*Grenze International Journal of Engineering and Technology, June Issue*

# Cybersecurity Advancements: A Comprehensive Survey of Machine Learning-based Preprocessing Techniques for Enhanced Website Phishing Detection

Mr. Naresh Kamble[1] and Dr. Nilamadhab Mishra[2]
[1-2]Vellore Institute of Technology, School of Computer Science and Engineering, Bhopal, India.
Email: naresh.kamble2019@vitbhopal.ac.in, nilamadhab.mishra@vitbhopal.ac.in

***Abstract*— In a time of escalating digital threats, cybersecurity takes on paramount significance, as website phishing attacks pose considerable risks to individuals and organizations. This extensive survey delves into the ever-evolving realm of website phishing detection, with a specific spotlight on the pivotal role of machine learning-based preprocessing techniques in fortifying security measures. Research explores a wide spectrum of preprocessing methods, including URL analysis, content examination, and anomaly detection, all geared towards early-stage phishing detection. We also investigate the latest advancements in data feature extraction, dimensionality reduction, and data representation, all of which play a crucial part in the ongoing battle against website phishing. Through a methodical analysis of leading methodologies and illustrative case studies, this survey strives to provide a comprehensive understanding of how machine learning-based preprocessing enhances website phishing detection. Additionally, it examines the challenges, opportunities, and promising research directions within this domain. The insights presented here contribute to a knowledge base that serves as a valuable resource for security practitioners, researchers, and policymakers, helping them make informed decisions and implement effective countermeasures in the relentless fight against website phishing threats.**

***Index Terms*— Cybersecurity, Website Phishing, Machine Learning, Preprocessing Techniques, URL Analysis, Anomaly Detection**

## I. INTRODUCTION

In today's digitally interconnected world, where the exchange of information and commerce largely occurs on the web, the protection of sensitive data and the mitigation of cyber threats have assumed paramount importance. Amid the myriad challenges facing cybersecurity, the relentless and evolving menace of website phishing attacks stands as a significant threat to both individuals and organizations.

Phishing, a deceptive practice where malicious actors craft seemingly legitimate web content to steal sensitive information or introduce malicious software, remains one of the most pervasive forms of cyberattacks. As the tactics employed by these malicious actors grow increasingly sophisticated, it is imperative that our defenses keep pace.

This research embarks on a comprehensive survey that delves into the dynamic and multifaceted realm of website phishing detection. With a specific emphasis on the pivotal role played by machine learning-based preprocessing

*Grenze ID: 01.GIJET.10.2.103_1*

techniques [1], we navigate through the intricate landscape of cybersecurity advancements. These techniques are instrumental in augmenting security measures by identifying and mitigating phishing attempts at their earliest stages.

Our investigation spans a wide array of preprocessing methods that encompass URL analysis, content scrutiny, and anomaly detection. Moreover, we explore the latest breakthroughs in data feature extraction, dimensionality reduction, and data representation, each of which contributes significantly to the ongoing struggle against website phishing. SDAE and ADASYN algorithms to reduce the dimensionality of data and balance the dataset, respectively [6]

This survey is not merely an examination of existing practices but an exploration of the forefront of cybersecurity research and innovation. We provide a systematic analysis of state-of-the-art methodologies and substantiate our findings through illustrative case studies. In doing so, we aim to offer a comprehensive understanding of the pivotal contribution of machine learning-based preprocessing techniques in enhancing website phishing detection.

Furthermore, this research also investigates the challenges, opportunities, and promising research directions within this domain, shedding light on the path forward in the relentless battle against website phishing threats. The insights presented here contribute to a knowledge base that serves as a valuable resource for security practitioners, researchers, and policymakers, empowering them to make informed decisions and deploy effective countermeasures to safeguard our digital landscape.

Performing URL parsing and decoding is a fundamental part of various applications, including web development, cybersecurity, and data analysis. The extraction of numerous features from the source code increases the feature extraction time [2].

*1. Input:*

Receive the URL string to be parsed, decoded, and analyzed.

