activity: usual IP addresses with Estonian geolocation, login attempts with error codes and descriptions (if the login attempt result was failed), and activity during working

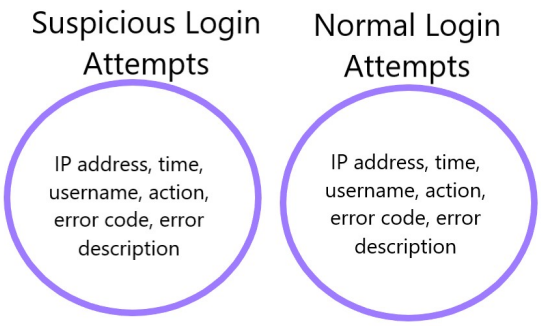

Figure 14. Disjoint Clusters

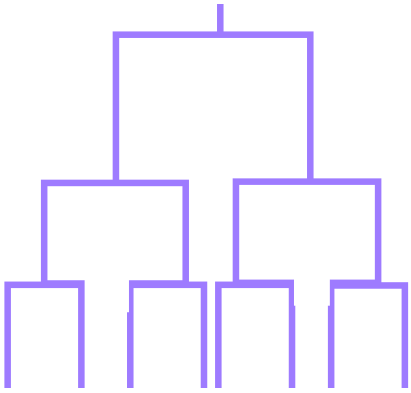

Figure 15. Dendrogram

hours.

For hierarchical clustering, the output can also be presented as a hierarchical tree structure - a dendrogram (Figure 15). This dendrogram illustrates the relationships between different clusters or subgroups of divided input data. For instance, the result shows data divided into groups: suspicious and normal login attempts, based on previous login attempts data. The first layers are divided based on anomaly (Kenya) and usual (Estonia) user geolocation. At the second dendrogram level, the data is divided into successful and unsuccessful login attempts. The third layer presents data divided into error codes and error descriptions.

Table 3. Association Rules Examples

Rule Number	**Rule Example**
Rule 1	If a user attempts to log in from an unusual geolocation, there is a high probability of subsequent failed login attempts.
Rule 2	If a user's login attempts fail multiple times due to incorrect passwords, then there is a high probability of subsequent successful login attempts using password reset.
Rule 3	If a user successfully logs in from an unusual geolocation, there is an increased risk that the user will attempt to access confidential data.

Association Rules Association rules are applicable to discover uncovered relationships and common patterns among variables in datasets [YMA+23]. The output can be presented as a set of association rules that identify correlations between attribute values based on analyzing relationships between different sets of items. Table 5 provides examples of three rules. The first rule indicates that when a user tries to log in from

an unusual location, there's a likelihood of subsequent login failures. The second rule indicates that repeated failed login attempts due to incorrect passwords may result in a successful login via password reset. The third rule highlights that successful login from an unusual location increases the risk of the user attempting to access confidential data.

4.3 Application of Supervised Deep Learning Methods

The following section observes the supervised deep learning algorithms for the UEBA case. RNN is designed for sequential data analysis, such as logs of user activities over time [Pra23]. For example, RNN can be applied to analyze login attempts over time and predict future actions in user behavior. The study "A Review Paper on Different Deep Learning Methodologies for User and Entity Behavior Analytics (UEBA)" [DDG+23] provides an example of RNN applicability for monitoring user behavior over time. Convolutional Neural Networks (CNNs) are also used for analyzing sequential data [IBMa] [YND+18]. However, CNNs require input data with a grid pattern type, which is more applicable to image data and natural language processing and may not directly apply to sequential event log data.

In summary, CNN could potentially be used for UEBA case analysis as provided in study "A Review Paper on Different Deep Learning Methodologies for User and Entity Behavior Analytics (UEBA)" [DDG+23]. However, they are mainly applied for grid-like topology data analysis, such as image classification, object detection, phishing detection, and natural language processing [Pra23] [YND+18]. Based on this, RNNs are considered more aplicable due to their ability to capture patterns, anomalies in the user's behavior logs, and temporal dependencies in this case [DDG+23]. Based on this, the example applicable to this case method output is provided below.

Recurrent Neural Network RNN is applicable for classifying user behavior patterns over time, detecting anomalies, partterns recognition, and predicting future events based on previous data [Pra23] [UM22] [Imr23]. The RNN output can provide a classification of user behavior into different categories: normal user behavior, anomaly user behavior (such as unauthorized login attempts from suspicious geolocations, or successful logins after multiple login failures in a short time). For example, based on algorithm training on historical logs (successful logins from Estonia during working hours with subsequent normal user activity), the algorithm output provides a classification of future user behavior as suspicious or non-suspicious. In this case, the future event is a successful login from a previously unseen IP. The event input data include time - midnight, a source IP address with a Kenya geolocation, and action - success. Based on information received from training, the algorithm already knows that the user's usual geolocation is Estonia, and normal user login time is morning. Due to this, the algorithm detects the event as suspicious user behavior.

4.4 Application of Unsupervised Deep Learning methods

This section assesses the application of unsupervised deep learning algorithms for UEBA case. In detecting unauthorized access attempts, SAEs can utilize their reconstruction capability to highlight anomalies in user behavior [KAH21]. The example of using the AE model for UEBA to detect and report anomalies is provided by article "An Artificial Neural Network Autoencoder for Insider Cyber Security Threat Detection" [SMD+23]. Also, the pros of SAE are their capability to learn complex, non-linear data representations, which makes them applicable for identifying patterns and deviations from those patterns, such as anomaly login geolocation - Kenya [BCR+23] [KAH21]. GAN is used to generate synthetic samples that imitate the distribution of the original input dataset, which is applicable for tasks requiring fake data usage but not unauthorized access detection [YXXZ20] [HSS+20].

In this case, SAE is more recommended for application than the GAN model. The GAN is more useful for penetration testing tasks to make the detection system more robust to detections than for the unauthorized access case [GZ19] [YXXZ20]. This analysis presents an output example from the recommended SAE method for this case.

Stacked Autoencoder Stacked Autoencoder is applicable for classifying and detecting anomalies, based on reconstruction errors [BCR+23] [KAH21]. The SAE output could be anomaly detection based on reconstructed input data. After the output generation, the reconstructed input log data is compared with the original input to detect user behavior as anomalous or normal. The model is first trained on typical historical user behavior, focusing on parameters like login times, username, source IPs, user actions, and error codes from security protocols. Each user action during training is monitored for reconstruction errors to establish anomaly scores. In this particular case of repeated login attempts from atypical geolocation with subsequent unauthorized access attempts to sensitive data, the difference between the original behavior data (which includes the timing of login attempts (non-working hours), geolocation deviations represented by source IPs (Kenya), attempted actions, and corresponding error codes) and the autoencoder's output (the reconstructed scenario) detects a high anomaly score for the user. This high anomaly score signals a significant divergence from the learned training representation of normal user behavior and detects anomalies in user behavior.

4.5 Summary

This chapter analyzed 10 machine learning and deep learning methods across 4 categories: supervised machine learning, unsupervised machine learning, supervised deep learning, and unsupervised deep learning. For each category, specific methods were recommended for this scenario based on their strengths, weaknesses, and applicability. Based on the analysis, it is recommended to apply 8 methods for criteria of classification, pattern

recognition, and anomaly detection. Penetration testing criteria are excluded from these recommendations since the QRadar UEBA model detected unauthorized access attempts successfully. Table 4 provides examples of methods output results for criteria they are applicable for in this case.

Table 4. Methods' Output for Criteria

ML/DL methods	**Classification**	**Patterns Recognition**	**Anomaly Detection**
Random Forest	Dormant Account, Unauthorized Access		
Support Vector Machines	Suspicious Geolocation, Possible Account Compromise		
Logistic Regression	Successful User Login, Suspicious Geolocation		
K-Nearest Neighbor	Suspicious Geolocation, Sensitive Data Access Attempt		
Clustering Algorithms		Unusual User Activity Cluster, Geolocation Anomaly Cluster	
Association Rules		Successful Login and Unusual Location Association	
Recurrent Neural Network	Possible Account Compromise, Unauthorized Access	Login Failure Patterns, Unusual User Activity	Geolocation Discrepancy, Anomalous Login Behavior
Stacked Autoencoder	Anomaly User Behavior, Anomaly Geolocation, Anomaly Time		Anomaly Activity Time, Geolocation Discrepancy

To classify events in this case, the recommended supervised machine learning methods include random forest, SVM, logistic regression, and KNN [LW12]. Also, from supervised deep learning, is recommended to use RNN. For uncovering hidden patterns in user behavior, clustering algorithms and association rules from unsupervised machine learning methods are recommended due to their capacity to identify relationships and dependencies in data [Ed20]. Also for investigating hidden relations in this case RNN is useful [Imr23]. For detecting anomalies, RNN and SAE are suggested due to their adaptability in identifying deviations from normal user behavior [KAH21] [UM22].

5 Vulnerability Detection

Vulnerability detection is a security process aimed at identifying weaknesses in software, networks, and systems through software fault templates [MC11]. These vulnerabilities, if exploited by threat actors, can result in security violations. To mitigate these risks, findings from vulnerability detections are used to apply patches, thereby improving the system's security posture.

Use Case The SOC team was alerted by the Trend Micro Vision One monitoring system of a potential exploit attempt targeting CVE-2017-11771 [NAT19]. This alert was triggered by a solitary network connection attempt from a registered internal host towards port 445 on the internal domain controller, classified as a potential exploit based on predefined security protocols. Trend Micro Vision One, which aggregates data from various sources, including endpoints, servers, and cloud infrastructures, provided detailed incident data encompassing the time of the event (17:30), source hostname, the internal IP address of the host within the accounting department's network, MAC address, source port, destination hostname, IP address of the destination, destination port, MAC address of the destination, operating system version of the destination host (Windows 2016), the action taken (block), the rule that flagged the potential exploit, and the specific vulnerability detected. Additional AI-driven analysis results provided a "high-risk" tag for this detection in the security information and event management (SIEM) system. This allowed the SOC team to prioritize the security alert quickly. Therefore, the team coordinated with the teams responsible for the source and destination devices, mitigating the risk of data leaks and maintaining the organization's security posture.

