КІБЕРБЕЗПЕКА: освіта, наука, техніка
CYBERSECURITY: EDUCATION, SCIENCE, TECHNIQUE
№ 4 (28), 2025
ISSN 2663 - 4023
DOI 10.28925/2663-4023.2025.28.774
UDC 004.056

Taras Melko
PhD Candidate, Department of Computer Science and Software Engineering
Private Higher Education Establishment "European University", Kyiv, Ukraine
ORCID ID: 0009-0005-7295-5863
tarasxmelko@gmail.com

Volodymyr Kotsun
PhD Tech, Associate Professor of the
Department of Computer Science and Software Engineering
Private Higher Education Establishment "European University", Kyiv, Ukraine
ORCID ID: 0000-0003-2363-8157
volodumur.kotsun@e-u.edu.ua

THEORETICAL AND TECHNICAL ASPECTS OF MACHINE LEARNING USAGE IN CYBERSECURITY

Abstract. The article explores the technical and theoretical aspects of machine learning (ML) in addressing the escalating complexities of cybersecurity threats in the digital age since the ever-growing rise in cybercrime has prompted users to utilize newer approaches to raise the bar on cybersecurity. Research considers the adoption of machine learning (ML) technology as a cornerstone of virtually any contemporary problem in cyber security, particularly processes and techniques involved in problem analysis, detection, attack prediction, and even behavioral profiling. Elaborated on how ML makes a better response compared to traditional methods like signature-based detection by explaining how real-time analysis of massive data becomes possible. An overview of the important features of supervised and unsupervised learning is provided in the context of anomaly detection and malicious activity recognition with a focus on Support Vector Machine and Isolation Forests algorithms as well as a detailed look at the LSTM model for phishing URL evolution analysis. Also, those algorithms have been highlighted from the technical implementation side: supervised learning with Support Vector Machines using Scikit-Learn to classify network traffic trained on features like IP addresses and ports, unsupervised learning with Isolation Forests for anomaly detection in multidimensional data, and deep learning with Long Short-Term Memory (LSTM) networks for phishing URL analysis. This paper investigates significantly important difficulties in carrying out ML algorithms, such as class imbalance, adversarial attacks, and lack of model transparency. Such techniques as SMOTE (Synthetic Minority Over-sampling Technique) are proposed for developing training datasets, whereas model adversarial training and robust optimization methods are suggested for defense against malicious model exploitation. Also, the role of explainability methods such as SHAP and LIME are emphasized to build the trust and acceptance of automated ML systems in cybersecurity. Identified research opportunities and suggested that further testing be done on improving model robustness and performance metrics in constrained environments.

Keywords: cybersecurity; cyberattack; cyberdefense; machine learning; deep learning; machine learning in cybersecurity.

INTRODUCTION

A massive flow of information has emerged, introducing new complexities along with advancing the amount of cyber attacks. It's not just a virus or a phishing email; there are far greater and more intricate, cunning assaults on pivotal infrastructure units such as electric grids, banks, plants, or moreover state institutions. The digital revolution has indeed opened plenty of opportunities and it certainly does put us at more risk. Cybercriminals inventing more and more threats every single day, and countries have begun to use cyberattacks as weapons in cyber wars [1]. Outdated security measures, which include signature tracking and manual rule setting, have

long gone outmatched by the speed and magnitude of the growth of the cyber world. This makes new measures such as machine learning (ML) more practical. This proposes new techniques for efficient data examination, classification, and real-time anomaly detection.

Machine learning is a very important component of artificial intelligence that works on the design and development of self-learning algorithms. Through experience, self-learning algorithms enhance their results, usually without further programmed input intervention by enabling automated improvement of output records [2]. In the world of digital dangers, where hackers are consistently developing new and improving old models, the adaptive capability of machine learning to new unknown threats is remarkable. Machine learning systems can automatically identify traffic behavior patterns, detect anomalous activity atypical to ordinary users, and classify malware based on its features. Statistical methods and techniques using machine learning, which are discussed in the book by Gareth James and other coauthors, make it possible. Here they explain that regression methods, also logistic regression, trees of decision, regression, and clustering (k-means), can be called for more compound multidimensional data sets [3]. These methods are applied in cybersecurity for the sake of developing models that identify atypical anomalies in system logs, apportion probabilistic values of attacks, and classify user actions for proactive detection of internal system threats.