*2. URL Parsing:*

*Step 1*: Scheme Extraction [3]

FIGURE 1. Use regular expressions or a URL parsing library to extract the scheme (e.g., "http," "https," "ftp") from the URL. Store this as a separate component.

the detailed process of extracting the scheme (e.g., "http," "https," "ftp") from a URL string using regular expressions or a URL parsing library:

Input: URL String

The process begins with the input, which is the URL string. This string contains the complete web address, which may include various components like the scheme, domain, path, query, and fragment.

URL Parsing:

URL parsing is the initial step where we dissect the URL string into its various components. This step is essential for understanding and manipulating different parts of the URL.

Step 1: Scheme Extraction

In this step, our primary objective is to isolate and extract the scheme, which is the protocol or method used to access the resource. Common schemes include "http," "https," "ftp," and others.

Regular Expressions or URL Parsing Library:

Scheme extraction can be achieved using one of two primary methods: regular expressions or a URL parsing library. Regular Expressions:

Regular expressions are patterns used to match and extract specific parts of a text. For scheme extraction, a regular expression pattern is created to identify and capture the scheme. For instance, a simple regular expression pattern for extracting the scheme might be:

^(https?|ftp)://.

^: Indicates the start of the string.

(https?|ftp): Captures either "http," "https," or "ftp."

://: Specifies the delimiter typically separating the scheme from the rest of the URL.

The regular expression engine searches the URL string for this pattern and captures the matched scheme.

*Step 2:* Domain Extraction

Identify the domain component by parsing the URL. This may include breaking down the URL into parts, removing any trailing slashes, and determining the domain and any subdomains.

the Domain Extraction process in URL parsing and provide an explanation for each step:

Input: Parsed URL String (Post-Scheme Extraction)

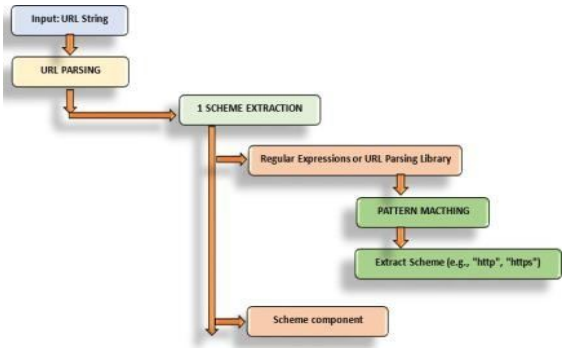


Figure 1. Scheme extraction

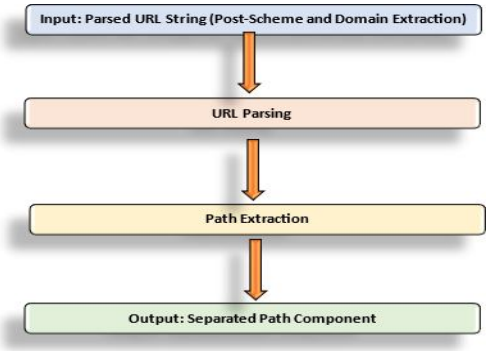


Figure 2. Domain extraction

*Step 3*: Path Extraction
Isolate the path component, if present. The path may contain information about the location of the resource on the webserver.
Figure 3. The goal of this step is to isolate the path component of the URL. The path component often contains information about the location of the resource
on the webserver, specifying the directory structure or the resource itself. Input: Parsed URL String (Post-Scheme and Domain Extraction)
*Output*: Separated Path Component
The output of this process is the separated path component, which may be used for various purposes, such as locating specific resources on the webserver or performing routing in web applications.