5.1 Application of Supervised Machine Learning Methods

The supervised machine learning methods could be useful for analyzing possible vulnerability exploits based on current logs, historical data, and classification. One of the applicable methods for this case is random forest. The method's pros are accuracy due to its ability to handle both regression and classification tasks, increased speed compared to a single decision tree, and ability to handle high-dimensional datasets, which would be suitable for analyzing the historical events between the internal host and domain controller [IBMf]. For example, random forests can analyze historical log data to identify patterns in past exploitation attempts and classify events as usual or malicious based on learned patterns. The instance of applying the random forest is shown in the study "Prediction of Software Vulnerabilities Using Random Forest Regressor" [KR23], where for the prediction of software vulnerabilities, the random forest regressor performed the detection result with a root mean square error of 0.01945. SVM can handle both linear and non-linear data, making it adaptable for analyzing different current and historical data patterns [IBM23a]. Also, compared to logistic regression, SVM is better at handling

high-dimensional data and is less prone to overfitting. The research "Construction of a SVM Learning Model in the Categorization Framework for CVE" [PH13] provides an example of the application of SVM for building a categorization framework for common vulnerabilities and exposures (CVEs). For this case, the SVM could analyze known exploit attempts and identify similar patterns in current log data. Logistic regression could be applied to classify and predict the likelihood of an exploit attempt occurring based on historical log data [IBMd]. Due to less variability in the data for the method, the result classification provides a high accuracy [A9]. In this case, logistic regression can analyze the historical correlations between hosts and previously detected exploit attempts, leveraging the learned information to provide probability calculations for new log data. KNN could be applied for classification tasks [IBMg]. For this case, the algorithm could classify log data based on similarity to previously observed data. However, KNN is prone to overfitting, where it may closely match the training data and perform poorly on new data [IBMg]. This makes the method unsuitable for cases with high-dimensional data such as historical and current network traffic logs. Additionally, it requires significant memory and data storage resources.

Based on the discussed methods, the recommended supervised machine learning algorithms for this vulnerability detection case include random forest, SVM, and logistic regression. Random forest accurately handles complex data [IBMf] [KR23]. SVM adapts to various data patterns, making it applicable for classifying network traffic [IBM23a]. Logistic regression is applicable in predicting event risk probabilities, making it suitable for vulnerability detection [A9]. However, KNN is not recommended due to its susceptibility to overfitting, which could impact its performance on new data [IBMg]. The output examples from the recommended methods for this case are following this analysis.

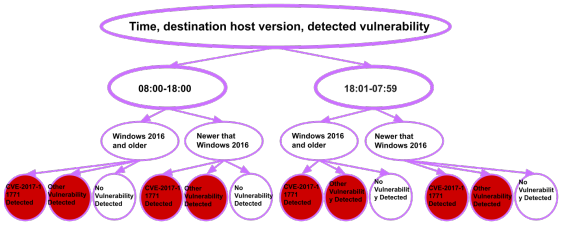

Figure 16. Decision Tree Example

Random Forest Random forest is applicable for data classification to identify anomalies and suspicious events, such as potential security threats, by analyzing various characteristics and behaviors in network traffic data [IBMf]. In this case, the output can be provided as predictions regarding the nature of network traffic, where the final prediction is determined by the majority vote among all the decision trees in the random forest. Figure 16 provides an example of a single decision tree. This decision tree

analyzes current events from the internal accounting department network. The decision tree processes a dataset that includes time, destination IP, MAC addresses, host versions, and detected vulnerabilities. The first layer categorizes data based on working hours (08:00-18:00) and non-working hours (18:01-07:59) when the likelihood of an attack is higher than during working hours. The second layer separates the data based on the version of the destination host's operating system, differentiating between versions up to Windows 2016 and more recent versions of the Windows operating system. This differentiation aids in identifying hosts that are operating on vulnerable operation system versions. The final layer accurately identifies the specific vulnerabilities detected; it differentiates between CVE-2017-11771 (indicated in the diagram by a red circle), other known vulnerabilities (indicated in the diagram by a red circle), or the absence of vulnerabilities. According to this decision tree, the current detection case is predicted as suspicious network activity. Figure 10 provides an example of a random forest comprising n number of decision trees, where each tree's prediction result is marked as a purple circle path from the start to the final node. The final random forest prediction is presented as the most commonly predicted class across the decision trees, which in this case is 'Suspicious Network Activity'.

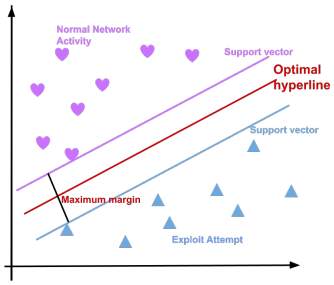

Figure 17. Support Vector Machines Classification Example

Support Vector Machines SVM is applied for classifying data into categories, such as 'Normal Network Activity' and 'Exploit Attempt,' by creating a hyperplane that maximizes the margin between these categories [**?**]. In the context of potential exploit detection, the output can be a binary classification: normal network activity or exploit attempt, as shown in Figure 17. The input data is divided by the optimal hyperplane into two classes, combining various parameters such as time, source hostname, source IP (local IPs from the internal client network for the accounting department), source host MAC address, source port, destination hostname, destination host IP address, destination port, destination host MAC address, destination host version, action taken, rule used for detection, and detected vulnerability (e.g., CVE-2017-11771). The 'Normal Network Activity' class (flagged in Figure 17 as purple) contains historical network activities characterized by regular communication patterns between internal hosts. The 'Exploit

Attempt' class (flagged in Figure 17 as blue triangles) contains unusual or unexpected communications to sensitive ports, with anomalies in communication patterns, such as rare interactions between a host and this specific domain controller on port 445, and detected vulnerabilities. Also, the 'Exploit Attempt' class includes current case data points with detected vulnerabilities - CVE-2017-11771, rare destination port 445, action taken as a block, and the corresponding triggered rule. Based on this classification, the current case event is counted as an exploit.

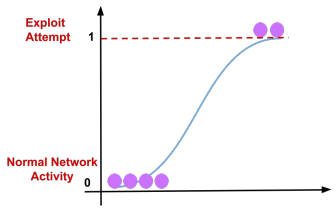

Figure 18. Logistic Regression Classification Example

Logistic Regression Logistic regression is suitable for predicting categorical outcomes based on estimated probabilities using a logistic function [IBMd]. For example, the model produces a probability estimate between 0 and 1, delineating the likelihood of a network event being classified as normal activity or an exploit attempt. Figure 18 interprets the logistic regression probability predictions for classification. The sigmoid function defined in Figure 18 separates events into two categories: 'Exploit Attempt' and 'Normal Network Activity.' The 'Exploit Attempt' class equals a value of 1, containing historical and current activities suspected of being exploit attempts. Characteristics indicative of this class include connections to vulnerable destination versions with unusual destination ports and related security rules. The detected event connected to a domain controller with a vulnerable version of Windows 2016 on port 445, associated with the detection of CVE-2017-11771 and the subsequent connection blocking by the set rules, also belongs to the Exploit Attempt class. The 'Normal Network Activity' class equals a value of 0. This class includes usual and expected historical network behavior events that align with known patterns of legitimate activity. Examples include DNS queries, such as a connection to port 53 directed at the website "err[.]ee" or successful authentication attempts during typical business hours.

5.2 Application of Unsupervised Machine Learning Methods

This section evaluates the applicability of unsupervised machine learning algorithms for analyzing the potential exploit of vulnerability CVE-2017-11771. Clustering algorithms could be applied for classifying data based on data similarity, making them useful

for categorizing network traffic to identify suspicious activities linked to CVE-2017-11771 [YLL23]. This method's benefit is its ability to identify unknown patterns in logs and datasets, as shown in article "Overview of clustering analysis algorithms in unknown protocol recognition" [LRC20]. Also, clustering could be applied in vulnerability clustering, as demonstrated in a study "Categorizing vulnerabilities using data clustering techniques" [VE04]. The study shows how these algorithms can be applied to classify vulnerabilities within the CVE repository and recommend a standardized categorization method using data clustering [VE04]. Association rules are applicable for finding relationships between variables in large datasets and can be applied to discover correlations between different types of network traffic activities or log data [Mat22] [YMA+23]. The advantages of this method are its versatility across various data types and the ability to uncover hidden patterns between data features. In this case, Association rules could be used to discover correlations between historical network events and known vulnerability CVE-2017-11771 exploits.

For this vulnerability detection case, it is recommended that unsupervised machine learning methods, clustering algorithms, and association rules be applied. Clustering algorithms could be applicable for dividing network traffic and log data into similar patterns [YLL23] [VE04]. At the same time, association rules could uncover relationships between data that correlate with known exploits and vulnerabilities [Mat22] [YMA+23]. After this analysis, examples of output for the recommended methods are provided.