As Friedman and Singer note, “Attacks are no longer limited to simple viruses; they include social engineering, zero-day exploits, and coordinated campaigns that adapt to the victim’s defenses” [1]. In this case, cyber threats are multifaceted and accompanied by different kinds of risk factors. It follows that cybersecurity systems are expected not only to mitigate established threats, but also to anticipate new ones leveraging big datasets from software signatures, network logs, and even user activities. Such circumstances make machine learning particularly useful as its algorithms can generalize even with noisy data from real-life situations [2]. For instance, such classifiers like Support Vector Machines and Random Forest can detect malware by analyzing its binary code or calls to its external interfaces [3]. In this manner, machine learning approaches assist in responding to well-known attacks and at the same time detecting new, unfamiliar, undocumented threats.

Even though machine learning offers numerous advantages, its incorporation in cybersecurity still poses several issues to consider. The model’s performance relies on various factors like the quality of input data, the appropriate configuration selection, and most importantly, the necessary computational resources to enable deep and efficient learning [3]. Developers have to deal with noisy information, unnecessary positive results, and data streams that cannot be handled as needed. Furthermore, machine learning models are prone to what is known as adversarial learning attack vulnerabilities. In this scenario, algorithms are tricked by attackers who create an additional layer of complexity by deliberately manipulating data, which is quite difficult for cybersecurity system developers to overcome [2]. All in all, machine learning in cybersecurity offers options for sophisticated data and hidden cyber threat analysis, but with these advances comes the responsibility to countermeasure techniques, so that they can cope with the gradual progression of cybercrime [1].

BASICS OF MACHINE LEARNING

Three fundamental methods of machine learning are mainly separated into three branches: supervised, unsupervised, and reinforcement learning. As of now, modern cybersecurity approaches make use of the first two types of learning systems.

In supervised learning, a labeled dataset is used. In other words, it contains marked instances where each input (features of the object) has an associated label (output). An essential part of this

kind of learning is to train the model to label new data. Classifying (spam versus not spam) and regression (predicting numerical values) tasks are examples of how they can be utilized. The definition of "learning with a teacher" has to do with how the data is structured, in which case it is given as (□, □(□)) the objective being to evaluate the function □ [2]. It means that linear regression or logistic regression is exposed to labeled data in order to lessen the Mean Square Error (MSE) of the so-called prediction error by taking into account the generalization to new data [3]. In this perspective, the model evaluation through cross-validation is crucial. Concerning deep learning, it is common that during the training procedure of the deep neural networks, they are to some degree supervised: "The backpropagation algorithm is used to optimize the parameters. The gradient descent method is fundamental for training deep models" [4]. The primary application of supervised learning is the classification of malware based on labeled data.

In contrast to supervised learning, unsupervised learning works with unlabeled data, aiming to find hidden structures, patterns, or groups. It is applied in anomaly detection within network traffic, where normal behavior is not explicitly labeled. Clustering (grouping similar objects) and dimensionality reduction (simplifying data while preserving important information) are considered the most common tasks using unsupervised learning. Tom Mitchell describes such tasks as "finding structure in data without explicit labels" [2]. As for example, he cites clustering and learning associative rules.

Deep learning has been mostly considered as a branch of machine learning where multi-layered neural networks model data with complex relationships. It is especially effective with images, texts, audio, and other categories of unstructured data. The impressive results achieved in recognition activities stem from deep neural networks using automatic detection of a feature's hierarchy within the data [4]. They give a detailed account of the architectures of Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN), primarily focused on user behavior analysis with recurrent networks or malware identification via convolutional neural networks.

MACHINE LEARNING IN CYBERSECURITY

Cybersecurity has been defined as a key tool in the field of cybersecurity with the advent of new cyber threats. Every new hyper-connected era enhances ML adoption. Over the last decade, automated systems based on ML techniques have rapidly scaled, complimenting traditional systems, improving overall threat detection, and speeding response times. One of the main reasons is that algorithms like Random Forest or Neural Network techniques are capable of analyzing huge datasets, deriving hidden patterns, and addressing newly emerged challenges. The growing problem of phishing attacks is eliminated with the help of Natural Language Processing (NLP) models like BERT that examine text in emails or URLs looking for patterns including misspelled fake domains [1]. ML enables the automation of these processes, significantly reducing the response time to threats, which is critical in today's digital world.