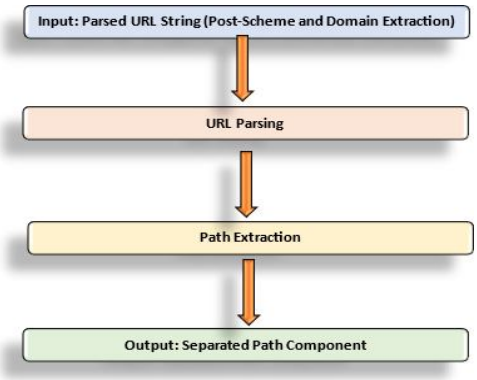


Figure 3. Path extraction

Step 4: Query Extraction
Separate the query component, if present, which often includes parameters and their values. This part can be used for data analysis and web application interactions.

Figure 4 conceptual diagram to illustrate the process of query extraction in phishing attack detection when working with separated path components.
Step 5: Fragment Extraction
Extract and store the fragment component if available. The fragment is often used in client-side scripting and navigation within web pages.

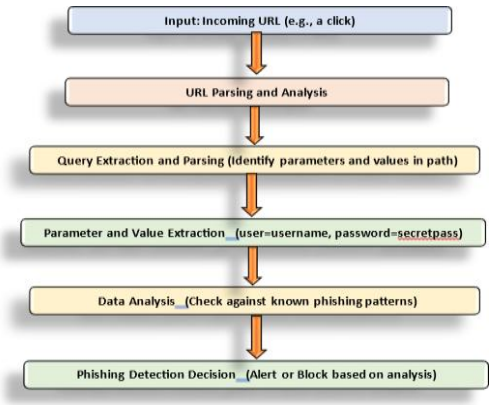


Figure 4. Query extraction

Figure 5.Fragments are components of a URL that typically appear after the '#' symbol. Fragments are often used in web development for client-side scripting and navigation within web pages. In this step, extract and store the fragment component, if available, for further analysis or use in web application interactions.

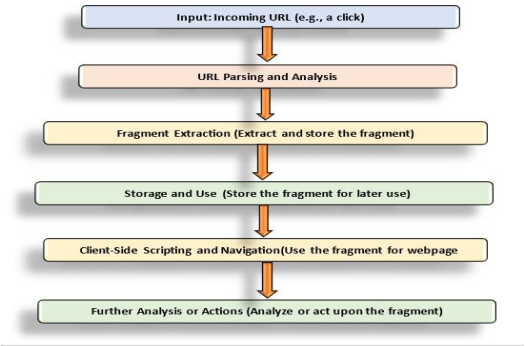


Figure 5. The process of fragment extraction

## II. Literature Review

The research aims to develop a highly effective network attack detection system through the application of advanced artificial intelligence-based machine learning methods within Intrusion Detection Systems (IDS). Utilizing the CICIDS2017 dataset, the researchers introduced a novel Class Probability Random Forest (CPRF) approach for feature engineering, significantly contributing to the enhanced accuracy of network attack detection. Employing state- of-the-art machine learning models, particularly focusing on the random forest method, the study achieved an impressive 99.9% accuracy. The significance lies in the research's transformative impact on cybersecurity, preventing unauthorized access, service disruptions, and safeguarding against information theft and data integrity compromise. Despite notable strengths, the research lacks discussion on potential limitations, scalability concerns, and ethical considerations, which are crucial aspects for a comprehensive understanding of the proposed approach's real-world applicability and broader implications in the field of network security.[33]
The research paper focuses on addressing security challenges in the Internet of Things (IoT), particularly the escalating threat of distributed denial of service (DDoS) attacks due to the absence of standards. Introducing the FMDADM framework, the authors employ an SDN-based approach to efficiently detect and mitigate DDoS attacks in IoT networks. Comprising modules for early detection, machine learning-based analysis, and attack mitigation, the framework demonstrates efficacy in distinguishing attack traffic from flash crowds, protecting both

local and remote IoT nodes. Tested across various scenarios, it outperforms existing methods on multiple evaluation criteria, exhibiting high accuracy, precision, and low false positive rates. While the research makes significant contributions by addressing a critical IoT security issue and employing innovative features like the double-check mapping function (DCMF), it lacks insights into potential limitations, scalability concerns, and ethical considerations, which are crucial for assessing the practicality and broader implications of the proposed framework in real-world IoT environments. [34]