Clustering Algorithms Clustering algorithms are suitable for segmenting incident data into clusters based on their similarity, which is useful in uncovering hidden patterns such as recurring exploit attempts from a specific host [OIO+19]. For the presented examples from various clustering algorithms, partitional clustering and hierarchical clustering are selected. The output of partitional clustering can be presented as several disjoint clusters, where each element belongs to only one cluster. For example, in Figure 19, the network traffic logs from the accounting department's network to the domain controller are divided into two clusters: 'Abnormal Behavior' and 'Normal Behavior'. The 'Abnormal Behavior' cluster groups incidents involving connections flagged as potential exploits. The current case with the CVE-2017-11771 exploit attempt to port 445 and the action taken - block - belongs to this cluster. The clustering criteria include source and destination IPs and MAC addresses, source port, destination port (port 445), actions taken (e.g., block), and the specific vulnerability detected (CVE-2017-11771). The 'Normal Behavior' cluster represents usual and expected network traffic from the accounting department's network to the domain controller, characterized by historical network activity patterns. It includes data on source and destination IPs and ports, MAC addresses, and usual actions taken. The output of hierarchical clustering can be presented as a hierarchical tree structure, a dendrogram. This dendrogram illustrates the relationships between different clusters or subgroups of divided input data. Figure 20

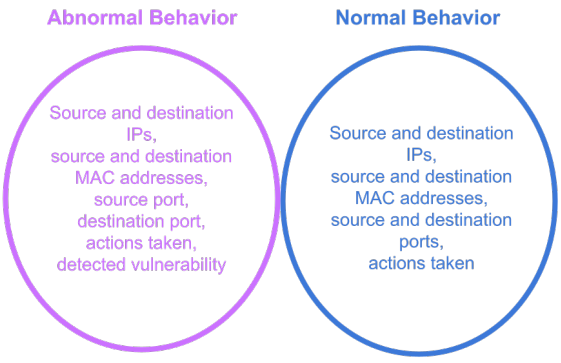

Figure 19. Disjoint Clusters

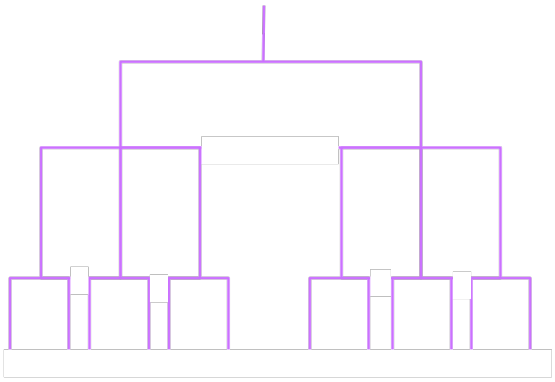

Figure 20. Dendrogram

illustrates the example of dividing historical and current logs data. In the example, the results are presented as data divided into groups: Traffic to the domain controller (internal requests, authentication attempts, or potential exploits to the domain controller) and Traffic from the Domain Controller (responses to legitimate requests or indicators of the Domain Controller being compromised). The second layer of both clusters is divided based on the action taken: block, allow, or reset. At the third level of the dendrogram, the data is divided by detected vulnerabilities, grouping data with CVEs or without detected vulnerabilities.

Table 5. Association Rules Examples

Rule Number	**Rule Example**
Rule 1	If an incident is detected at 17:30, then it could be related to end-of-day activities, possibly exploiting reduced vigilance or specific operational routines.
Rule 2	If the CVE-2017-11771 vulnerability is detected, then the action is to block the connection.
Rule 3	If a destination host is running Windows 2016, then there is a higher likelihood of CVE-2017-11771 exploitation attempts.

Association Rules Association rules are suitable for discovering relationships and correlations between data points, such as the correlation between specific network behavior events [YMA+23]. The output can represented as a collection of association rules that pinpoint correlations between attribute values. This is achieved by analyzing the relationships between various sets of items. The example is illustrated in Table 5.

The first rule illustrates a behavioral pattern based on timing, indicating that attackers may choose a time when vigilance is lower and initiate an exploit. The second rule shows an automated security response, where the detection of CVE-2017-11771 triggers the blocking of the connection, demonstrating the system's proactive actions against known vulnerabilities. The third rule provides an understanding of how the detected CVE-2017-11771 correlates with the destination host operating system version (hosts running Windows 2016 are more susceptible to exploitation attempts of CVE-2017-11771).

5.3 Application of Supervised Deep Learning Methods

The section analyzes supervised deep learning methods, including CNN and RNN. RNN is a category of neural networks that can be applied for various tasks with sequential data, making them applicable for tasks involving time-dependent log data analysis [Pra23]. The applicability of RNN in this case is proven by the study "Recurrent Neural Network Based Binary Code Vulnerability Detection" [ZPZ+20], which demonstrates detecting various vulnerabilities with RNN usage. In this detection case, RNN can learn standard network traffic behavior patterns over time and flag deviations. The following method, CNN, is also useful for analyzing sequential data and can handle large volumes of data [IBMa] [YND+18]. Despite this, CNN is primarily used in tasks processing grid-like topology data, such as image classification, pose estimation, text detection, and action recognition.

In conclusion, CNNs can be an applicable method for analyzing log data in this vulnerability detection case due to their focus on grid-like topology data and their capability to detect suspicious known patterns in logs. However, for CNNs, it is challenging to detect the temporal dependencies crucial for classifying in this scenario [IBMa]. Therefore, RNN is considered the optimal choice among the supervised deep learning methods in the thesis. Their ability to handle sequential data makes them applicable for detecting security vulnerabilities [ZPZ+20]. The output examples from the RNN for this case are detailed following this analysis.

Recurrent Neural Network RNN is applied for predicting and classifying sequential data, making them suitable for identifying patterns based on historical and current data from logs [Pra23] [IBMb]. Also, RNNs are useful for uncovering hidden patterns and anomaly detection [UM22] [Imr23]. The output example can be presented as a prediction of future network activities based on learned Trend Micro log source historical data. In this example, the RNN is trained to determine the likelihood of future network activities being either exploit attempts or normal network activities. This training leverages historical data containing regular network operations and documented exploit attempts. Based on the data learned during training, the RNN analyzes a new event: a connection attempt to the domain controller targeting port 445, which is associated with

CVE-2017-11771. The characteristics of this event, including the timing (at the end of the workday), source and destination details, and the CVE-2017-11771 vulnerability, help the trained model to predict its probability of being an exploit attempt with high accuracy.

5.4 Application of Unsupervised Deep Learning methods

In this section, the applicability of unsupervised deep learning algorithms was analyzed. SAE is designed to reconstruct input data while minimizing the reconstruction error, enabling them to identify deviations as deviations from regular traffic based on historical data for this case [BCR+23] [KAH21] [LTJY21]. Also, this model is applicable for determining complex data features and predicting non-linear relationships in logs. For example, SAE can be used to identify network traffic anomalies that divide from normal behaviors, as figured from historical data. On the other hand, GAN is applied for creating synthetic samples that closely mimic real datasets, making them applicable for intrusion tasks such as penetration testing [YXXZ20] [HSS+20]. The GAN is not directly applied for detection but is applicable to improving detection systems.

Based on this analysis, it is recommended to use SAE to analyze CVE-2017-11771 due to its suitability for feature learning and anomaly detection in the high-dimensional data typical of network traffic and logs [BCR+23] [KAH21] [LTJY21]. However, GAN is not directly applicable to this case since there is no need to improve the detection system for this threat recognition [YXXZ20] [HSS+20]. Trend Micro successfully detected and blocked the potential exploit attempt. This analysis includes example of recommended method output specific to this case.

Stacked Autoencoder Stacked autoencoder is suitable for classifying and detecting data as anomalous or normal by learning to reconstruct baseline behavior and later identifying deviations from this norm based on reconstruction errors [BCR+23] [KAH21]. The output can be presented as anomaly detection based on reconstructed input data. As an example of this case, the output detects anomalies in the current event based on reconstruction errors and anomaly scores. Firstly, the model is trained on a dataset containing historical normal network behavior to the domain controller, which includes parameters like IP addresses, ports, and actions taken by the network's security protocols. Each event during this process is evaluated through reconstruction error (difference between the input and the output), assigning anomaly scores. In the case of a detected CVE-2017-11771 vulnerability exploit attempt, a difference between the input data (unusual time, source and destination IP addresses, MAC addresses, ports, action taken, and detected CVE-2017-11771 vulnerability) and the model's output (the reconstructed event) detects a high anomaly score. This high score indicates the event's significant deviation from learned patterns of historical normal network behavior, flagging it as a

potential exploit attempt.

5.5 Summary

This section analyzed 10 different ML and DL algorithms across four categories: supervised machine learning, unsupervised machine learning, supervised deep learning, and unsupervised deep learning. The methods that should be applied for each category were identified based on their characteristics, advantages, and disadvantages to detect potential exploit attempts of CVE-2017-11771. In total, it is recommended to apply 7 methods for the criteria of classification, pattern recognition, and anomaly detection. Penetration testing criteria are excluded from recommendations since the Trend Micro Vision One solution successfully detected and blocked potential exploit attempts. Table 6 presents sample outputs of methods based on the criteria applicable to this scenario.

Table 6. Methods' Output for Criteria

ML/DL methods	**Classification**	**Patterns Recognition**	**Anomaly Detection**
Random Forest	Internal Exploit Attempt, Blocked Connection		
Support Vector Machines	Critical Port Access Attempt, Vulnerable Destination		
Logistic Regression	Internal Threat, Vulnerable Destination		
Clustering Algorithms		Exploit Attempt Cluster, Anomaly Cluster, Internal Activity Cluster	
Association Rules		Port 445 Exploit, Internal IP Block	
Recurrent Neural Network	Internal Exploit Attempt	Unusual Internal Traffic Pattern	Anomalous Traffic to Domain Controller
Stacked Autoencoder	Abnormal Port Access Attempt, Internal Host Compromise		Anomaly Port 445 Traffic, Anomalous Internal Host Behavior

To classify events in this case, it is suggested to employ supervised machine learning methods such as random forest, SVM, and logistic regression [LW12]. RNN is recommended from the supervised deep learning category due to its applicability in handling sequential data. For classifying events in this scenario, the application of SAE is also

recommended [AK19] [LPL23]. To identify hidden patterns in data, applying clustering algorithms and association rules from the unsupervised machine learning category is recommended [Ed20]. Unsupervised methods' are adept at uncovering relationships and dependencies in the data, which is crucial for understanding exploit attempts behaviors. Additionally, RNN from the supervised deep learning category applies to hidden patterns uncovering [Imr23]. RNN and SAE are suggested to apply for anomaly detection, as both methods can identify deviations from normal user behavior that indicate an anomaly [UM22] [KAH21].