An additional important use of machine learning is in User and Entity Behavior Analytics (UEBA) programs. Such methodology is predicated on formulating a baseline of normative behavior for users or devices and subsequently detecting deviations from the set norms. The use of ML for behavior analysis shifts the burden of combating cyber threats from reactive to proactive for organizations since these systems can detect problems before considerable damage is inflicted [5]. Typical actions are clustered by K-means or DBSCAN algorithms, while outliers such as abnormal logon hours or access to sensitive information, may suggest insider threat or account takeover. Since recurrent neural networks (RNN) or Long Short Term Memory networks

(LSTM) can store and recall previous operations, they are commonly used to interpret time-ordered streams of actions and that is successfully applicable to the above issues.

For attacks such as DDoS (Distributed Denial of Service), machine learning utilizes time series analysis to forecast future events using historical patterns. For instance, ARIMA (AutoRegressive Integrated Moving Average) or LSTM models forecast the anticipated traffic metrics like 'requests per second', that correlate with periods of DDoS attacks. Such advanced notice enables proactive defense action, for example, re-routing traffic or bringing more servers online. Predicting attacks with ML is crucial given the cybercriminal's constant evolution and improvement of automated attack tools where conventional static defense mechanisms become nearly obsolete [4].

Machine learning can demonstrate self-sufficient learning within the automation of a threat response system. Models can be configured to strategically and automatically obstruct malicious traffic impressions, quarantine afflicted machines, or even create chronic rule modifications for firewalls in real time. For instance, reinforcement learning-based systems can improve response strategies by determining the optimal balance between blocking and minimizing false positive responses. Applying ML to cybersecurity systems improves performance while enabling experts to concentrate more on strategic work and less on operational tasks [6]. In the long run, the incorporation of ML into cybersecurity is more than a modern-day technology; it is an evolution of how systems, data, and information will be protected in the digital environment.

TECHNICAL ASPECTS OF USING ML IN CYBERSECURITY

In cyber security, machine learning is increasingly becoming an important asset to improve capabilities like automating threat identification, data mining, and real-time adaptive attacks. Integration of ML in this industry requires an array of algorithms, tools, and libraries organized in a system capable of handling complex multidimensional data including the network traffic, system logs, user activities, and even the malware's binary codes. As it has been said, in the context of machine learning we have three broad classifications of algorithms and methods: supervised learning, unsupervised learning, and reinforcement learning. Each of these has distinguishing technical features that make them appropriate for different uses and problems within cyber security.

Algorithms for supervised learning are employed in the classification and regression of problems with labeled training data. One of the domains for the implementation of these algorithms in cybersecurity is labeling network packets as either safe or malicious. Some effective approaches for these tasks are logistic regression and decision trees, along with support vector machines (SVM) which perform well on small well-annotated datasets [8]. Also, these algorithms extend to the classification of malware, making use of features like API calls, or behavioral patterns [9]. Let's review a few essential stages of implementing SVM in Scikit-Learn for traffic classification below:

- *Data Preparation*: this stage consists of preparing the dataset to make sure that it is ready for training;
- *Model Creation and Training*: in this step, already prepared data is used to design a "classifier model" and train it;
- *Dataset Requirements*: an SVM requires a dataset with a variety of traffic attributes and their corresponding class labels for the model to be accurately trained.

The data can be described in a tabular form as below, with each row referring to a network event, event attributes, and the last column value set as the class label:

ISSN 2663 - 4023

Source IP	Destination IP	Source Port	Destination Port	Protocol	Data	Time	Packets	Label
192.168.1.1	205.42.5.1	8080	443	TCP	204	10s	24	N
192.155.2.2	11.12.34.5	156	3000	TCP	11	50s	32	M
192.168.2.1	201.28.2.1	50000	60555	UDP	256	1s	225	M
192.55.1.1	38.2.55.2	344	252	UDP	234	2s	62	N

N — Normal
M — Malicious

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report, confusion_matrix

network_data = {
    'source_ip': [19216811, 19215522, 19216821, 1925511, 19265211, 17224566,
5636231],
    'destination_ip': [2054251, 1112345, 2012821, 382552, 23454542, 124111,
211263464],
    'source_port': [8080, 156, 50000, 344, 8000, 3245, 60000],
    'destination_port': [443, 3000, 60555, 252, 4433, 3001, 50001],
    'protocol_type': [1, 1, 0, 0, 1, 1, 0],  # 1 for TCP, 0 for UDP
    'data': [2048, 11, 256, 234, 512, 235, 73],
    'time': [10, 50, 1, 2, 6, 66, 8],
    'packets': [24, 32, 225, 62, 78, 2, 7],
    'label': ['n', 'n', 'm', 'm', 'n', 'm', 'm']
}