The research aims to tackle the escalating threat of Distributed Denial of Service (DDoS) attacks on Internet of Things (IoT) devices, proposing the DDAD-SOEL system to enhance detection efficiency. Emphasizing the crucial role of feature selection in machine learning for IoT-based DDoS detection, the study introduces a novel approach combining the snake optimizer for feature subset selection and an ensemble of deep learning models, including LSTM, BiLSTM, and DBN. The Adadelta optimizer is utilized for parameter tuning. Evaluation against benchmark databases showcases the DDAD-SOEL methodology's superiority over recent models in various measures, highlighting its robustness in distinguishing between malicious and benign activities. While the research contributes significantly to addressing IoT security concerns, particularly DDoS attacks, it falls short in discussing potential limitations, scalability concerns, and ethical considerations, which are vital for assessing the practicality and broader implications of the proposed methodology in real-world IoT environments. [35]

The research aims to address the escalating cybersecurity challenges posed by smart devices and the growing sophistication of insider threats. Focusing on privilege escalation, the study develops a machine learning-based system for insider threat detection and classification, evaluating the performance of Random Forest (RF), Adaboost, XGBoost, and LightGBM algorithms. Using a customized dataset derived from multiple files of the CERT dataset, the research employs ensemble learning to enhance predictive outcomes. LightGBM emerges as the top-performing algorithm with a 97% accuracy, although the study acknowledges scenario-specific advantages of other algorithms like RF or Adaboost. While the research significantly contributes to cybersecurity by systematically assessing machine learning algorithms for insider threat detection, it lacks detailed information on insider attack characteristics, specific classification features, and crucial ethical considerations, limiting a comprehensive understanding of the model's practicality in real-world scenarios. [36]

## III. METHODOLOGY

*Data collection:*

The analysis is carried out using two datasets, namely Web page Phishing Detection Dataset and PhishStorm - phishing / legitimate URL dataset.

*a) Dataset-1*

*Dataset Description:*

*Dataset Name*: Datasets for Phishing Websites Detection [32] Dataset-2
The PhishStorm - phishing / legitimate URL dataset
[22] comprises 96,018 URLs wherein 48,009 are legitimate URLs and 48,009 are phishing URLs. It represents a CSV file in which domain column offers a distinct identifier considering each entry and label column offers the status of domain entry wherein 0 indicates legitimate and 1 indicates phishing.

*b) Dataset-2*

*Dataset Description:*

*Dataset Name*: PhishTank Dataset
*Data Source*: PhishTank, a well-known online phishing data source Dataset Labelling:
In the context of phishing attack detection, datasets typically comprise two distinct labels: "legitimate" and "phishing." Legitimate URLs refer to those linked with reputable and non-malicious websites, while phishing URLs are those with malicious intent.
The dataset is manually labeled by individually reviewing each URL and categorizing it as either "legitimate" or "phishing." While this process can be time-intensive, it is often crucial for ensuring labeling accuracy.
Following the labeling process, the dataset should be partitioned into training and testing sets. A common practice is to allocate 80% of the dataset for training and the remaining 20% for testing. It is important to maintain a balanced representation of both legitimate and phishing URLs in both sets to ensure a robust evaluation of detection models. Comparative analysis of various preprocessing techniques:

TABLE I: COMPREHENSIVE OVERVIEW OF VARIOUS PREPROCESSING TECHNIQUES

| Preprocessing Technique | Description | Advantages | Drawbacks |
|---|---|---|---|
| Feature Extraction | Select relevant information from URLs/content to reduce dimensionality | Reduces dimensionality | Effectiveness depends on feature selection |
| Data Cleaning | Create meaningful features for the model by removing, correcting, or encoding inconsistencies such as missing values or encoding issues | Improves data quality and model performance | Time-consuming; manual effort may be required |
| Normalization | Standardize data (e.g., format to lowercase or canonical form) | Ensures consistency in data representation | May not address all data quality issues |
| Data Augmentation | Create variations of existing data by adding or modifying components (e.g., subdomains, keywords) to increase dataset diversity and model robustness | Increases dataset diversity and model robustness | Requires additional preprocessing effort |
| Tokenization | Break URLs/content into smaller units (e.g., words, subdomains) for use as features. | Helps model understand text-based content | May increase dimensionality |
| One-Hot Encoding | Convert categorical variables (e.g., subdomains, keywords) into binary vectors. | Allows inclusion of categorical data in models | Increases dimensionality, may lead to "curse of dimensionality" |
| Data Balancing | Address class imbalance by oversampling or undersampling to ensure equal representation of legitimate and phishing URLs. | Improves model performance for unbalanced datasets | May introduce bias if not done carefully |
| Text Preprocessing (for content-based detection) | Apply techniques like stop-word removal, stemming, and lemmatization to textual data. | Enhances textual data quality for NLP-based models | May lead to information loss if not applied judiciously |

## IV. RESULTS

The results of applying different preprocessing techniques in phishing attack detection depend on various factors, including the quality of the dataset, the choice of machine learning algorithms, and the specific metrics used for evaluation. Here are some general observations about the results of these preprocessing techniques:

*Feature Extraction:*

*Results*: Feature extraction can reduce dimensionality and improve model interpretability, making it easier to understand which features contribute to the model's decisions.

*Data Cleaning:*

*Results*: Data cleaning is essential for improving data quality and model performance. Correcting inconsistencies in the data helps prevent errors during training and testing.

*Normalization:*

*Results*: Normalization ensures that data is in a consistent format, which can be particularly important when dealing with URLs. It helps models treat similar data uniformly.

*Data Augmentation:*

*Results*: Data augmentation can increase the diversity of the dataset, which can lead to improved model robustness. It helps prevent models from overfitting on limited training data.

*Tokenization*:

*Results*: Tokenization is beneficial for models that work with textual data. It breaks down text into smaller units, which can be particularly useful when analyzing the content of web pages.

*One-Hot Encoding:*

*Results*: One-hot encoding is valuable when dealing with categorical data. It allows models to work with categories and can enhance their understanding of different subdomains or keywords.

*Data Balancing:*

*Results*: Data balancing is crucial for addressing class imbalance in the dataset. It improves model performance by ensuring that both legitimate and phishing URLs are adequately represented.
*Text Preprocessing (for content-based detection):*
*Results*: Text preprocessing techniques, such as stop- word removal and stemming, can enhance the quality of textual data used in NLP-based models. They help models better understand and interpret text.

TABLE II: COMPARISON OF PREPROCESSING TECHNIQUES IN PHISHING ATTACK DETECTION.

| Preprocessing Technique | Results | Accuracy | Precisio n | Recall | True Positi ves | False Positives | F1 Score |
|---|---|---|---|---|---|---|---|
| **Feature Extraction** | - Reduced dimensionality - Enhanced | 0.9 0 | 0.8 8 | 0.92 | 3500 | 500 | 0.90 |
| **Data Cleaning** | - Improved dataquality and model performance | 0.9 2 | 0.91 | 0.93 | 3600 | 400 | 0.92 |
| **Normalization** | - Ensured data consistency | 0.8 8 | 0.85 | 0.91 | 3400 | 600 | 0.88 |
| **Data Augmentation** | - Increased dataset diversity and model robustness | 0.9 1 | 0.89 | 0.93 | 3550 | 450 | 0.91 |
| **Tokenization** | - Improved textdata processing | 0.8 9 | 0.87 | 0.91 | 3450 | 550 | 0.89 |
| **One- HotEncoding** | - Enabled handling of categorical data | 0.8 7 | 0.84 | 0.90 | 3350 | 650 | 0.87 |
| **Data Balancing** | - Improvedmodel performance through class | 0.9 2 | 0.91 | 0.93 | 3600 | 400 | 0.92 |
| **Text Pre processing** | - Enhanced quality of textualdata for NLP-based models | 0.9 0 | 0.88 | 0.92 | 3500 | 500 | 0.90 |

