WillRo: A Deep Learning App for Potential Phishing Threat Detection

Willy Sotelo
Faculty of Information Systems Engineering, Universidad Peruana de Ciencias Aplicadas, San Isidro, Lima, Peru
u202015236@upc.edu.pe

Alvaro Roque
Faculty of Information Systems Engineering, Universidad Peruana de Ciencias Aplicadas, San Isidro, Lima, Peru
u202013176@upc.edu.pe

Sandra Wong-Durand
Faculty of Information Systems Engineering, Universidad Peruana de Ciencias Aplicadas, San Isidro, Lima, Peru
pcsiswon@upc.edu.pe (corresponding author)

Pedro Castaneda
Faculty of Systems Engineering and Electrical Mechanics, Universidad Nacional Toribio Rodriguez de Mendoza, Amazonas, Peru
pedro.castaneda@untrm.edu.pe

Alejandra Onate-Andino
Escuela Superior Politecnica de Chimborazo (ESPOCH), Riobamba, Ecuador
monate@espoch.edu.ec

Received: 19 August 2025 | Revised: 22 September 2025 and 11 October 2025 | Accepted: 13 October 2025

 DOI: https://doi.org/10.48084/etasr.14161

ABSTRACT

This article presents an application called WillRo App, designed to detect potential phishing by analyzing website screenshots in real time. The system integrated Robotic Process Automation (RPA) to capture screenshots, and the YOLOv5 deep learning model in order to classify phishing and no-phishing content. The results demonstrated a precision of 85.80%, a recall of 93.00%, a mAP@0.5 of 66.60%, and a mAP@0.5 -0.95 of 32.70%. These values showed a reliable detection performance, making WillRo a possible model for phishing detection. Future work should focus on improving the model with additional features to increase its accuracy.

Keywords-phishing detection; deep learning; malicious content; Robotic Process Automation (RPA)

I. INTRODUCTION

Phishing is a cyberattack type that can both enable and facilitate the stealing of highly sensitive data. Hackers often impersonate employees or reliable contacts, creating fake websites that look like the originals to provide malicious content. According to [1], phishing was responsible for 22%-24% of the total cyberattacks in the first months of 2024, leading to substantial losses of personal data. In 2023, authors in [2] stated that threat actors applied BEC 2.0 methods to take control of the email accounts of university students, which they used for leaking academic and research data without detection. During the same period, both individuals and companies suffered from phishing campaigns, leading to the loss of confidential information and the gaining of unauthorized access to bank services and online platforms.

Several approaches based on Machine Learning (ML) have been proposed to mitigate this issue. Specifically, authors in [3] developed a phishing detection system that reached an accuracy of 98.20%, while in [4], a precision rate of 99.60% was achieved in detecting phishing emails. However, these solutions remain restricted to controlled environments and email-based detection, limiting their applicability to real-world scenarios involving websites. Based on these prior

contributions, the present study introduces an application designed to detect phishing through image recognition combined with RPA for automated screenshot acquisition. The system combines a deep learning algorithm to analyze content, generate detection results, and provide user recommendations through an interactive dashboard, while maintaining records in a database.

II. LITERATURE REVIEW

The related works reviewed in this study were organized into four groups. The first group focuses on ML and comprises five contributions. In the first work [5], the K-Nearest Neighbors (KNN) algorithm was applied to detect phishing attacks on mobile devices, with an accuracy of 95.62% after training and testing. Similarly, authors in [6] presented a model for identifying phishing websites through a Hybrid Feature Set, achieving 99.17% accuracy with a balanced dataset of 13,000 benign and 13,000 malicious URLs. The third contribution developed a model that was trained using two datasets consisting of 49,400 benign and 50,175 malicious URLs, resulting in an average accuracy of 96.05% [7]. In the fourth study [8], different techniques for training ML models were examined, using 54,000 records for training and 12,000 for testing. Naïve Bayes delivered the best results at 95.65% accuracy. Authors in [9] proposed a method for the identification of phishing URLs based on a dataset of 114,996 URLs for training and 23,000 for validation. This approach, along with the Light Gradient Boosting Machine (LGBM) model, reached an accuracy of 95.41%.

