User-Centric Web Application for Phishing URL Detection by Machine Learning Model

Palida Yingwatchara
International School of Engineering (ISE)
Faculty of Engineering
Chulalongkorn University
Bangkok, Thailand
poonpalida@gmail.com

Pavarit Wiriyakunakorn
International School of Engineering (ISE)
Faculty of Engineering
Chulalongkorn University
Bangkok, Thailand
pavarit.guide@gmail.com

Thiti Srikao
International School of Engineering (ISE)
Faculty of Engineering
Chulalongkorn University
Bangkok, Thailand
thiti.chopper555@gmail.com

Sirin Nitinawarat
International School of Engineering (ISE)
Faculty of Engineering
Chulalongkorn University
Bangkok, Thailand
sirin.n@chula.ac.th

***Abstract*— The proliferation of phishing attacks poses a significant threat to individuals and organizations worldwide. While existing phishing detection tools predominantly focus on technical aspects, there exists a notable void in addressing the informational needs of users beyond numerical data. To bridge this gap, we have developed a user-centric phishing URL detection tool empowered by machine learning (ML) models. The objective is twofold: to categorize phishing links effectively and to provide users with contextual understanding and educational resources to enhance their online safety. By integrating insights from interdisciplinary fields and leveraging advanced ML techniques, the research endeavors to empower users with actionable insights and foster informed decision-making in the digital realm Using a methodology that includes feature selection, algorithm selection, and model development, our goal is to deliver an interpretable model with high performance. The outcome of this project is a user-centric web application for phishing.**

Keywords—cyber security, feature selection, phishing, machine learning

I. Introduction

In today digitally interconnected world, the prevalence of online threats, particularly phishing attacks, pose significant challenges to personal cybersecurity. Phishing attacks, often disguised as legitimate communications, aim to deceive unsuspecting users into divulging sensitive information such as passwords, financial details, or personal data. These attacks exploit various vulnerabilities that make individuals susceptible to deception. These vulnerabilities include a lack of awareness about Uniform Resource Locators (URLs) and their functions, difficulty in distinguishing trustworthy URLs from potential threats, URL concealment or redirection, accidental clicks, and the inability to differentiate between legitimate and phishing URLs [1]. To address these multifaceted challenges, various approaches have emerged, including heuristic analysis, visual similarity assessment, list-based evaluations, and advanced techniques rooted in machine learning and deep learning [2].

Although several machine-learning based techniques have been studied in the area of phishing detection, there are still few feature selection approaches explored to determine which are the optimal set of features for a model. Moreover, few sample data sets were collected. In addition, available web applications phishing intervention found online, focus predominantly on technical aspects of phishing link detection. Consequently, there exists a notable gap in addressing users' informational needs comprehensively [3]. Users, often lacking deep technical knowledge, require more than just numerical data or verdicts; they seek contextual understanding and educational resources to navigate the digital landscape safely.

Recognizing this critical gap, this article endeavors to develop a comprehensive phishing URL detection tool enhanced with a capability of machine learning models. By categorizing phishing links and providing relevant informational contents tailored to identified threats, the tool not only can enhance users' awareness but also equip them with the knowledge necessary for making informed decisions online.

II. Design Description

The design aims to provide an effective phishing detection solution through two main components: Model Development and Web Application. The Model Development involves training a machine learning model to distinguish between legitimate and phishing URLs, while the web application serves as the user interface for accessing the detection functionalities.

A. Model Development

1.) Data Preparation: The process begins with data preparation, which involves data aggregation, balancing and preprocessing. Then, feature extraction is performed.

2.) Feature Engineering and Experimentation: Extracted features undergo experiments for feature selection and model training. The best model among all candidates is then selected for deployment in the web application. The chosen model uses the LightGBM algorithm and Recursive Feature Elimination with Cross-Validation (RFECV) for feature selection. The process of conducting experiments and the criteria for choosing the best model will be discussed later in Section III, IV.

B. Web Application

The goal of developing the web application is to provide a way to be able to effectively utilize machine learning models. To accomplish this, a front-end application and back-end application program were developed. The Front-end application will allow users to interact with the input URLs for analysis, where the server-side component integrates the trained model and performs the URL classification.

Fig. 1. Overview of the application structure and technologies in each part.