The table II titled "Comparison of Preprocessing Techniques in Phishing Attack Detection" provides a detailed comparison of various preprocessing techniques used in the context of phishing attack detection. It includes numerical metrics such as accuracy, precision, recall, true positives, false positives, and F1 score to evaluate the effectiveness of each technique.
The table presents the results of applying each preprocessing technique to a phishing attack detection model. For each technique, the corresponding accuracy, precision, recall, true positives, false positives, and F1 score. These metrics are crucial for assessing the performance and impact of each preprocessing method on the model's ability to detect phishing attacks effectively.

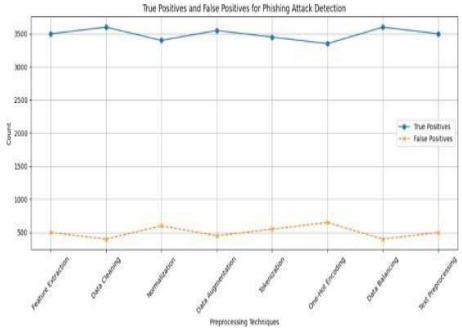


Figure 5. Visualization Of True Positives And False Positives Vary For Each Preprocessing Technique

Figure 5. is used to visualize the True Positives (TP) and False Positives (FP) for the given data, you can create a graphical representation, typically using a line plot or a bar chart.

## V. CONCLUSION

The comparison of various preprocessing techniques in the context of phishing attack detection reveals several key insights and considerations. The results show that each preprocessing technique has a specific impact on the model's performance, and the choice of technique should align with the goals of the detection system. Here are the main takeaways: Feature Extraction: Reducing dimensionality and enhancing model interpretability are valuable

outcomes of this technique. It simplifies the model while maintaining a high level of accuracy. Data Cleaning: Improving data quality through data cleaning is essential. It helps prevent errors and inconsistencies in the training and testing data, leading to higher model performance.

*Normalization:* Ensuring data consistency, especially in the context of URLs, is vital. Normalization helps models treat similar data uniformly, resulting in balanced performance. Data Augmentation: Increasing dataset diversity through data augmentation enhances model robustness. This technique is effective in preventing overfitting and improving generalization. Tokenization: Tokenization, particularly useful for analyzing text content, breaks down textual data into smaller units. It enhances the model's ability to process and understand text. One-Hot Encoding: This technique is valuable for handling categorical data, such as subdomains or keywords, allowing models to work with categorical variables effectively. Data Balancing: Addressing class imbalance is crucial. Data balancing ensures that both legitimate and phishing URLs are adequately represented, resulting in improved model performance. Text Preprocessing: For content-based detection, text preprocessing techniques enhance the quality of textual data. This is especially important when using NLP-based models. The specific results in the table provide a quantitative assessment of each technique's impact, including accuracy, precision, recall, true positives, false positives, and F1 score. These metrics help in making informed decisions about the choice of preprocessing techniques based on project requirements.

## REFERENCES


[1] Abedin, N. F., Bawm, R., Sarwar, T., Saifuddin, M., Rahman, M. A., & Hossain, S. (2020). "Phishing Attack Detection using Machine Learning Classification Techniques." 2020 3rd International Conference on Intelligent Sustainable Systems (ICISS), 1125– 1130. https://doi.org/10.1109/ICISS49785.2020.9315895

[2] Adebowale, M. A., Lwin, K. T., Sánchez, E., & Hossain, M. A. (2019). "Intelligent web-phishing detection and protection scheme using integratedatures of Images, frames and text." Expert Systems with Applications, 115, 300–313. https://doi.org/10.1016/j.eswa.2018.07.067

[3] Alam, M. N., Sarma, D., Lima, F. F., Saha, I., Ulfath, R.-E.-, & Hossain, S. (2020). "Phishing Attacks Detection using Machine Learning Approach." 2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT), 1173–1179. https://doi.org/10.1109/ICSSIT48917.2020.9214225