The second group addressed deep learning, including six works. The first one introduced a browser extension using Convolutional Neural Networks (CNNs), trained on 651,191 URL samples and achieving 98.42% accuracy [10]. Authors in [11] evaluated five algorithms, including ANN, CNN, RRN, BRNN, and AN, reporting accuracies of 88.0% with 50 samples and 94.0% with 10,000 samples. A BERT-based deep learning technique trained on 549,346 Kaggle entries was investigated, obtaining 96.66% accuracy [12]. Similarly, in [13], a model was developed to identify spam comments on YouTube, using the UCI dataset of benign and malicious URLs, yielding 96.43% accuracy. Additionally, the AntiPhishStack framework combined stacking generalization with LSTM networks to detect phishing across multiple web services, using datasets from Alexa and PhishTank [14]. The best performance was achieved with an SVM model, which reached 88.72% accuracy.

The third group emphasized the use of RPA. The first study demonstrated an RPA-based model that could perform human-like tasks 100% accurately when tested with 10, 20, and 50 cases [15]. In [16], an RPA model was presented, which was able to classify and prioritize customer complaints into three categories: "did not complain," "complain," and "micro average," with each class achieving 99% accuracy. Similarly, authors in [17] automated RPA and utilized it to streamline financial accounting tasks in companies, resulting in reconciliation time from 60 min to just 2 s. In [18], a framework was proposed integrating RPA with Computer Vision (CV) and Natural Language Processing (NLP). Experiments with a dataset of 252 scanned documents and 18,326 entities demonstrated a 15% efficiency improvement. The last study presented a methodology that applied RPA for digital task selection in recruitment processes, with evaluation times ranging between 1 and 20 min depending on the case [19].

The fourth and final group focused on detecting vulnerabilities with ML. The first work introduced a framework capable of detecting both phishing attempts and web vulnerabilities, using datasets comprising 4,300 URLs, which reached 88.90% accuracy [20]. Authors in [21] developed models for classifying URLs as benign or phishing, reporting exceptionally high precision rates between 99.93% and 99.98%. A further contribution proposed an MLP-based detection algorithm that supports the training of other models, yielding 91.28% accuracy [22]. The ContractWard model was also presented, trained with 49,502 samples, and produced a precision rate of 93.33% [23]. Finally, the last work introduced a method for phishing detection on websites, reporting accuracy results ranging from 90.61% to 99.00% after testing [24].

III. SYSTEM DESIGN

A. Architecture

The architecture of the WillRo App is designed to detect phishing threats and is structured into four main components: User, Front-End, Functions, and Cloud Service, as illustrated in Figure 1. The architecture begins with the user. The latter, in this case, is a university student who accesses both the application and a website through Google Chrome. It is important to emphasize that the application should be installed beforehand on a laptop or desktop computer to ensure proper operation. Once this requirement is satisfied, the user initiates the system's workflow.

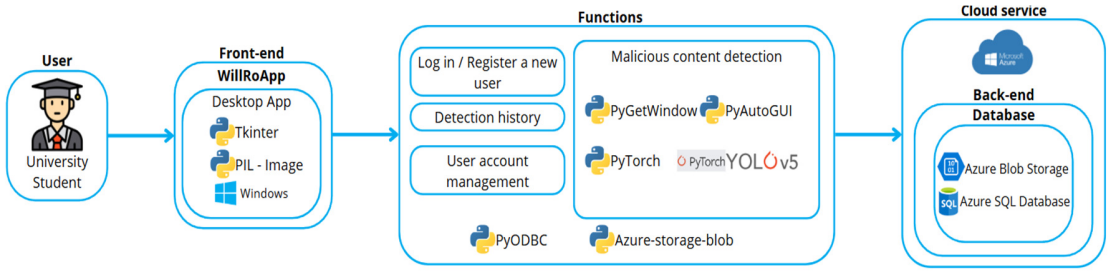

Fig. 1. Architecture of the WillRo app.

The second component, the front-end, operated on the Windows platform to enhance accessibility. Python is employed as the main programming language, while Tkinter provides the Graphical User Interface (GUI), including interactive elements such as buttons and menus. For image processing, the PIL Image library is used to resize and manipulate screenshots.

The third component involves the core functionalities of the application. These include user authentication, which allows account registration and login; a detection history module, which stores and retrieves previous detection results; and account management, which permits users to update personal details or permanently delete their accounts. The primary functionality, however, is malicious content detection, where screenshots are analyzed to identify potential threats. This section also integrates several supporting libraries:

- PyODBC allows the connection between the application and the database.
- Azure Storage Blob is used for the cloud storage of screenshots.
- PyGetWindows allows window control within the application. For instance, when it finishes the detection, it closes and opens another window to proceed with the results.
- PyAutoGUI automates the actions of the device in order to take a screenshot automatically.
- PyTorch is for model training and use, combined with YOLOv5, which helps with the detection of malicious content.