1) Front-end Application: The Front-end application needs to fulfill several key functionalities: allow users to input URLs to be scanned, display scan results, and provide a way to explore other results that were previously created. It is also important to create an good baseline for development by considering what technology to use and the cost to deploy the application.

a) Framework. NextJS is chosen to take advantage of its server-side rendering, routing, and other features it provides out of the box. With Typescript, static typing can further help improve code quality during the development process. Tailwind CSS is implemented to eliminate the need to write CSS classes for each individual UI component.

b) Deployment. The Front-end application is deployed on Vercel. The service is chosen due to the ease of deployment and other useful features such as automatic scaling, continuous deployment, and built-in analytics.

2) User Interface Design: The design of the application needs to follow concepts such as simplicity, clarity, and consistency, which can be achieved by following UI/UX best practices.

a) URL Scan Page. Concepts such as Miller's Law and Gestalt Principles are also utilized to help create a more cohesive UI by reducing distance between elements and giving them appropriate sizing with appropriate grouping to improve ease of use.

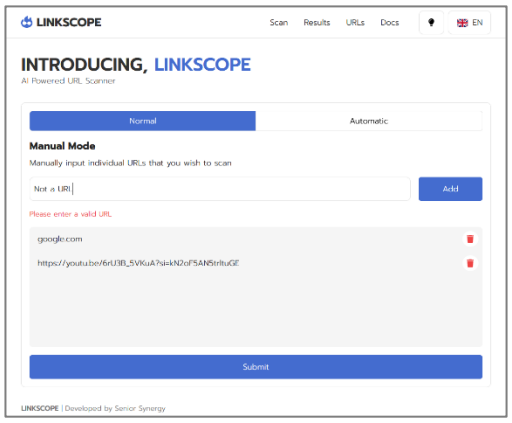

Fig. 2. A screenshot of the scan page of the application and the different states the scanner interface can be in.

b) Search Page. This page utilize Jakob's Law, which focuses on the use of UI design with established conventions. In this case, search and filters follows the design that would be commonly found on other established websites or applications.

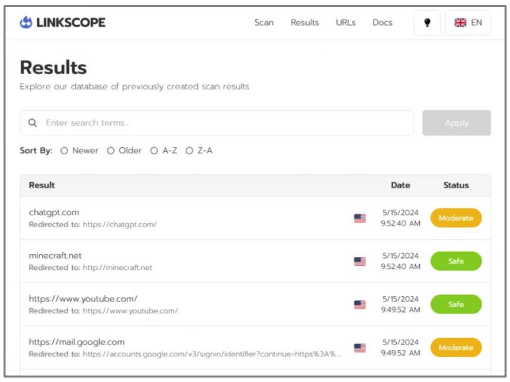

Fig. 3. A screenshot of the search page of the application.

c) Result Page. Since there is a need to display a lot of information in this page, Miller's Law, information of different groups are broken into different parts so that the user can focus on each part of the result without being overwhelmed.

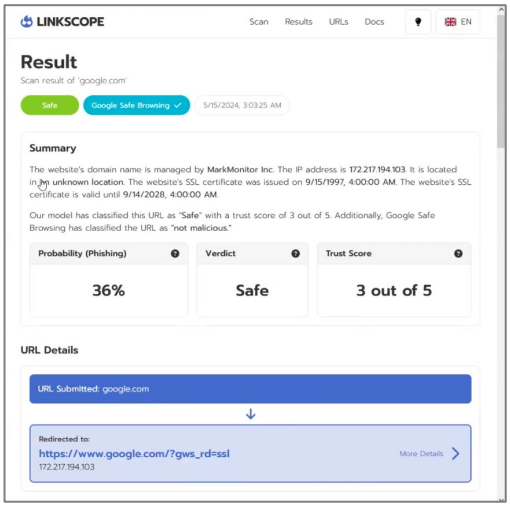

Fig. 4. A screenshot of the result page of the application.

3) Back-end Application: The back-end application needs to facilitates all the requests from the Front-end application.

a) Python for Back-end.. FastAPI is utilized to create RESTful APIs for the Front-end application. It also unifies the need for a python environment in the back-end application for the serialized trained model to be deployed and utilized

b) Database. MySQL is chosen as the relational database management system dut to its robust support for a wide range of applications and its seamless integration with AWS.

c) Deployment. AWS Lambda was chosen to deploy the back-end application. While not commonly used, benefitting from cost efficiency of serverless back-end is key for sustainability.