6 Phishing Detection

Phishing detection is a security process that helps to detect the phishing attacks [ZMC23]. The phishing attack is a social engineering attack that exploits psychological and social engineering techniques to deceive the target [ZMC23] [KIJ13]. This attack aims to force a person to submit sensitive information or credentials to threat actors [ZMC23]. The impact of this attack could be compromised user accounts, stolen credit card information, and sensitive data leakage. For big organizations, the impact can include brand abuse, bid data leaks, financial losses, and reputation damages.

Use Case The SOC was alerted by Microsoft Defender XDR when the head of the security department reported an email that was phishing from their inbox. This email mimicked an official Microsoft support notification and asked the recipient to verify a new policy via a link. The details provided by Microsoft Defender XDR included the time of receipt, email header, sender's email (support@microsafttt[.]com), sender's IP address (noted for its Bhutanese geolocation and association with phishing activities), email title ("!!!URGENT!!! LOGIN AND AGREE WITH NEW POLICY!!!"), recipient's address, the Microsoft logo image, the initial location of the email in the inbox, and its final location in the quarantine folder. Additionally, the email included a fake Microsoft login page designed to gather user credentials via the URL (microsafttt[.]com) mentioned in the message. Following its report, the email was promptly moved from the inbox to the quarantined mailbox. The AI system analyzed and identified the alert as phishing, which helped the SOC prioritize the alert, block the sender, and further investigate any related sender data in the logs to uncover potentially undetected phishing activities.

6.1 Application of Supervised Machine Learning Methods

Supervised machine learning models can classify email for the described case. For instance, the random forest method can classify emails into phishing and non-phishing categories based on features extracted from email content and metadata. Despite requiring more storage resources and slower training due to many decision trees, random forest is still a beneficial method [IBMf]. It has strength against overfitting and can deal with the unbalanced and high-dimensional data commonly found in phishing detection tasks. The article "Classification of Phishing Email Using Random Forest Machine Learning Technique" [AA14] demonstrates the accuracy of phishing email classification using a random forest model, achieving a rate of 99.7%. The SVM is applicable for linear and non-linear classification tasks [IBM23a]. For instance, it can be applied in phishing email detection by identifying the hyperplane that separates phishing emails from legitimate ones based on features extracted from email content and metadata. The method's advantages include its ability to work with high-dimensional data and its robustness against overfitting. The application of SVM in phishing detection is demonstrated in

article "Phishing Attacks Detection by Using Support Vector Machine" [NG23], which showed that the SVM-based model achieved an accuracy of 98.8% in detecting phishing emails. Logistic regression is applicable for classification tasks and could be used for determining between categories like phishing and legitimate emails by assigning probabilities between 0 and 1 [IBMd]. The advantage of logistic regression is its low variance, which contributes to less variability in model predictions and provides classification accuracy [A9]. For instance, logistic regression can predict the likelihood of an email being phishing or legitimate by analyzing email features like the email header, sender information, and embedded URL. KNN is valued for its easy implementation and adaptability to new data [IBMg]. However, it is prone to high-dimensional data due to increased memory and storage requirements and sensitivity to data scale. Based on these limitations, KNN is less suitable for analyzing complex and high-dimensional email data.

Based on the discussed methods, the suggested supervised machine learning algorithms to use are random forest, SVM, and logistic regression [IBMf] [IBM23a] [IBMd]. These algorithms are adaptable to complex and high-dimensional datasets and are resistant to overfitting, which applies to the high-dimensional nature of email data. However, KNN is less suitable in this scenario because it struggles with the high dimensionality typical of email data [IBMg].

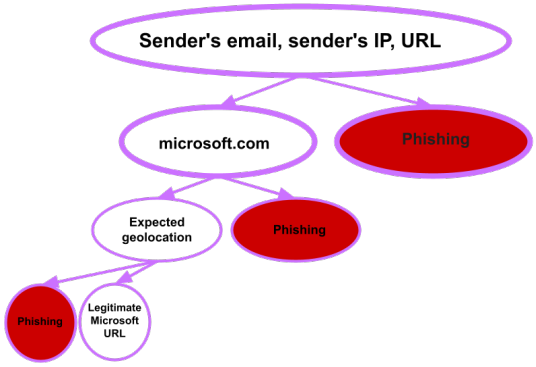

Figure 21. Decision Tree Example

Random Forest Random forest is suitable for data classification, such as classifying email as phishing or legitimate [IBMf]. The majority of predictions among the decision trees determine the random forest output. Figure 21 provides an example of a single decision tree created for this phishing case. The first layer assesses the sender's email domain: it checks whether the domain matches 'microsoft.com' or closely resembles it without belonging to Microsoft. If the domain does not belong to Microsoft, the email is immediately classified as phishing. Otherwise, the analysis progresses to the second layer, which evaluates the sender's IP geolocation. The email is marked as phishing if the IP's geolocation is not associated with legitimate Microsoft IP geolocations; if the

IP's geolocation is associated with legitimate Microsoft IP geolocations, the process moves to the third layer. This final layer analyzes the URL in the email to determine its legitimacy. If a URL is identified as a legitimate Microsoft web page, the email is categorized as legitimate; conversely, a suspicious URL leads to a classification of the email as phishing. For the current phishing case, the decision tree classifies the email as phishing. The Random Forest generates multiple such trees with variations in conditions and thresholds to cover a wide range of phishing characteristics. Figure 10 illustrates the Random Forest model consisting of n decision trees, with the final results marked by purple circles along the path. In this case, the final prediction of the email is phishing.

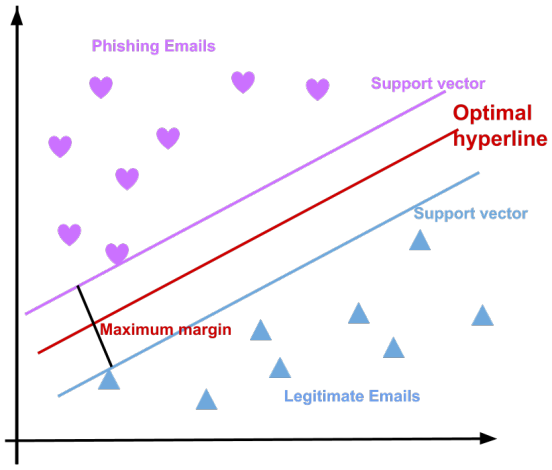

Figure 22. Support Vector Machines Classification Example

Support Vector Machines SVM are suitable for data classification tasks, such as determining emails as phishing or legitimate, by finding a hyperplane that maximizes the margin between these categories [IBM23a]. The output can be an email classification based on an optimal hyperplane that maximizes the margin between phishing and legitimate emails. Figure 22 illustrates this binary classification for current and historical emails. The model uses historical data to train and recognize future emails based on patterns learned during training. The input data includes the following email information: received time, email header, sender's email, sender's IP, email title, recipient (user's email), the initial email location, the final email location, embedded images, and URLs. The 'Phishing Emails' class, defined as purple in Figure 22, includes both previous phishing attempts and the current email due to its red flags (such as sender's email - support@microsafttt[.]com, sender geolocation - Bhutan, keywords associated with phishing in the title, and malicious URL - microsafttt[.]com). The 'Legitimate Emails' class, defined as blue in Figure 22, includes local emails, legitimate Microsoft emails, and emails from trusted sources (such as ut[.]ee) with expected geolocation.

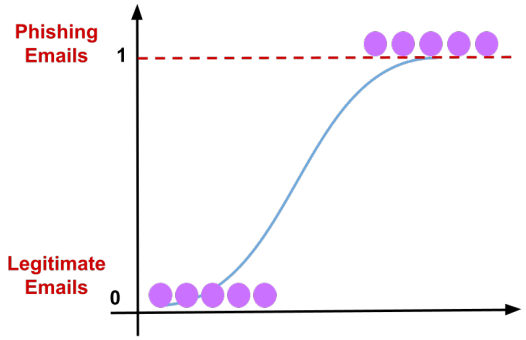

Figure 23. Logistic Regression Classification Example

Logistic Regression Logistic regression is suitable for predictive classification tasks, such as classifying an email as phishing or legitimate, based on estimated probabilities using a logistic function [IBMd]. The output can be presented as a probability estimate ranging from 0 to 1, reflecting the likelihood of an email being a phishing attempt. The model uses a sigmoid function to transform the email data into a probability between 0 and 1. Values approaching 1 indicate an increased likelihood of phishing, whereas values closer to 0 indicate a legitimate email. Figure 23 demonstrates the method's binary output in the context of both historical and current email data. In this instance, emails identified as 'Phishing Emails' include historical examples and a present case determined by factors such as the suspicious sender's email address, the sender's unusual IP geolocation (such as Bhutan), the urgent tone in the email's title, and the deceptive URL prompting for login credentials. Conversely, 'Legitimate Emails' include historical emails from verified and reliable senders, such as official Microsoft, IBM, and Trend Micro support addresses, and internal company communications from recognized and trustworthy IP addresses.

6.2 Application of Unsupervised Machine Learning Methods

The following section analyzes the applicability of the unsupervised machine learning methods. Clustering algorithms are applicable for organizing similar data and identifying common characteristics in phishing emails [YLL23] [OIO+19]. These algorithms are adaptable to large volumes of data, which helps analyze email datasets. For example, they can categorize phishing attempts by analyzing features such as email headers, sender's email addresses, sender's IP addresses, email titles, and URLs contained in the emails. The applicability of clustering methods for detecting and classifying phishing was demonstrated in the study "Using Clustering Algorithms to Automatically Identify Phishing Campaigns" [AWAV23], where clustering was integrated into the IT workflow to identify phishing campaigns. Association rules can uncover relationships between variables in extensive datasets, making it applicable for analyzing various datasets in

phishing email data [YMA+23]. One of the benefits of employing association rules is their applicability to various data types [Mat22] [YMA+23]. However, by creating complex rules, the method may produce false positives or overfitting when the rule set becomes large.

Based on the analysis, both methods for phishing detection are advised. Clustering algorithms can identify phishing attempts by grouping similar emails, while association rules can provide specific criteria for detecting phishing attempts based on associations discovered in the data [OIO+19] [Mat22] [YMA+23]. As a result, these methods can produce complementary results that can uncover hidden patterns in the data. Examples of output from the advised methods for this scenario are presented after this analysis.