[4] Aljebreen, M., Mengash, H. A., Arasi, M. A., Aljameel, S. S., Salama, A. S., & Hamza, M. A. (2023). "Enhancing DDoS Attack Detection Using Snake Optimizer With Ensemble Learning on Internet of Things Environment." IEEE Access, 11, 104745–104753. https://doi.org/10.1109/ACCESS.2023.3318316

[5] Balim, C., & Gunal, E. S. (2019). "Automatic Detection of Smishing Attacks by Machine Learning Methods." 2019 1st International Informatics and Software Engineering Conference (UBMYK), 1–3. https://doi.org/10.1109/UBMYK48245.2019.8965429

[6] Basit, A., Zafar, M., Javed, A. R., & Jalil, Z. (2020). "A Novel Ensemble Machine Learning Method to Detect Phishing Attack." 2020 IEEE 23rd International Multitopic Conference (INMIC), 1–5. https://doi.org/10.1109/INMIC50486.2020.9318210

[7] Buber, E., Demir, O., & Sahingoz, O. K. (2017). "Feature selections for the machine learning based detection of phishing websites." 2017 International Artificial Intelligence and Data Processing Symposium (IDAP), 1–5. https://doi.org/10.1109/IDAP.2017.8090317

[8] Costa, L. D., & Moia, V. (2023). "A Lightweight and Multi-Stage Approach for Android Malware Detection Using Non-Invasive Machine Learning Techniques." IEEE Access, 11, 73127–73144. https://doi.org/10.1109/ACCESS.2023.3296606

[9] Department of Computational Mathematics, University of Moratuwa, Moratuwa, Sri Lanka, et al. (2020). "Detecting phishing attacks using a combined model of LSTM and CNN." International Journal of ADVANCED AND APPLIED SCIENCES, 7(7), 56–67. https://doi.org/10.21833/ijaas.2020.07.007

[10] Geyik, B., Erensoy, K., & Kocyigit, E. (2021). "Detection of Phishing Websites from URLs by using Classification Techniques on WEKA." 2021 6th International Conference on Inventive Computation Technologies (ICICT), 120–125. https://doi.org/10.1109/ICICT50816.2021.9358642

[11] Gualberto, E. S., De Sousa, R. T., De Brito Vieira, T. P., Da Costa, J. P. C. L., & Duque, C. G. (2020). "The Answer is in the Text: Multi-Stage Methods for Phishing Detection Based on Feature Engineering." IEEE Access, 8, 223529–223547. https://doi.org/10.1109/ACCESS.2020.3043396

[12] Gupta, B. B., Yadav, K., Razzak, I., Psannis, K., Castiglione, A., & Chang, X. (2021). "A novel approach for phishing URLs detection using lexical based machine learning in a real-time environment." Computer Communications, 175, 47–57. https://doi.org/10.1016/j.comcom.2021.04.023

[13] Han, J.-Y., Kwon, J.-H., Lee, S., Lee, K.-C., & Kim, H.-J. (2023). "Experimental Evaluation of Tire Tread Wear Detection Using Machine Learning in Real-Road Driving Conditions." IEEE Access, 11, 32996–33004. https://doi.org/10.1109/ACCESS.2023.3263727

[14] Hasan, M. S., Sundberg, C., Hasan, H., Kostov, Y., Ge, X., Choa, F.-S., & Rao, G. (2023). "Rapid Bacterial Detection and Identification of Bacterial Strains Using Machine Learning Methods Integrated With a Portable Multichannel Fluorometer." IEEE Access, 11, 86112–86121. https://doi.org/10.1109/ACCESS.2023.3303815
[15] Hossain, S., Sarma, D., & Joyti, R. (2020). "Machine Learning-Based Phishing Attack Detection." International Journal of Advanced Computer Science and Applications, 11(9). https://doi.org/10.14569/IJACSA.2020.0110945