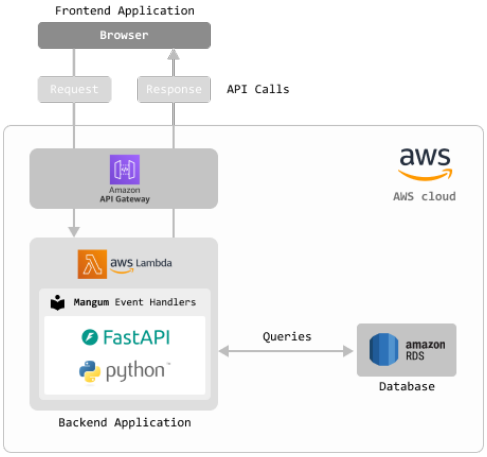

Fig. 5. A diagram of the overall structure of the back-end application and how it interacts with other parts of the application.

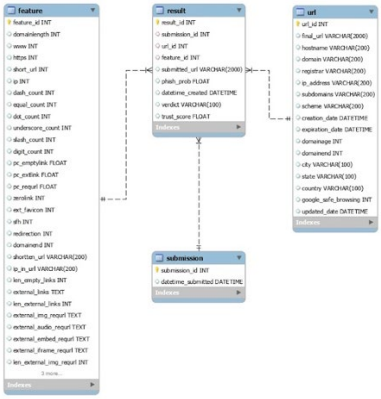

Fig. 6. Entity relationship diagram.

III. Experiment Setup and Methods

The experimental setup consists of a systematic and comprehensive approach to develop an effective phishing detection model and gather insights from usability testing to understand users' needs. This section is divided into two parts: experimental setup and methods for both the model experiment and usability testing.

A. Model Experiment

For the Model Experiment Setup, four components must be defined before conducting the experiment to select the best model: dataset, features explored, feature selection approaches, and algorithms used.

1) Dataset: The dataset consists of a total of 70,000 URLs, with 35,000 phishing URLs collected from PhishTank and PhishScore, and 35,000 safe URLs derived from top SEO domain lists. All the links have been checked, and no duplication was found. The data then underwent k-fold cross-validation with k=5. This technique divides the dataset into five equally sized subsets (folds). For each fold, the model is trained on four folds (56,000 URLs) and tested on the remaining fold (14,000 URLs). This process is repeated five times, with each fold serving as the test set exactly once. This approach ensures that the model is validated on different subsets of the data, providing a more robust evaluation of its performance.

2) Features Explored: After obtaining all the URLs, feature extraction is performed. In this study, 29 features are selected for analysis, categorized into four types:

a) Address Bar-based: This category examines the characteristics of the URL address.

TABLE I. Address Bar Based Features Explored.

Feature Name	Explanation
domainlength	count the characters in hostname string
www	If url has 'www' as subdomain then return 0, else 1
subdomain	If url has more than 1 subdomain then return 1, else 0
https	If url contains 'https' then return 0, else 1
http	If url contains 'http' then return 0, else 1
short_url	If url is a short url returns 1,else 0
ip	If url contains 'ip address' returns 1,else 0
@	Count the '@' characters in url
-	Count the '-' characters in url
=	Count the '=' characters in url
.	Count the '.' characters in url hostname
_	Count the '_' characters in url
/	Count the '/' characters in url
digit	Count the digits (0-9) characters in url
log	If url contains 'log' word in url then return 0, else 1
pay	If url contains 'pay' word in url then return 0, else 1
web	If url contains 'web' word in url then return 0, else 1
cmd	If url contains 'cmd' word in url then return 0, else 1
account	If url contains 'account' word in url then return 0, else 1

b) HTML/DOM Structure-based: This category involves analyzing the HTML or DOM structure of the web page.

TABLE II. HTML/DOM Structure Based Features Explored.