The fourth and final component is the cloud service, implemented using Microsoft Azure. It serves as the database infrastructure responsible for storing user account information, screenshots, detection outcomes, and the recommendations generated by the application.

B. User Interface Design

User interface design is important for achieving good interaction between the user and the application. It welcomes the user and offers them the option to log in with an existing account or register a new one. From this screen, the user can proceed to the registration form, where a username and a password (entered twice for confirmation) are required to create a new account. Alternatively, the user may access the login interface, which requests valid credentials (username and password) to enter the system. Both the registration and login views are accessible as continuations of the initial screen, with navigation buttons allowing users to return to the welcome page or switch between these options. Figure 2 illustrates the main menu interface, which welcomes the user. From this menu, the first button directs the user to the malicious content detection module, the primary functionality of the application. The second button grants access to the detection history, where previous analyses, results, and recommendations are displayed. The gear icon leads to the account management section, enabling the user to update personal data or delete the account. Finally, the last button allows the user to sign out.

Fig. 2. Interface of the main menu.

As depicted in Figure 3, the interface of the malicious content detection menu allows the user to either initiate the detection process or return to the main menu. From this screen, the process continues with the first phase, in which a screenshot of the website is captured using RPA, followed by the second phase, where the captured screenshot is analyzed to identify potential threats.

Fig. 3. Interfaces of the malicious content detection.

Figure 4 presents the final two phases of the malicious content detection process. On the left, the third phase shows the analysis results, presented through a pie chart, indicating the probability of phishing versus non-phishing, along with a button to proceed to the final stage. On the right, the fourth phase provides recommendations based on the analysis outcomes, with an option to save the results and return to the main menu.

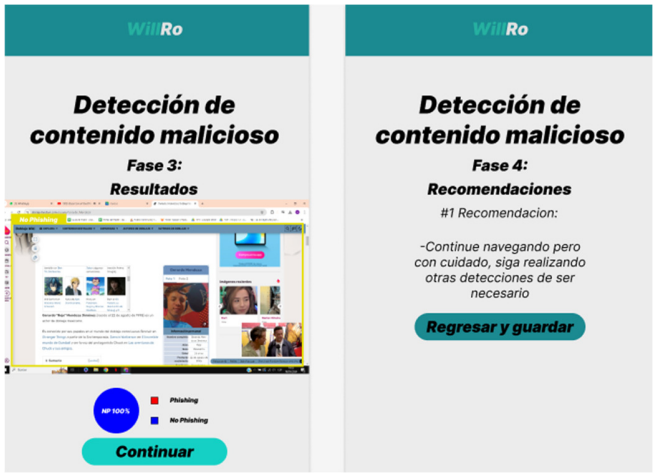

Fig. 4. Interfaces of results and recommendations.

Figure 5 displays the detection history interface. This screen provides detailed information about past analyses, including the date and time of detection, whether phishing was identified, the corresponding screenshot stored as a URL, and the generated recommendations.

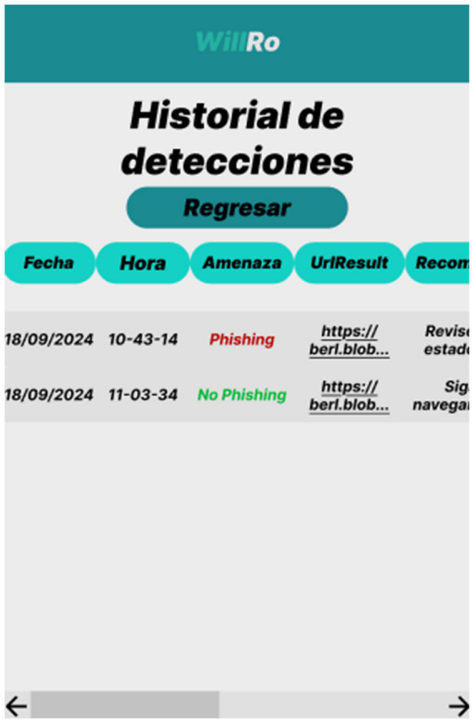

Fig. 5. Detection history interface.

The main screen provides options to update user information, delete the account, or return to the main menu. From this menu, the user can proceed to the update screen, where credentials, such as username and password, may be modified with buttons to save the changes or return to the account management menu. As a further continuation, the account deletion screen allows the user either to confirm the removal of the account or return to the previous menu.