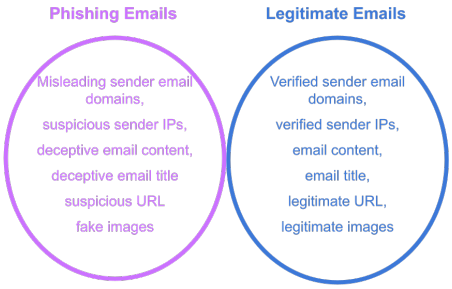

Figure 24. Disjoint Cluster

Clustering Algorithms Clustering algorithms are suited for organizing data into groups based on similarity, which helps uncover hidden patterns in the data, such as grouping emails into similar clusters to detect patterns and anomalies in email data [OIO+19]. For the examples, partitional and hierarchical clustering are selected. The output of partitional clustering can be presented as multiple disjoint clusters, each containing elements that are only associated with that cluster. Figure 24 illustrates an example of disjoint clusters. The data is divided into two clusters: 'Phishing Emails' and 'Legitimate Emails.' The 'Phishing Emails' cluster includes current and historical emails showing typical phishing characteristics such as fake URLs, suspicious sender IPs, misleading sender email domains, and deceptive email content. The current reported email is classified under the 'Phishing Emails' category due to several red flags: sender IP (Bhutanese IP address with phishing categorization by VirusTotal), sender email (support@microsafttt[.]com), email title ("!!!URGENT!!! LOGIN AND AGREE WITH NEW POLICY!!!"), the URL (microsafttt[.]com), and deceptive email content. The 'Legitimate Emails' cluster includes historical emails with legitimate sender information, the absence of suspicious URLs, and legitimate content.

Table 7. Dendrogram

Layer Number	**Example**
Layer 1	Layer categorizes the data based on sender information: the sender's IP address and email address.
Layer 2	Layer categorizes the data based on content analysis: email title, Microsoft logo image, text, and URL in the Email.
Layer 3	Layer categorizes the data based on target email and email processing: Recipient (user email), Initial Email Location (Inbox, Quarantine), and Final Email Location (Inbox, Quarantine).

The output of hierarchical clustering can be presented as a hierarchical tree structure or a dendrogram. This dendrogram represents the relationships between different clusters or subgroups of divided input data. For instance, Table 7 provides an example of how data is divided into three layers of the dendrogram. Due to data division by layers into more than 2 or 3 nodes, the example is presented as a table with a description of the categorizations of each layer. The first layer categorizes the data based on the origin of the email, focusing on the sender's IP address and email address. These attributes are crucial for identifying the sender's geographical (Bhutanese IP address with phishing categorization) and digital footprint. The second layer splits the data based on content analysis, including the Microsoft logo image, email title, text, and the malicious URL in the email. This layer aims to evaluate the content's intent to deceive by exploring the trusted and fake logos, analyzing the title ("!!!URGENT!!! LOGIN AND AGREE WITH NEW POLICY!!!"), email text, and embedded URLs (microsafttt[.]com). The final layer focuses on the target of the phishing attack and how the email is handled upon detection. It includes the recipient's email address, which is the target, and tracks the email's journey from the inbox to quarantine. Based on all this, the current email can be categorized as phishing.

Association Rules Association rules are suitable for uncovering relationships between various data points, such as the characteristics of emails, which helps identify and predict phishing attempts [YMA+23]. The output is presented as a set of association rules, which provide the dependencies between different data attributes. For example, three rules are provided in Table 8. The first rule shows the correlation between the sender's email domain attributes and phishing likelihood. Specifically, if the sender's email domain is not listed in trusted and whitelisted domains and contains keywords like "support" or "admin," the email is likely to be phishing. The second rule shows that emails originating from high-risk geolocations, such as emails with senders' IPs from countries known for

Table 8. Association Rules Examples

Rule Number	Rule Example
Rule 1	If the sender's email domain is not listed in trusted and whitelisted domains and contains keywords like "support" or "admin", then the email is hight likely to be phishing.
Rule 2	If the sender's IP address originates from a high-risk country, then the email is hight likely to be phishing.
Rule 3	If the email title contains words like "urgent" or "immediately" with "login", combined with a suspicious URL, then the email is hight likely phishing.

hosting phishing servers, can be associated with phishing. The last rule shows that the usage of urgent-sounding words in the email title, such as "urgent" or "immediately," combined with "login" with a suspicious URL, is highly related to email phishing. This rule highlights the manipulative language often used by attackers to provoke immediate action from the recipients.

6.3 Application of Supervised Deep Learning Methods

This section analyzes the applicability of supervised deep learning models to identify suitable methods for detecting phishing email. RNN is designed for tasks with sequential data, which could be applied to analyze temporal dependencies for understanding the context of phishing email [IBMb] [Pra23] [HAN20]. For instance, RNN can analyze the sequence of words in an email to understand its context and detect phishing attempts. The applicability of this method is proven in the conference paper "Catching the Phish: Detecting Phishing Attacks Using Recurrent Neural Networks (RNNs)" [HAN20] that proposes a novel automated system for mitigating phishing emails using RNNs. The study demonstrates that the flexibility of RNNs allows for continuous adaptation to new phishing trends, making RNNs an adaptable method for this case [HAN20]. CNN is designed for tasks involving grid-like topology data, such as image classification and text detection in emails [IBMa] [YND+18]. For phishing detection, CNN can identify phishing patterns in the email, such as specific phrases, word combinations, or how the URL is presented. Also, CNN can analyze Microsoft logo image for inconsistencies in phishing email. The application of this method for phishing email detection presented in the conference paper "Convolutional Neural Network Optimization for Phishing Email Classification" [MM21], which investigated CNN models for identifying phishing attacks through text analysis alone, achieving accuracy rates of over 98%.

In conclusion, employing both RNN and CNN to detect phishing email is suggested, as each method is suited to analyzing different aspects of email content. RNN is

applicable for analyzing sequential and contextual data, which helps examine the content and intent of email [IBMb] [Pra23]. CNN is applicable for classification and pattern recognition tasks, enabling them to analyze images and textual phishing patterns within emails [IBMa] [YND+18]. This analysis is followed by presenting output examples from the suggested methods for this case.

Recurrent Neural Network RNN is applied for classification and prediction tasks that involve analyzing sequential data, such as determining whether a current email is phishing or legitimate based on historical data [Pra23] [IBMb]. Also, RNN, depending on the given task, can be used for anomaly detection and pattern recognition [UM22] [Imr23]. The output could classify the email as phishing or legitimate based on historical and current data features. For this case, the RNN is trained on datasets containing both legitimate and phishing email cases to predict the likelihood of incoming emails being phishing or legitimate. Based on the patterns learned during training, the RNN classifies the current email by analyzing various indicators: a suspicious sender domain (microsafttt[.]com), an unusual sender IP location previously associated with phishing (Bhutan), a title that includes alarming symbols atypical for historical emails, and a suspicious URL. Therefore, the RNN classifies the current email as phishing.

Convolutional Neural Network CNN is suitable for classification tasks, such as email classification as phishing or normal, based on image and text data [YND+18] [IBMa]. Also, CNN is applied for pattern recognition and anomaly detection [AC20] [Imr23]. In this case, the CNN output can be presented as a classification of the email as phishing or normal based on image data. In this case, the input data is a Microsoft logo image from the email. CNN uses its convolutional layers to analyze the pixel arrays of the logo image to detect visual discrepancies from the authentic Microsoft logo. These layers utilize filters designed to identify anomalies in colors and shapes by comparing them against a verified database of official logos. The result of this analysis is a feature map highlighting areas of the image that deviate from the expected norms. Subsequently, the pooling layers reduce the complexity of the data by focusing on these discrepancies and summarizing the detected anomalies while keeping critical spatial hierarchies. After that, the fully connected layers use the filtered image features to classify the email as phishing or legitimate. This final classification tags the current email as phishing based on the deviations in the Microsoft logo image, which align with characteristics typically associated with phishing emails (e.g., unusual colors for the Microsoft logo).

6.4 Application of Unsupervised Deep Learning methods

This section surveys unsupervised deep learning algorithms' applicability for phishing detection cases. SAE is applicable for anomaly detection in emails as they re-

construct input data to capture essential features and identify deviations from normal patterns [BCR+23] [KAH21] [LTJY21]. Although SAEs require significant resources for training, their ability to manage the high-dimensional nature of email data is an advantage, enabling the identification of complex features and non-linear relationships within the data. Conference paper "Apply Stacked Auto-Encoder to Spam Detection" [MGT15] has demonstrated the suitability of SAEs in filtering spam, highlighting their classification accuracy and strength compared to traditional machine learning methods. GANs focus on generating synthetic samples, such as creating synthetic phishing emails to train detection models [YXXZ20] [HSS+20]. Research "Mitigation of Phishing Attacks using Generative Adversarial Networks (GAN)" [Uzo23] has proved that GANs, by generating synthetic phishing emails, can test the robustness of existing detection tools and mitigate real-time phishing attacks. In this case, Defender XDR was unable to detect a phishing email, which was then received in a user's inbox. This underscores the necessity of training systems to detect similar cases in the future, which also involves the generation of synthetic samples to recognize similar cases.

Overall, the suggested methods contain both SAE and GAN for phishing detection. SAEs are directly applicable for anomaly detection in incoming emails because they can reconstruct input data and identify deviations from normal patterns [BCR+23] [KAH21]. On the other hand, GANs provide strategic long-term advantages by enhancing the robustness of detection models by generating synthetic phishing emails for adversarial training, thus improving the detection of emails, similar to these synthetic examples [YXXZ20] [HSS+20]. This analysis is followed by presenting output examples from the suggested methods for this case.