Feature Name	Explanation
pcemptylinks	Percentage of empty links ***Note:*** An empty link is a hyperlink that does not lead to a different web page. When clicked, it typically results in staying on the current page or displaying a blank page. This can occur when the link's URL is missing or invalid.
pcextlinks	Percentage of external links that direct you to another site with a different domain from the submitted URL. ***Note:*** An external link is a hyperlink that directs users from one website to a different website with a different domain.
pcrequrl	Percentage of external resource URLs: ***Note:*** An external resource URL refers to the web address of a resource, such as images, audio files, or embedded content, that is hosted on a different domain from the main or submitted URL.

zerolink	If the URL page has no links in the HTML body, return 1; otherwise, return 0.
extfavicon	If the favicon URL is from a different domain than the submitted URL, return 1; otherwise, return 0.
submit2email	If the html page contains "\b(mail\(\)|mailto:?)\b" then return 1, else 0
sfh	SFHs that contain an empty string or "about:blank" or lead to different domain sites from submitted url, like form['action'] == "" or form['action'] == "about:blank" then return 1, else return 0

c) Abnormal-based: This category assesses the abnormal behavior of the submitted URL.

TABLE III. ABNORMALITY BASED FEATURES EXPLORED.

Feature Name	Explanation
redirection	If clicking the submitted URL results in a redirection to another URL, return 1; otherwise, return 0. For example, clicking www.eabc1255.com and being redirected to www.eabc5255.com.

d) Domain-based: This category considers factors related to domain registration and expiration dates.

TABLE IV. LIST OF DOMAIN BASED FEATURES EXPLORED.

Feature Name	Explanation
domainage	The difference between expiration time and creation time, if the domain age is less than 6 months then return 1, else return 0
domainend	Calculate the difference in days between the current date and the expiration date (registration length). If the difference is less than or equal to one year, return 1; otherwise, return 0.

3) Feature Selection Approaches: The purpose of feature selection is to select a subset of relevant features from a large number of available features to achieve similar or even better classification performance than using all features.

a) Recursive Feature Elimination with Cross-Validation (RFECV): Recursive Feature Elimination with Cross-Validation is a robust technique for feature selection that combines recursive feature elimination with the power of cross-validation. The process begins by building a model with all available features and then recursively removing the least significant features, one by one. At each step, the model is evaluated using cross-validation to determine its performance. This approach helps to identify the optimal set of features that contributes to the best model performance while avoiding overfitting.

b) Particle Swarm Optimization (PSO): Particle Swarm Optimization (PSO) is a population based technique to address feature selection problems in this project due to better representation, capability of searching large spaces, and being less expensive computationally. In Particle Swarm Optimization (PSO), a group of candidate solutions, known as a swarm, is represented by particles within a defined search space. The algorithm begins by randomly initializing the position of each particle. The particles then traverse the search space, adjusting their positions based on their own individual experiences, aiming to find the optimal solution.

c) Random Forest Feature Selection: Random Forest Feature Selection is an embedded method that uses the importance scores from a Random Forest model to select the most relevant features. As the model is trained, each feature's contribution to prediction is assessed by its frequency in splitting decision tree nodes. Features with higher scores are deemed more significant. This approach automatically identifies key features, reducing the need for manual selection, and is especially effective in high-dimensional data scenarios.

4) Algorithms: Algorithm selection involves choosing the most suitable algorithms for a specific task based on their performance characteristics. The following are the selected algorithms in three different approaches.

a) Random Forest (RF) Algorithm: Random Forest is recognized for offering the highest accuracy among machine learning algorithms and is frequently employed in phishing URL detection. Random forests address the problem of overfitting in decision trees by creating multiple trees, each trained on random subsets of features.

b) LightGBM: Identified as the second-highest performer in the research. It is considered to be one of the best choices in various machine learning applications, including phishing URL detection. LightGBM is a fast and efficient tool for building machine learning models.

c) Support Vector Classification (SVC): Support Vector Classification (SVC) is a machine learning method used to classify data into different categories. It's a part of the larger family of Support Vector Machines (SVMs). SVC works by finding the best dividing line (or hyperplane) that separates data into its respective classes. It does this by maximizing the margin, which is the distance between the line and the nearest data points from each class.

According to Fig.1, the experimental process for selecting the best model begins after data preparation and feature extraction. The experiment encompasses three main processes: feature selection, algorithm selection, and model evaluation. For feature selection, as stated in Section 5.1.1, RFECV, PSO, and Embedded methods are used to select optimal features from the total of 29 features derived from the feature extraction process. This approach yields three optimal subsets of features, one from each selection method. Next, three algorithms—Random Forests, LightGBM, and SVC—are used to train models on each of these subsets.