# Data Preparation
data_frame = pd.DataFrame(network_data)
features = data_frame.drop('label', axis=1)
labels = data_frame['label']

# Dataset Splitting (30%-70%)
F_train, F_test, L_train, L_test = train_test_split(features, labels,
test_size=0.3, random_state=12)

# Normalization (ensure that features are on a similar scale, optional)
scaler = StandardScaler()
F_train = scaler.fit_transform(F_train)
F_test = scaler.transform(F_test)

# SVM Model Creation and Training
model = SVC(kernel='linear')
model.fit(F_train, L_train)

# Predictions: the model makes predictions on the test set
predictions = model.predict(F_test)

# Model Evaluation
accuracy = accuracy_score(L_test, predictions)
print(f'Accuracy score: {accuracy * 100:.2f}%')
```

ISSN 2663 - 4023

To achieve a reasonable output, it's required to train the model with a large scale of data (labeled network traffic). Additionally, we can try to optimize the model by experimenting with different parameters and types of kernels like 'rbf', 'poly', and 'sigmoid'.

In situations where there is no labeled data available to discover unknown dangers or detect anomalies — an unsupervised learning strategy is very useful. Certain common algorithms are clustering methods like k-means and the Isolation Forest technique. Isolation Forest modality within Scikit-Learn is technically powerful when it applies to anomaly detection of multidimensional data, which is quite common with abnormal network traffic [7]. This algorithm employs the ensemble strategy and operates under the premise that the anomalies are infrequent events that can be isolated from most of the data in the feature space without much effort [10]. An Isolation Forest algorithm builds a collection of binary trees called Isolation Trees, which segment the data space by random partitioning. The essence is as follows: outliers are separated in a shorter time than normal observations because they are usually located at a greater distance from the center of the data. This algorithm is advantageous due to its great performance and speed, linear complexity □(□ · □□□(□)), and independence of any parameter that affects the data distribution which increases flexibility, and speed in separating outliers, since they are closer to the tree root [10].

```
from sklearn.ensemble import IsolationForest
import pandas as pd
import matplotlib.pyplot as plt

sample_data = {
    'val1': [1, 6, 8, 10, 7, 6, 2, 7, -88, 3, 5, 102, 8, 7, 1, 9, 3, 5, 77, 8],
    'val2': [6, 8, 1, 55, 3, 3, 9, -81, 6, 8, 1, 2, 6, 9, 5, 3, 1, 9, 43, 56],
}

data = pd.DataFrame(sample_data)

iso_forest = IsolationForest(contamination=0.1, random_state=12)
iso_forest.fit(data)

predictions = iso_forest.predict(data)

data['anomaly'] = predictions

anomalies = data[data['anomaly'] == -1]
print(anomalies)

plt.scatter(data['val1'], data['val2'], c=data['anomaly'], cmap='coolwarm')
plt.xlabel('Val 1')
plt.ylabel('Val 2')
plt.title('Isolation Forest')
plt.show()
```

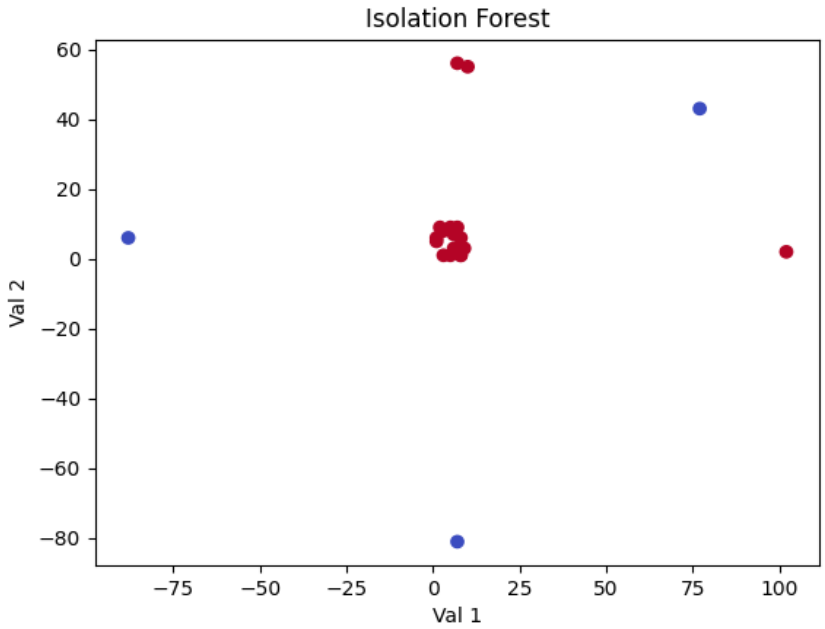

Fig. 1. Visualization of the Isolation Forest algorithm output

An Isolation Forest is employed in a lot of domains, like fraud detection on financial systems where the system needs to verify whether a certain credit card purchase is greatly outside the normal user activity scope; detection of anomalies like unusual surges in network traffic that can be an intensive DDoS attack; anomalous behavior in server logs suggestive of an intrusion; anomaly detection in serial data like spikes in industrial sensors readings indicative of a malfunctioning device and defect detection in manufacturing automates where data from the production system indicate nonconformities or violations of the process standards [11].

Deep Learning helps us examine complicated pieces of information like text (images, phishing emails) or sequences (logs). Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN) are highly effective for sequential data, for instance, Deep Learning is useful for encrypted traffic analysis, revealing encrypted information without decryption [9]. Evaluation of phishing URLs using Long Short-Term Memory (LSTM) is a perfect example of the use of recurrent neural networks in the domain of cybersecurity. Phishing URLs often contain patterns that are different from those found in legit URLs, such as domains that seem suspicious, odd characters, and unusually elongated paths. LSTM can toggle multiple parameters against themselves to train toward recognizing these patterns of known phishing URLs.

LSTM network implementation stages for analyzing phishing URLs:

Data collection and preparation: to train the model we should have a labeled dataset containing both phishing and legitimate URLs.