Stacked Autoencoder Stacked Autoencoder is suitable for tasks that require anomaly detection and classification, such as classifying emails as phishing or legitimate based on deviations from reconstructed inputs that capture normative data behavior [BCR+23] [KAH21]. The output can be presented as a classification of email as phishing or legitimate, based on reconstruction error and anomaly score. In this scenario, a Stacked Autoencoder model is trained on historical datasets of legitimate and phishing emails. During the training phase, the model analyzes emails based on various features such as the time received, email header, sender's email address, sender IP address, email title, body text, embedded images, and URLs. The model then reconstructs the emails. Each email is subsequently assigned a specific reconstruction error and anomaly score. For this case, the differences between the input data (which includes an anomalous email address, an IP address associated with a high-risk location, an urgent email title and body text, misleading images, and a suspicious URL) and the reconstructed inputs result in a high anomaly score. This high anomaly score, indicating a significant deviation from the learned patterns of historically legitimate emails, categorizes the current email as phishing.

Generative Adversarial Network GAN is suitable for penetration testing, enhancing the detection system's robustness against new attacks, including identifying previously unknown phishing email models [YXXZ20] [GZ19]. For this case, GAN is utilized for penetration testing and enhancing phishing email detection capabilities because Defender XDR cannot detect this phishing email and isolate it. The Generator component of the GAN method creates synthetic phishing emails with sender email, sender IP address from Bhutan, email titles with alarming symbols, fake Microsoft logos, and embedded URLs from actual detection cases. These fake phishing emails allow the system to analyze and learn to identify new and developing phishing tactics without the risk of compromising real sensitive data. The final result of this method is an improved detection model that is more robust and adaptive to new phishing attacks.

6.5 Summary

This section analyzes 10 machine learning and deep learning methods across four categories: supervised machine learning, unsupervised machine learning, supervised deep learning, and unsupervised deep learning. Specific methods from each category were considered for their applicability to a phishing detection scenario by assessing their strengths and weaknesses. Based on the analysis, 9 methods are recommended to be applied to four criteria: classification, pattern recognition, anomaly detection, and penetration testing. The example outputs for the methods applicable to the criteria provided are shown in Table 9.

For classification, the recommended methods from the supervised machine learning category include random forest, SVM, and logistic regression [LW12]. In supervised deep learning, RNN and CNN are applicable methods. SAE, though typically used in unsupervised contexts, can also be adapted for classification tasks [AK19] [LPL23]. For pattern recognition, clustering algorithms and association rules from unsupervised machine learning are advised [Ed20]. Additionally, RNN and CNN from supervised deep learning are also suggested, as the specific application of these neural networks depends on the tasks they are designed for [Imr23]. In anomaly detection, RNN and CNN from supervised deep learning are recommended [UM22] [AC20]. From unsupervised deep learning, SAE is advised for its ability to reconstruct data and identify deviations from normal patterns [KAH21]. For penetration testing, GAN is recommended to apply for adversarial training, which helps the system learn to detect and prevent new phishing techniques [GZ19].

Table 9. Methods' Output for Criteria

ML/DL methods	**Classification**	**Patterns Recognition**	**Anomaly Detection**	**Penetration Testing**
Random Forest	Phishing Attempt, Suspicious Sender Domain			
Support Vector Machines	Microsoft Impersonation, Credential Theft Attempt			
Logistic Regression	Microsoft Impersonation, Suspicious Sender			
Clustering Algorithms		Phishing Keyword Content Cluster,		
Association Rules		Quarantine Movement Pattern, Urgent Phishing Keyword Usage		
Recurrent Neural Network	Fake Policy Update, Phishing Email Content, Malicious URL	Phishing Email Pattern, Fake Microsoft Email	Anomalous Email Content, Atypical Email Address	
Convolutional Neural Network	Fake Logo Image, Fake Policy Update	Email Header Irregularities, Logo Image Misuse	Deviation from Normal Email, Abnormal Email Source	
Stacked Autoencoder	Suspicious Sender Domain, Suspicious URL		Anomalous Email Header, Abnormal Login Page	
Generative Adversarial Network				Fake Emails Creation, System Testing

7 Concluding Remarks

The thesis analyzes the applicability of machine learning and deep learning methods in addressing IT security threats across three cases: UEBA, vulnerability detection, and phishing detection. The ML and DL methods were observed and assessed based on their applicability in classifying, recognizing patterns, detecting anomalies, and penetration testing. By assessing each method's strengths and limitations for the three threat detection cases, the thesis answers the research question and identifies recommended ML and DL techniques to be applied for each analyzed security threat.

7.1 Limitations

Limitations observed in the thesis include the analysis of a limited set of machine learning and deep learning methods. While this thesis analyzed 10 ML and DL methods, new methods continue to develop. The thesis focused on analyzing the applicability of individual methods for 3 specific threat detection cases. However, AI employs a combination of methods across threat detection. Additionally, although the thesis assessed the applicability of methods based on their characteristics, strengths, and weaknesses, not all methods' adaptability for 3 threat detection cases was supported by practical application proof provided by researchers and studies.

7.2 Answer to Research Question

Based on the thesis analysis, the research question — "What machine learning and Deep Learning methods are applicable for detecting different security threats?"—is addressed by identifying specific ML and DL methods' applicability to security threats. The number of security threats in the real world is large, and each case contains unique elements; the thesis analyzes ML and DL methods' applicability for the 3 security scenarios. These scenarios include User and Entity Behavior Analytics, Vulnerability Detection, and Phishing Detection, each presenting unique characteristics requiring individual method recommendations.

For the UEBA scenario, it is recommended that random forest, SVM, logistic regression, and KNN be applied to classify events. It also highlights the use of RNN and SAE from the deep learning categories for classification. Clustering algorithms, association rules, and RNN are applicable to uncover hidden patterns in user behavior. Anomaly detection in this scenario could be performed by recommended RNN and SAE to identify anomalies by deviations from normal behavior.

In the vulnerability detection scenario, methods such as random forest, SVM, logistic regression, RNN, and SAE are recommended for classifying potential vulnerabilities. Clustering algorithms, association rules, and RNN are suggested for pattern recognition.

SAE and RNN are suggested for identifying deviations that may indicate anomalies and following security breaches.

For phishing detection case, the thesis recommends employing random forest, SVM, logistic regression, RNN and CNN for classification. Also, RNN and CNN, along with clustering algorithms and association rules, are suggested for pattern recognition. For anomaly detection suggested the application of RNN, CNN, and SAE, which can identify unusual patterns indicative of phishing attempts. In this case, GAN is proposed for penetration testing to enhance the system's resilience against new and developing phishing techniques.

7.3 Future Work

Future research will focus on developing a mixed-methods approach, combining the most applicable ML and DL techniques for threat detection. This will involve integrating new ML and DL methods, ensuring that the mixed-methods approach stays up-to-date with the latest advancements in AI. Furthermore, practical implementation involves creating a framework based on the mixed-methods approach that not only detects threats but prioritizes them based on their potential impact and criticality. This capability of creating a framework will enable security professionals to react to critical threats first, enhancing the overall security posture.

References

[AA14] Andronicus A. Akinyelu and Aderemi O. Adewumi. Classification of phishing email using random forest machine learning technique. *Volume 2014*, Article ID 425731, 2014. Open Access.

[AACM20] Giovanni Apruzzese, Mauro Andreolini, Michele Colajanni, and Mirco Marchetti. Hardening random forest cyber detectors against adversarial attacks. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 4(4):427–439, August 2020.

[AC20] Montdher Alabadi and Yuksel Celik. Anomaly detection for cybersecurity based on convolution neural network : A survey. In *2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, pages 1–14, 2020.

[Ahm19] Hassan Ahmad. What is ai and will it affect me as a gp?, February 4 2019. Retrieved April 15, 2024, from https://www.linkedin.com/pulse/what-ai-affect-me-gp-hassan-ahmad/.

[AK19] Kemal ADEM and Serhat KILIÇARSLAN. Performance analysis of optimization algorithms on stacked autoencoder. In *2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, pages 1–4, 2019.

[ALMdO+23] Giovanni Apruzzese, Pavel Laskov, Edgardo Montes de Oca, Wissam Mallouli, Luis Brdalo Rapa, Athanasios Vasileios Grammatopoulos, and Fabio Di Franco. The role of machine learning in cybersecurity. *Digital Threats*, 4(1), mar 2023.

[AMB21] Alankrita Aggarwal, Mamta Mittal, and Gopi Battineni. Generative adversarial network: An overview of theory and applications. *International Journal of Information Management Data Insights*, 1(1):100004, 2021.

[AMK+20] Javed Asharf, Nour Moustafa, Hasnat Khurshid, Essam Debie, Waqas Haider, and Abdul Wahab. A review of intrusion detection systems using machine and deep learning in internet of things: Challenges, solutions and future directions. *Electronics*, 9(7), 2020.

[AMM24] Pierpaolo Artioli, Antonio Maci, and Alessio Magrì. A comprehensive investigation of clustering algorithms for user and entity behavior analytics. *Frontiers in Big Data*, 7, 2024.

[AWAV23] Kholoud Althobaiti, Maria K. Wolters, Nawal Alsufyani, and Kami Vaniea. Using clustering algorithms to automatically identify phishing campaigns. *IEEE Access*, 11:96502–96513, 2023.

[A9] Berke Akkaya and Nurdan Çolakoğlu. Comparison of multi-class classification algorithms on early diagnosis of heart diseases. In *y-BIS 2019 Conference: ISBIS Young Business and Industrial Statisticians Workshop on Recent Advances in Data Science and Business Analytics*, Istanbul, Turkey, September 2019.

[Bah22] Pratik Baheti. Pattern recognition in machine learning [basics & examples], September 13 2022. Retrieved May 4, 2024, from https://www.v7labs.com/blog/pattern-recognition-guide: :text=Pattern

[Bar97] Andrew G. Barto. Chapter 2 - reinforcement learning. In Omid Omidvar and David L. Elliott, editors, *Neural Systems for Control*, pages 7–30. Academic Press, San Diego, 1997.