Finally, model evaluation is conducted to select the best model and determine which subset of features and which algorithm produces the best performance. This assessment uses a range of performance metrics, including accuracy, precision, recall, and false positive rate. This comprehensive evaluation process allows for an accurate and reliable assessment of the model's effectiveness in detecting phishing attempts.

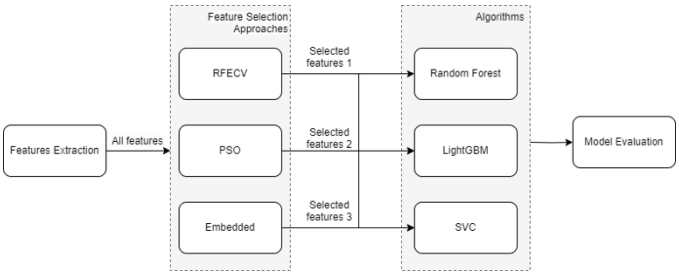

Fig. 7. Machine Learning Experiment Process

B. *Usability Testing*

In conducting usability testing for our phishing detection application, our primary objective is to assess its effectiveness, efficiency, and user satisfaction. To achieve this, the test operator will inquire participants to ensure a diverse range of perspectives. The testing environment can be either online, using platforms such as Zoom and Discord, or in-person for face-to-face interaction. Participants will begin with an introductory session where they provide demographic information and details about their prior experience with similar applications.

During the testing session, participants will be asked to perform specific tasks such as scanning a URL for phishing, submitting a URL, and interpreting the results provided by the application. Additionally, they will be tasked with searching within the application. We will meticulously record task completion times, identify any errors encountered, and evaluate the effectiveness and efficiency of the application in fulfilling its intended purpose. Following the tasks, we will conduct post-interviews to gather feedback on usability, user satisfaction, and areas for improvement, ensuring that the application meets the needs and expectations of its users.

The web application usability testing aimed to assess the effectiveness, efficiency, and user satisfaction of the phishing detection application. Recruitment for the test involved inviting participants, primarily friends and family, who were briefed on the test's purpose and invited to participate online or in person. During the test execution, participants were asked to complete two tasks: using a scan function and a search function.

In the scan function, the test operator provided a URL, submitted it to the phishing detection application, and interpreted the results. The search function required participants to find information about the phishing web on the application's database. Task completion time and number of errors were recorded, and participants were evaluated based on their performance.

Following functionality testing, participants filled out a questionnaire form using a Likert scale to assess various aspects of usability. They rated their self-efficiency, ease of use, user-friendliness, behavioral intention, and security awareness.

Self Efficiency. The goal is to determine if users can use the application skillfully, and whether they feel the need for a specialist to help using it. This will help evaluate the level of confidence when the user is using the application.

Ease of Use. The goal is to determine if users can think that the overall design of the application is easy and straightforward enough to navigate during the real usage.

User-friendliness. The goal is to determine if the design of the application is well-organized and easy to find different elements, as well as how visually appealing the user interface is to the average user.

Behavior Intention. The goal is to see if users are likely going to be using the application in the future.

Security Awareness. The goal is to evaluate if the application itself has provided the value that it aimed to provide to the users regarding phishing URL detection and the informational content that the application provides.

Additionally, post-interviews were conducted to gather qualitative feedback. Participants rated their overall experience, identified useful features or redundancies, commented on task completion times, described any encountered issues, expressed confidence in using the application in the future, and provided suggestions for improvement.

Data from the questionnaire forms, task performance, and post-interviews were collected and analyzed to evaluate the application's usability. This involved analyzing Likert scale responses, identifying patterns in task completion times and errors, and extracting insights from participants' feedback to inform future improvements to the application.

IV. Results and Discussions

A. *Model Results*

The results are obtained from applying the machine learning algorithms with different feature selection approaches. In order to select the best model for the web application, LightGBM is chosen for its high accuracy, leading among three algorithms.

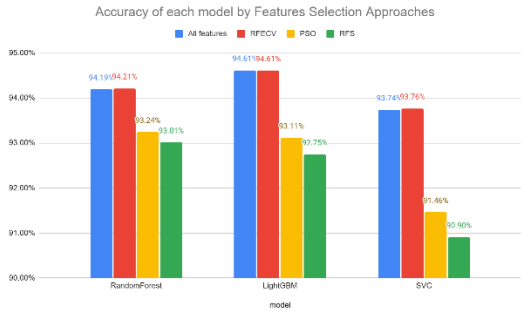

Fig. 8. Accuracy of each model by Feature Selection Approaches.