```
# Data
urls = [
    "www.facebook.com", "www.google.com", "www.stripe.com",
    "www.g00gle.com", "secure-login-paypal.com", "www.amaz0n.net"
]
# Labelling: 0 - legit, 1 - phishing
labels = [0, 0, 0, 1, 1, 1]
```

ISSN 2663 - 4023

Data preprocessing: URLs should be converted into numerical sequences to be passed to the LSTM model. Tokenization involves splitting the URL into individual tokens where each character is going to be treated as a separate token. The padding ensures that all URLs are of a similar length by adding zeros.

```
tokenizer = Tokenizer(char_level=True)
tokenizer.fit_on_texts(urls)
sequences = tokenizer.texts_to_sequences(urls)

max_len = max([len(seq) for seq in sequences])
padded_sequences = pad_sequences(sequences, maxlen=max_len, padding='post')
```

Data transformation and splitting: transforming data into numpy arrays, followed by dividing into training and test sets is essential for machine learning methodology. Initially, tokenized and normalized URL data is converted into numpy arrays which allows for working with numerical data quite easily. The arrays are then divided into a training set (normally 70%–80% of the data for model training) and a testing set (20%–30% of the data to determine the generalization and model accuracy) where the division is typically random but can be reproduced via fixed shuffling.

```
# Converting data to numpy arrays
X = np.array(padded_sequences)
y = np.array(labels)

# Splitting into training and test data
train_size = int(len(X) * 0.8)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
```

Create LTSM model: constructing the LSTM model with Keras for URL classification involves some important features: adding an embedding layer which helps transform each character of the URL into a dense vector at a constant size, enabling text data in number format representation; next, the application of an LSTM recurrent neural networks layer to processes the URLs on a character-by-character basis in a sequential manner and as a whole is referring to the timing and order of the characters (to be able to detect phishing URL patterns); last, but not least, add a Dense layer with a sigmoid function to a final layer which predicts the probability the given URL is phishing in a binary classification manner.

```
# Create the model
vocabulary_size = len(tokenizer.word_index) + 1

model = Sequential()
model.add(Embedding(input_dim=vocabulary_size, output_dim=16,
input_length=max_len))
model.add(Bidirectional(LSTM(64, return_sequences=False)))
model.add(Dropout(0.5))
model.add(Dense(1, activation='sigmoid'))

# Compile
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Get the structure of the model
model.summary()
```

ISSN 2663 - 4023

Model training: with the training phase, our learning process has to be set up wherein a model is compiled with the correct loss function, here being binary cross-entropy in case it is a binary classification. An optimizer such as Adam is defined, and appropriate metrics like accuracy are assigned. Then, the 'fit' method is executed, which requires defining training data, number of epochs (iterations of training), batch size, and optional validation information for performance tracking.

```
# Training
batch_size = 32
epochs = 10
history = model.fit(X_train, y_train, batch_size=batch_size, epochs=epochs,
validation_data=(X_test, y_test))
```

Model evaluation: this is done on a test set to see how effectively the model can operate on data it was not aware of during the training process. In the Keras library, the evaluate function is called and the test data (X_test) and test labels (y_test) are passed. The model returns the metrics defined during the compilation stage, enabling the evaluation of the model's generalization capabilities through the obtained accuracy and loss values.