[Bat23] Chandu P. Bathula. Understanding classification, regression, and clustering in machine learning: Machine learning concept 83, June 7 2023. Retrieved May 4, 2024, from https://medium.com/@chandu.bathula16/understanding-classification-regression-and-clustering-in-machine-learning-machine-learning-8b77b4b27c87.

[BBC23] BBC Wildlife Magazine. Giant panda guide: why they're threatened, how they raise young and captive breeding, April 12 2023. Retrieved January 28, 2024, from https://www.discoverwildlife.com/animal-facts/mammals/giant-pandas-facts.

[BBCV21] Sidahmed Benabderrahmane, Ghita Berrada, James Cheney, and Petko Valtchev. A rule mining-based advanced persistent threats detection system, 2021.

[BCR+23] Sundaravadivazhagan Balasubaramanian, Robin Cyriac, Sahana Roshan, Kulandaivel Maruthamuthu Paramasivam, and Boby Chellanthara Jose. An effective stacked autoencoder based depth separable convolutional neural network model for face mask detection. *Array*, 19:100294, 2023.

[Bre01] L. Breiman. Random forests. *Machine Learning*, 45:5–32, 2001.

[CN17] Oskar Carlsson and Daniel Nabhani. User and entity behavior anomaly detection using network traffic, 2017.

[CSD12] Gajendra Singh Chandel Chandrapal Singh Dangi, Ravindra Gupta. Cyber security approach in web application using svm. *International Journal of Computer Applications*, 57(20):30–34, November 2012.

[CV95] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995.

[DALM+21] Gabriel Dulac-Arnold, Natasha Levine, Daniel J. Mankowitz, et al. Challenges of real-world reinforcement learning: definitions, benchmarks and analysis. *Machine Learning*, 110:2419–2468, 2021.

[DDG+23] Pratik Dhaygude, Nilesh Dhulshette, Omkar Ganjale, Umesh Sawant, and Pranali Navghare. A review paper on different deep learning methodologies for user and entity behavior analytics (ueba). *International Journal of Research Publication and Reviews*, 4(5):3757–3762, May 2023.

[Ed20] Khadija El Bouchefry and Rafael S. de Souza. Chapter 12 - learning in big data: Introduction to machine learning. In Petr Škoda and Fathalrahman Adam, editors, *Knowledge Discovery in Big Data from Astronomy and Earth Observation*, pages 225–249. Elsevier, 2020.

[EM17] Thomas W. Edgar and David O. Manz. Chapter 4 - exploratory study. In Thomas W. Edgar and David O. Manz, editors, *Research Methods for Cyber Security*, pages 95–130. Syngress, 2017.

[FZH+19] Yong Fang, Cheng Zhang, Cheng Huang, Liang Liu, and Yue Yang. Phishing email detection using improved rcnn model with multilevel vectors and attention mechanism. *IEEE Access*, 7:56329–56340, 2019.

[Gee] GeeksforGeeks. Unsupervised neural network models. Retrieved April 29, 2024, from `https://www.geeksforgeeks.org/unsupervised-neural-network-models/`.

[Gee23] GeeksforGeeks. Introduction to recurrent neural network, December 4 2023. Retrieved January 26, 2024, from https://www.geeksforgeeks.org/introduction-to-recurrent-neural-network/.

[Gon23] Hari Gonaygunta. Machine learning algorithms for detection of cyber threats using logistic regression. *International Journal of Smart Sensor and Adhoc Network*, 3(4):36–42, January 2023.

[GZ19] Liang Gonog and Yimin Zhou. A review: Generative adversarial networks. In *2019 14th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, pages 505–510, 2019.

[HAN20] Lukáš Halgaš, Ioannis Agrafiotis, and Jason R. C. Nurse. *Catching the Phish: Detecting Phishing Attacks Using Recurrent Neural Networks (RNNs)*, page 219–233. Springer International Publishing, 2020.

[Hol22] Alex Hollister. 3 major threat detection methods explained, December 14 2022. Retrieved May 2, 2024, from https://www.helpnetsecurity.com/2022/12/14/3-major-threat-detection-methods-explained/.

[HSS+20] Srinidhi Hiriyannaiah, A.M.D. Srinivas, Gagan K. Shetty, Siddesh G.M., and K.G. Srinivasa. Chapter 4 - a computationally intelligent agent for detecting fake news using generative adversarial networks. In Siddhartha Bhattacharyya, Václav Snášel, Deepak Gupta, and Ashish Khanna, editors, *Hybrid Computational Intelligence*, Hybrid Computational Intelligence for Pattern Analysis and Understanding, pages 69–96. Academic Press, 2020.

[IBMa] IBM. What are convolutional neural networks? Retrieved April 18, 2024, from https://www.ibm.com/topics/convolutional-neural-networks.

[IBMb] IBM. What are recurrent neural networks? Retrieved April 18, 2024, from https://www.ibm.com/topics/recurrent-neural-networks.

[IBMc] IBM. What is a decision tree? Retrieved April 20, 2024, from `https://www.ibm.com/topics/decision-trees`.

[IBMd] IBM. What is logistic regression? Retrieved March 5, 2024, from https://www.ibm.com/topics/logistic-regression.

[IBMe] IBM. What is penetration testing? Retrieved May 4, 2024, from https://www.ibm.com/topics/penetration-testing.

[IBMf] IBM. What is random forest? Retrieved April 20, 2024, from `https://www.ibm.com/topics/random-forest`.

[IBMg] IBM. What is the k-nearest neighbors (knn) algorithm? Retrieved March 12, 2024, from https://www.ibm.com/topics/knn.

[IBM23a] IBM. What are support vector machines (svms)?, December 27 2023. Retrieved April 18, 2024, from https://www.ibm.com/topics/support-vector-machine.

[IBM23b] IBM. What is anomaly detection?, December 12 2023. Contributors: Joel Barnard, Cole Stryker. Retrieved May 4, 2024, from https://www.ibm.com/topics/anomaly-detection.

[Imr23] Inas Ismael Imran. Cnn-rnn deep learning networks for pattern recognition problems. In *2023 International Conference on Business Analytics for Technology and Security (ICBATS)*, pages 1–5, 2023.

[KAH21] B.T. Kinh, D.T. Anh, and D.N. Hieu. A comparison between stacked auto-encoder and deep belief network in river run-off prediction. In P.C. Vinh and A. Rakib, editors, *Context-Aware Systems and Applications, and Nature of Computation and Communication*, volume 343 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*. Springer, Cham, 2021.

[KIJ13] Mahmoud Khonji, Youssef Iraqi, and Andrew Jones. Phishing detection: A literature survey. *IEEE Communications Surveys Tutorials*, 15(4):2091–2121, 2013.

[KR23] N. Kamal and S. Raheja. Prediction of software vulnerabilities using random forest regressor. In A. Shukla, B.K. Murthy, N. Hasteer, and JP. Van Belle, editors, *Computational Intelligence*, volume 968 of *Lecture Notes in Electrical Engineering*, Singapore, 2023. Springer.

[Leo98] Andrew C. Leon. 3.12 - descriptive and inferential statistics. In Alan S. Bellack and Michel Hersen, editors, *Comprehensive Clinical Psychology*, pages 243–285. Pergamon, Oxford, 1998.

[Li22] Yuxi Li. Reinforcement learning in practice: Opportunities and challenges, 2022.

[LMM+21] Sang-Woong Lee, Haval Mohammed sidqi, Mokhtar Mohammadi, Shima Rashidi, Amir Masoud Rahmani, Mohammad Masdari, and Mehdi Hosseinzadeh. Towards secure intrusion detection systems using deep learning techniques: Comprehensive analysis and review. *Journal of Network and Computer Applications*, 187:103111, 2021.

[LPL23] Pengzhi Li, Yan Pei, and Jianqiang Li. A comprehensive survey on design and application of autoencoder in deep learning. *Applied Soft Computing*, 138:110176, 2023.

[LRC20] Liu, Zhiguo, Ren, Changqing, and Cai, Wenzhu. Overview of clustering analysis algorithms in unknown protocol recognition. *MATEC Web Conf.*, 309:03008, 2020.

[LTJY21] Zhichao Li, Li Tian, Qingchao Jiang, and Xuefeng Yan. Distributed-ensemble stacked autoencoder model for non-linear process monitoring. *Information Sciences*, 542:302–316, 2021.

[LV02] Yihua Liao and V.Rao Vemuri. Use of k-nearest neighbor classifier for intrusion detection11an earlier version of this paper is to appear in the proceedings of the 11th usenix security symposium, san francisco, ca, august 2002. *Computers Security*, 21(5):439–448, 2002.

[LW12] Qiang Liu and Yiming Wu. Supervised learning. In Norbert M. Seel, editor, *Encyclopedia of the Sciences of Learning*. Springer, Boston, MA, 2012.

[Mat22] Vidhi Mathur. Association rule mining: Importance and steps, September 26 2022. Retrieved April 1, 2024, from https://www.analyticssteps.com/blogs/association-rule-mining-importance-and-steps.

[MB22] Andrew Mkwashi and Irina Brass. The future of medical device regulation and standards: Dealing with critical challenges for connected, intelligent medical devices. *Zenodo*, 2022.

[MC11] Nikolai Mansourov and Djenana Campara. Chapter 3 - how to build confidence. In Nikolai Mansourov and Djenana Campara, editors, *System Assurance*, The MK/OMG Press, pages 49–80. Morgan Kaufmann, Boston, 2011.

[McC23] Shaun McCullough. Threat detection: Stopping threats before they become a problem, May 8 2023. Retrieved May 2, 2024, from https://www.sans.org/blog/what-is-threat-detection/.

[MGT15] Guyue Mi, Yang Gao, and Ying Tan. Apply stacked auto-encoder to spam detection. In Ying Tan, Yuhui Shi, Fernando Buarque, Alexander Gelbukh, Swagatam Das, and Andries Engelbrecht, editors, *Advances in Swarm and Computational Intelligence*, pages 3–15, Cham, 2015. Springer International Publishing.