However, in the context of a web application, processing time is also a critical factor. When users submit URLs for analysis, the feature extraction process takes time, adding to the overall latency. This consideration makes it important to balance accuracy with processing time.

According to Fig. 8, LightGBM with all features and with RFECV provides approximately the same highest accuracies at 96.49%. However, having all features extracted is a drawback for the web application, where user experience depends on quick responses. Given this context, LightGBM with RFECV is chosen, reducing the features to 26.

TABLE V. RESULTS OF LIGHTGBM WITH RFECV

Algorithm	Accuracy	Precision	Recall	F1-score
LightGBM	94.61%	95.68%	93.44%	94.61%
LightGBM (Hyperparameterized)	95.07%	96.00%	94.05%	95.07%

According to Table V, the classification results of LightGBM with RFECV are presented. The hyperparameter tuning process increased the accuracy to 95.07%. The features were selected using Recursive Feature Elimination with Cross-Validation (RFECV), resulting in a reduced set of 26 features. These features include 'domainlength', 'www', 'subdomain', 'https', 'short_url', '@', '-', '=', '.', '_', '/', 'digit', 'log', 'pay', 'web', 'account', 'pcemptylinks', 'pcextlinks', 'pcrequrl', 'zerolink', 'extfavicon', 'submit2email', 'sfh', 'redirection', 'domainage', and 'domainend'. This set of features, along with the hyperparameterized LightGBM model, is chosen for deployment in the application.

B. Web Application Usability Testing Result

From the results from usability testing, average duration for completing each task ranged from 30 seconds to 1, providing valuable insights into the clarity and user-friendliness of the application interface. Nonetheless, no errors were encountered during task completion, indicating that even novice users were able to navigate and utilize the application effectively.

TABLE VI. RESULT FROM THE QUESTIONNAIRE.

Category	Question	Avg. Score	Category Avg. Score
Self efficiency	I can use the application skillfully	3.88	3.67
	I don't need specialist to help using the application	3.50	
	I feel confident using the application	3.63	
Ease of use	Using the application is easy and straightforward	3.75	4.04
	I find it easy to navigate the application	4.25	
	The steps to use the application are clear and simple	4.13	
User friendliness	The application has a user friendly interface	4.25	4.13
	The design of the application is visually appealing	4.00	
	The features in the application are well-organized and easy to find.	4.13	
Behavior Intention	I intend to use the application myself in the future	4.25	3.96
	I would recommend the application to my friends	4.13	
	I would likely choose the application over other similar tools in the future.	3.50	
Security Awareness	Using the application raises me awareness in security aspects	4.25	4.38
	I understand more about phishing websites after using the application	4.38	
	The application helps me recognize phishing website characteristics	4.50	

V. CONCLUSION

This report outlines the development of a user-centric phishing URL detection tool, leveraging interpretable machine learning models. By addressing the gap in existing tools that often overlook user information needs, we utilized Recursive Feature Elimination (RFECV) technique for feature selection and LightGBM model due to its highest performance. The resulting web application received positive feedback from usability testing. In the future, efforts will focus on enhancing the user interface, integrating multilingual support, and continuously updating the detection model to counter evolving phishing techniques.

REFERENCES

[1] E. Büber, Ö. Demir and Ö. K Şahingöz, "Feature Selections for the Machine Learning Based Detection of Phishing Websites," 2017 International Artificial Intelligence and Data Processing Symposium (IDAP), Malatya, Turkey, 2017, pp.1-5, doi: 10.1109/idap.2017.809031.

[2] A. Safi and S. Singh, "A Systematic Literature Review on Phishing Website Detection Techniques," J. of King Saud University: Computer and Information Sciences, vol. 35(2), pp. 590–611, Feb 2023, doi: 10.1016/j.jksuci.2023.01.004.

[3] O. Sarker, S. Haggag, A. Jayatilaka and C. Liu, "Personalized Guidelines for Design, Implementation and Evaluation of Anti-Phishing Interventions," *2023 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, New Orleans, LA, USA, 2023, pp. 1-12, doi: 10.1109/ESEM56168.2023.10304861.