```
# Evaluation
loss, accuracy = model.evaluate(X_test, y_test)
print(f"Test accuracy: {accuracy * 100:.2f}%")
```

Sample application of a trained LSTM model for analyzing phishing URLs:

```
def test(url, model, tokenizer, max_len):
    seq = tokenizer.texts_to_sequences([url])
    padded_seq = pad_sequences(seq, maxlen=max_len, padding='post')
    prediction = model.predict(padded_seq)[0][0]
    if prediction > 0.5:
        print(f"URL: {url} -- Phishing (probability: {prediction:.2f})")
    else:
        print(f"URL: {url} -- Legit (phishing probability: {prediction:.2f})")

# Testing on a new URL
new_url = "http://www.google.com/login"
test(new_url, model, tokenizer, max_len)
```

Implementation of LSTM networks is an effective approach to analyzing phishing URLs. It stems from the ability of LSTMs to process sequences of characters and identify long-term dependencies and patterns of the data, which allows for precise classification of URLs into legitimate or phishing URLs. The model displays high accuracy, sometimes as high as 99% (in certain studies), notably with large datasets like PhishTank, a crowdsourced free repository for collecting, validating, and sharing information about phishing websites. It also outperforms conventional machine learning techniques like Random Forests by featuring self-extraction, which is done without the need for manual engineering. However, such systems do not come without challenges since they require massive amounts of training data to meet the targeted outcome. The high degree of computational complexity, their notorious black box nature, and the frequent need to refresh the underlying model to capture emerging phishing trends also make it difficult to interpret their results [13].

CHALLENGES AND LIMITATIONS

Along with the integration of machine learning into cybersecurity comes several effectiveness challenges, one of which is data quality. In practice, data is often skewed, with normal network operations significantly outnumbering attack instances, which may be as low as 0.1%. If the ML models are trained on this data, they may suffer from overfitting because of the excessive normal data and as a result, we might get a poor detection of anomalies. Although there are solutions to this problem such as oversampling methods (e.g. SMOTE) or anomaly detection approaches. Synthetic Minority Oversampling Technique (SMOTE) is an oversampling technique used to fight against an imbalanced dataset by oversampling and creating new examples of the minority class through linear interpolation in the examples and corresponding □ nearest neighbors (which is usually set to □ = *5*). This technique enhances the model's generalization capabilities and helps to avoid simple replication. However, it can produce unreasonable data in scenarios with high-class overlap, and therefore, must be handled with caution in high-dimensional datasets [12]. In less complicated terms, SMOTE is a technique used to add more instances of data that relate to a rare class, like phishing URLs for example, which are highly outnumbered by normal class phishing URLs. Instead of just repopulating existing samples, it comes up with a new synthetic sample created through the mingling of samples. For example, if two phishing URLs are too similar, SMOTE creates an "average" of them so it can be used as an additional training sample for the model. This helps the model detect outliers more frequently, but those fictitious examples can be a bit unrealistic at times.

Adversarial attacks comprise yet another perilous challenge. This is where malicious users actively attempt to mislead an ML model by injecting noise into the data. These 'adversarial examples' have been shown to completely alter the decisions of the model with little to no changes of input data, creating problems especially for intrusion detection systems (IDS). There are protective measures such as Adversarial Training and Robust Optimization. Adversarial Training stands as one of the most favored approaches used to improve the resilience of ML models from attacks. The core concept is to enhance the training set's pre-processing by integrating adversarial examples, which are data points that closely resemble the original samples, but are deceptively altered in a manner that is bound to confuse the model, thereby resulting in misclassifications. The model learns to classify both normal and altered data correctly, and therefore, it can tolerate such manipulations later on. Another approach, Robust Optimization, defines models with resistance to a great deal of possible perturbations within the data. Rather than reacting to specific attacks as in adversarial training, this method aims to ensure that a model's decisions remain accurate within the worst-case perturbation scenarios that exist within a given radius, such as L2 or L∞ norms variance [6]. Still, developing robust models is quite a challenging task because of the constantly changing nature of hacking techniques.