[MIA24] Raimundas Matulevičius, Mubashar Iqbal, and Abasi-Amefon Obot Affia. Research methods in cybersecurity (ltat.05.028), February 22 2024. Spring semester 2023/24. Retrieved May 10, 2024, from https://courses.cs.ut.ee/2024/RMCs/spring/Main/Lectures.

[MM21] Cameron McGinley and Sergio A. Salinas Monroy. Convolutional neural network optimization for phishing email classification. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 5609–5613, 2021.

[MW12] Wenji Mao and Fei-Yue Wang. Chapter 8 - cultural modeling for behavior analysis and prediction. In Wenji Mao and Fei-Yue Wang, editors, *New Advances in Intelligence and Security Informatics*, pages 91–102. Academic Press, Boston, 2012.

[Nad16] Prakash Nadkarni. Chapter 10 - core technologies: Data mining and "big data". In Prakash Nadkarni, editor, *Clinical Research Computing*, pages 187–204. Academic Press, 2016.

[NAT19] NATIONAL VULNERABILITY DATABASE. Cve-2017-11771 detail, October 2 2019. Retrieved March 22, 2024, from https://nvd.nist.gov/vuln/detail/CVE-2017-11771.

[NG23] Majeed Jasim Nabet and Loay E. George. Phishing attacks detection by using support vector machine. *Journal of Al-Qadisiyah for Computer Science and Mathematics*, 15(2):Comp Page . 180–189, Sep. 2023.

[OIO+19] Jelili Oyelade, Itunuoluwa Isewon, Olufunke Oladipupo, Onyeka Emebo, Zacchaeus Omogbadegun, Olufemi Aromolaran, Efosa Uwoghiren, Damilare Olaniyan, and Obembe Olawole. Data clustering: Algorithms and its applications. In *2019 19th International Conference on Computational Science and Its Applications (ICCSA)*, pages 71–81, 2019.

[Ope24] OpenAI. Chatgpt (gpt-4). `https://chatgpt.com`, 2024. (April 2024 version) [large language model].

[PH13] TANG Zan-Yu] [PENG Hua, MO Li-Ping. Construction of a svm learning model in the categorization framework for cve. *Journal of Jishou University(Natural Sciences Edition)*, 34(4):62–66, 2013.

[Pra23] Prajeesh Prathap. Feed-forward and recurrent neural networks: The future of machine learning, July 20 2023. Retrieved March 3, 2024, from https://medium.com/@prajeeshprathap/feed-forward-and-recurrent-neural-networks-the-future-of-machine-learning-8b2c1975f0c5.

[PSSS20] Darshan Jagannath Pangarkar, Rajesh Sharma, Amita Sharma, and Madhu Sharma. Assessment of the different machine learning models for prediction of cluster bean (*cyamopsis tetragonoloba* l. taub.) yield. *Advances in Research*, 21(9):98–105, 2020. Article no. AIR.60327.

[RRM20] Christian Reimers and Christian Requena-Mesa. Chapter 13 - deep learning – an opportunity and a challenge for geo- and astrophysics. In Petr Škoda and Fathalrahman Adam, editors, *Knowledge Discovery*

in Big Data from Astronomy and Earth Observation, pages 251–265. Elsevier, 2020.

[RVS18] Mohammed Harun Babu R, Vinayakumar R, and Soman Kp. Rnnsecurenet: Recurrent neural networks for cybersecurity use-cases. 2018.

[Sab23] John Sabo. Machine learning and ai in soc: Enhancing threat detection, October 28 2023. Retrieved May 2, 2024, from https://www.linkedin.com/pulse/machine-learning-ai-soc-enhancing-threat-detection-john-sabo-9syqc/.

[SBZH21] Lihua Su, Wenhua Bai, Zhanghua Zhu, and Xuan He. Research on application of support vector machine in intrusion detection. *Journal of Physics: Conference Series*, 2037(1):012074, sep 2021.

[SK19] Jérémie Sublime and Ekaterina Kalinicheva. Automatic post-disaster damage mapping using deep-learning techniques for change detection: Case study of the tohoku tsunami. *Remote Sensing*, 11(9), 2019.

[SKK23] Rohini Srivastava, Shailesh Kumar, and Basant Kumar. 7 - classification model of machine learning for medical data analysis. In Tilottama Goswami and G.R. Sinha, editors, *Statistical Modeling in Machine Learning*, pages 111–132. Academic Press, 2023.

[SMD+23] Karthikeyan Saminathan, Sai Tharun Reddy Mulka, Sangeetha Damodharan, Rajagopal Maheswar, and Josip Lorincz. An artificial neural network autoencoder for insider cyber security threat detection. *Future Internet*, 15(12), 2023.

[SW19] Chawin Sitawarin and David Wagner. On the robustness of deep k-nearest neighbors, 2019.

[Tay23] Mohammad Mustafa Taye. Theoretical understanding of convolutional neural network: Concepts, architectures, applications, future directions. *Computation*, 11(3), 2023.

[TGRD23] Sumaiya Tasneem, Kishor Datta Gupta, Arunav Roy, and Dipankar Dasgupta. Generative adversarial networks (gan) for cyber security: Challenges and opportunities. In *2022 IEEE Symposium Series On Computational Intelligence*, Singapore, 2023. Kennesaw State University, Clark Atlanta University, The University of Memphis.

[TMG23] H. Torabi, S.L. Mirtaheri, and S. Greco. Practical autoencoder based anomaly detection by using vector reconstruction error. *Cybersecurity*, 6(1):1, 2023.

[UM22] Imtiaz Ullah and Qusay H. Mahmoud. Design and development of rnn anomaly detection model for iot networks. *IEEE Access*, 10:62722–62750, 2022.

[Uzo23] Chibuike Israel Uzoagba. Mitigation of phishing attacks using generative adversarial networks (gan). Research Proposal, May 2023.

[VE04] H.S. Venter and Jan H.P. Eloff. Categorizing vulnerabilities using data clustering techniques. *University of Pretoria*, January 2004. PDF Available.

[vEH20] Joaquin Vanschoren van Engelen and Holger H. Hoos. A survey on semi-supervised learning. *Machine Learning*, 109:373–440, 2020.

[Wan19] Lishan Wang. Research and implementation of machine learning classifier based on knn. *IOP Conference Series: Materials Science and Engineering*, 677:052038, 2019.

[WDCP22] Mohammad Wazid, Ashok Kumar Das, Vinay Chamola, and Youngho Park. Uniting cyber security and machine learning: Advantages, challenges and future research. *ICT Express*, 8(3):313–321, 2022.

[WLZ22] Shiyu Wang, Zehao Li, and Xiaotian Zhao. The application of convolutional neural network in malware images classification. In *Proceedings of the 2021 International Conference on Public Art and Human Development (ICPAHD 2021)*, pages 240–245. Atlantis Press, 2022.

[YJ21] Rasheed Yousef and Mahmoud Jazzar. Measuring the effectiveness of user and entity behavior analytics for the prevention of insider threats. *Journal of Xi'an University of Architecture Technology*, XIII(X):175–181, 2021.

[YLL23] H. Yang, L. Li, and K. Li. Real-time progressive compression method of massive data based on improved clustering algorithm. *Cluster Computing*, 26:3781–3791, 2023.

[YMA+23] S. Yacoubi, G. Manita, H. Amdouni, et al. A modified multi-objective slime mould algorithm with orthogonal learning for numerical association rules mining. *Neural Computing & Applications*, 35:6125–6151, 2023.

[YND+18] Ryo Yamashita, Mizuho Nishio, Ron K. G. Do, et al. Convolutional neural networks: An overview and application in radiology. *Insights into Imaging*, 9:611–629, 2018.

[YQW+20] Yu Yan, Lin Qi, Jie Wang, Yun Lin, and Lei Chen. A network intrusion detection method based on stacked autoencoder and lstm. In *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, pages 1–6, 2020.

[YXXZ20] Dong Yang, Tao Xiong, Daguang Xu, and S. Kevin Zhou. Chapter 7 - segmentation using adversarial image-to-image networks. In S. Kevin Zhou, Daniel Rueckert, and Gabor Fichtinger, editors, *Handbook of Medical Image Computing and Computer Assisted Intervention*, The Elsevier and MICCAI Society Book Series, pages 165–182. Academic Press, 2020.

[Zai18] Houda Zaidi. Supervised deep learning with keras, tensorflow and theano, January 24 2018. Retrieved April 30, 2024, from https://medium.com/@zaidi.houd/supervised-deep-learning-with-keras-tensorflow-and-theano-9dfd4fa17358.

[ZMC23] Rasha Zieni, Luisa Massari, and Maria Carla Calzarossa. Phishing or not phishing? a survey on the detection of phishing websites. *IEEE Access*, 11:18499–18519, 2023.

[ZPZ+20] Jianyun Zheng, Jianmin Pang, Xiaochuan Zhang, Xin Zhou, MingLiang Li, and Jun Wang. Recurrent neural network based binary code vulnerability detection. In *Proceedings of the 2019 2nd International Conference on Algorithms, Computing and Artificial Intelligence*, ACAI '19, page 160–165, New York, NY, USA, 2020. Association for Computing Machinery.

Appendix

I. Glossary

- **ML** - Machine learning
- **DL** - Deep learning
- **AI** - Artificial intelligence
- **IT** - Information technology
- **UEBA** - User and Entity Behavior Analytics
- **IDS** - Intrusion detection systems
- **SVM** - Support vector machines
- **KNN** - K-nearest neighbor
- **GAN** - Generative adversarial network
- **CNN** - Convolutional neural network
- **RNN** - Recurrent neural network
- **SAE** - Stacked autoencoder
- **AE** - Autoencoder
- **PGF** - Please provide the full form for PGF
- **SOC** - Security Operations Center
- **MAC address** - Media Access Control address
- **URL** - Uniform Resource Locator
- **CVE** - Common Vulnerabilities and Exposures
- **IP** - Internet Protocol

II. Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, **Diana Šramova**,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to

   reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   A Survey of Machine Learning Methods and their Applicability for Security Analysis,

   supervised by Raimundas Matulevičius.

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Diana Šramova
15/05/2024