Computational resources impose further constraints as one of the leading limitations. The deep learning approaches often implemented in the realm of cyber security to automatically scan large datasets tend to be highly complex and resource-dependent on energy, GPU, TPU (Tensor Processing Unit), and time for training. This is often a limitation for low-budget organizations to use modern ML systems. These problems are compounded in the sphere of cybersecurity with the question of model interpretability. Most machine learning algorithms, and especially neural networks, are operated as "black boxes", which do not provide explanations about the reasons that underlie their actions. For the human analyst understanding why some traffic has been classified as potentially harmful is crucial. Some tools such as SHAP

or LIME give us the opportunity to understand explanations generated by machine learning models, for example, to get an explanation of LSTM model decisions during phishing URL detection. SHAP functions like a detective that accurately divides “blame” of a decision among all parts of a URL (like words or patterns), showing each component's impact on the resulting decision. LIME is a quicker “detective” who only works with specific URLs simply by covering more complex ones with models. This comes with a cost in accuracy and a broader scope. If rapid explanations of particular URLs are the goal, use LIME. In contrast, SHAP is preferred when time is not an issue and greater analysis is required [6]. Both of these different modern methods of interpreting information searched for and present explanations and practical usability, which may still not exist.

CONCLUSION

Machine learning is fundamental to solving modern cybersecurity problems, especially regarding threat analysis, attack forecasting, and user activity monitoring. Unlike traditional cybersecurity methods, it does not rely on static systems. The theoretical analysis highlights the requirement for counteractive measures to multifaceted cyberattacks as well as the existence of problems like dataset imbalance, adversarial susceptibility, and the need for explainability, particularly regarding “black boxes” which deep learning models are.

The complexities involved with using algorithms to automate threat detection and response in cybersecurity demonstrate both the promise and risks of machine learning. Machine learning techniques, whether supervised or unsupervised, make it easier to work with intricate datasets, allowing users to find deviations and trends in network traffic, user activities, and other security risks. Nonetheless, the efficient implementation of ML in this area needs a special focus on data accuracy, available computation resources, and model interpretation. Imbalance dataset problems which can lead to overfitting require the implementation of methods such as SMOTE in order to improve the model training through synthetic example generation. Furthermore, these tools are essential for ensuring that the model withstands smart attacks from hostile forces through adversarial training and robust optimization techniques. In addition, automating trust with models requires understanding their workings which is why tools for interpretability such as SHAP and LIME are needed.

The future scope of research includes both theoretical and practical facets. An area that requires special attention is improving model explainability with SHAP and LIME, as well as improving model robustness by some novel approaches to adversarial training. On a more practical level, there is scope in the optimization of models for use in real-time on resource-limited devices, such as through TensorFlow Lite, and the use of Generative Adversarial Networks (GANs) for producing synthetic data for the evaluation of security systems.

REFERENCES

1. Singer, P. W., & Friedman, A. (2014). *Cybersecurity and cyberwar: What everyone needs to know*. Oxford University Press.
2. Mitchell, T. M. (1997). *Machine learning*. McGraw-Hill.
3. James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning: With applications in R*. Springer.
4. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.
5. Jacobs, J., & Rudis, B. (2014). *Data-driven security: Analysis, visualization and dashboards*. Wiley.

6. Joseph, A. D., Nelson, B., Rubinstein, B. I. P., & Tygar, J. D. (2019). *Adversarial machine learning*. Morgan & Claypool Publishers.
7. Géron, A. (2019). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems* (2nd ed.). O'Reilly Media.
8. Gulli, A., Kapoor, A., & Pal, S. (2019). *Deep learning with TensorFlow 2 and Keras: Regression, ConvNets, GANs, RNNs, NLP, and more with TensorFlow 2 and the Keras API* (2nd ed.). Packt Publishing.
9. Chio, C., & Wagner, D. (2018). *Machine learning and security: Protecting systems with data and algorithms*. O'Reilly Media.
10. Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2008). Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, 413–422. IEEE. https://doi.org/10.1109/ICDM.2008.17
11. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12(10)*, 2825–2830.
12. Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, *16*, 321–357.
13. Sahoo, D., Liu, C., & Hoi, S. C. H. (2019). Malicious URL detection using machine learning: A survey. *ACM Computing Surveys*, *52(3)*, 1–37.

КІБЕРБЕЗПЕКА: освіта, наука, техніка
CYBERSECURITY: EDUCATION, SCIENCE, TECHNIQUE
№ 4 (28), 2025
ISSN 2663 - 4023

Мелько Тарас
PhD, кафедра комп'ютерних наук та програмної інженерії
Приватний вищий навчальний заклад «Європейський університет», Київ, Україна
ORCID ID: 0009-0005-7295-5863
tarasxmelko@gmail.com

Коцун Володимир
кандидат технічних наук, доцент кафедри комп'ютерних наук та програмної інженерії
Приватний вищий навчальний заклад «Європейський університет», Київ, Україна
ORCID ID: 0000-0003-2363-8157
volodumur.kotsun@e-u.edu.ua

ТЕОРЕТИЧНІ ТА ТЕХНІЧНІ АСПЕКТИ ВИКОРИСТАННЯ МАШИННОГО НАВЧАННЯ В КІБЕРБЕЗПЕЦІ

Анотація. У статті досліджуються технічні та теоретичні аспекти машинного навчання (ML) у вирішенні проблем, пов'язаних зі зростаючою складністю загроз кібербезпеки в цифрову епоху, оскільки постійне зростання кіберзлочинності спонукає користувачів використовувати новітні підходи для підвищення планки кібербезпеки. Дослідження розглядає впровадження технології машинного навчання (ML) як наріжний камінь практично будь-якої сучасної проблеми кібербезпеки, зокрема процесів і методів, пов'язаних з аналізом проблем, виявленням, прогнозуванням атак і навіть поведінковим профілюванням. Розглянуто, як ML забезпечує кращу реакцію порівняно з традиційними методами, такими як виявлення на основі сигнатур, пояснюючи, як стає можливим аналіз великих обсягів даних у реальному часі. Огляд важливих особливостей контрольованого та неконтрольованого навчання надається в контексті виявлення аномалій та розпізнавання зловмисної активності з акцентом на алгоритми Machine of Support Vector Machine та Isolation Forests, а також детальний огляд моделі LSTM для аналізу еволюції фішингових URL-адрес. Крім того, ці алгоритми були висвітлені з точки зору технічної реалізації: контрольоване навчання за допомогою машин опорних векторів з використанням Scikit-Learn для класифікації мережевого трафіку, навченого на таких характеристиках, як IP-адреси і порти, неконтрольоване навчання за допомогою ізоляційних лісів для виявлення аномалій у багатовимірних даних, а також глибоке навчання за допомогою мереж з довгою короткочасною пам'яттю (LSTM) для аналізу фішингових URL-адрес. У цій статті досліджуються суттєві труднощі в реалізації алгоритмів ML, такі як дисбаланс класів, ворожі атаки та недостатня прозорість моделі. Такі методи, як SMOTE (Synthetic Minority Over-sampling Technique), пропонуються для розробки навчальних наборів даних, тоді як для захисту від зловмисного використання моделей пропонуються методи змагального навчання та робастної оптимізації. Крім того, підкреслено роль методів пояснюваності, таких як SHAP і LIME, для побудови довіри і прийняття автоматизованих систем ML в кібербезпеці. Визначено можливості для досліджень і запропоновано провести подальше тестування щодо покращення надійності моделей та показників продуктивності в обмежених умовах.

Ключові слова: кібербезпека; кібератаки; кіберзахист; машинне навчання; глибоке навчання; машинне навчання в кібербезпеці.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Singer, P. W., & Friedman, A. (2014). *Cybersecurity and cyberwar: What everyone needs to know*. Oxford University Press.
2. Mitchell, T. M. (1997). *Machine learning*. McGraw-Hill.
3. James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning: With applications in R*. Springer.
4. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.
5. Jacobs, J., & Rudis, B. (2014). *Data-driven security: Analysis, visualization and dashboards*. Wiley.
6. Joseph, A. D., Nelson, B., Rubinstein, B. I. P., & Tygar, J. D. (2019). *Adversarial machine learning*. Morgan & Claypool Publishers.

7. Géron, A. (2019). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems* (2nd ed.). O'Reilly Media.
8. Gulli, A., Kapoor, A., & Pal, S. (2019). *Deep learning with TensorFlow 2 and Keras: Regression, ConvNets, GANs, RNNs, NLP, and more with TensorFlow 2 and the Keras API* (2nd ed.). Packt Publishing.
9. Chio, C., & Wagner, D. (2018). *Machine learning and security: Protecting systems with data and algorithms*. O'Reilly Media.
10. Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2008). Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, 413–422. IEEE. https://doi.org/10.1109/ICDM.2008.17
11. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research, 12(10)*, 2825–2830.
12. Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research, 16*, 321–357.
13. Sahoo, D., Liu, C., & Hoi, S. C. H. (2019). Malicious URL detection using machine learning: A survey. *ACM Computing Surveys, 52(3)*, 1–37.