IV. METHODOLOGY

A. Dataset

In this study, the dataset was obtained from Kaggle [25] through a combination of three different sources, each containing website images classified as phishing and non-phishing:

- Phish-Iris Dataset, which consists of 1,101 screenshots divided into phishing and non-phishing categories [26]
- Counterfeit Affiliate Programs (CAP) dataset, containing a total of 121 screenshots [27]
- Phishing dataset, which includes 983 website screenshots [28]

After merging the three datasets, the final collection comprised 2,205 screenshots, distributed as follows: 1,159 for training (52.56%), 686 for validation (32.84%), and 85 for testing (14.60%). The screenshots were labeled using MakeSense.ai with the labels "Phishing" and "No Phishing," enabling the algorithm to distinguish between the two classes during the training process.

B. Model

The utilized model was YOLOv5 [29], which is based on the You Only Look Once (YOLO) framework. As an open-source algorithm, YOLOv5 is widely used for object detection in images and videos. In this work, it was applied to analyze screenshots in order to determine whether they contain malicious content.

C. Pre-Processing

YOLOv5 requires datasets in JPG or PNG format, along with corresponding labels. For this project, the labeling process was carried out using MakeSense.ai to ensure the proper training of the model. Additionally, a YAML configuration file was prepared to define the parameters and associate the labels with the model.

D. Training

The training phase was conducted on Google Colab, employing Python and the prepared dataset to evaluate the model's precision. A total of 70 epochs were executed, using the labels "Phishing" and "No Phishing" to optimize detection performance.

E. Evaluation and Statistical Analysis

Table I shows the metrics used for evaluation and statistical analysis [30], along with their descriptions and formulas.

TABLE I. METRICS WITH DESCRIPTION AND FORMULA

#	Metric	Description	Formula
1	Precision	Precision on the correct detection of objects	TP/(TP + FP)
2	Recall (R)	Ability of the model to identify the instances of objects in images	TP/(TP + FN)
3	mAP@0.5	Mean average precision calculated at an Intersection over Union (IoU) threshold of 0.50	$\frac{1}{N}\sum_{i=1}^{N} APi(IoU = 0.5)$
4	mAP@0.5-95	The average of the mean average precision calculated at varying IoU thresholds, ranging from 0.50 to 0.95	$\frac{1}{N}\sum_{i=1}^{N}\frac{1}{10}\sum_{j=1}^{10} APi(IoU = 0.5 + 0.05x(j-1)$
5	F1-score	Harmonic mean between precision and recall	$2\,x\frac{Precision\ x\ Recall}{Precision\ +\ Recall}$

V. RESULTS

Figure 6 illustrates the training and testing results for each class. The sum of each metric for all classes gave a total precision of 85.8% indicating strong accuracy in detecting malicious content. A recall of 93% demonstrated a high proportion of true positives and the ability to correctly identify most actual phishing instances. Additionally, the mAP@0.5 metric yielded 66.6%, suggesting that the system provides a reasonable detection performance within an acceptable margin of error, effectively fulfilling its intended purpose. Conversely, the mAP@0.5 -0.95 metric recorded 32.70%, which reflects an acceptable but limited performance, highlighting the need for further improvements in detecting specific screen elements. Finally, the F1-score achieved 86.26%, showing a balanced trade-off between precision and recall. This means that the application can be both reliable and efficient, producing accurate results while minimizing errors.

Class	P	R	mAP50	mAP50-95:
all	0.286	0.31	0.222	0.109
Phishing	0.224	0.134	0.124	0.063
No Phishing	0.348	0.486	0.32	0.155

Fig. 6. Results of training and testing.

In Figure 7, the F1-confidence curve illustrates the optimal confidence threshold required to achieve a balanced performance of the model and the application. This balance relied on the integration of both precision and recall, as these metrics are fundamental components of the F1-score formula. All classes 0.29 at 0.147 implied that the model's highest average F1-score (0.29) occurred at a confidence threshold of 0.147, reflecting the best balance between precision and recall. Figure 8 presents the precision-recall curve, which reflects the trade-off between the two metrics. A downward trend indicates that the model identified more actual positives but at the cost of introducing incorrect predictions. With a precision of 85.80%, the results demonstrated that the application achieved a strong accuracy in detecting malicious content, with potential for improvement as the dataset expands. All classes 0.222 mAP@0.5 denoted that the model reached a mean average precision of 0.222 across all classes at an IoU threshold of 0.5, summarizing its overall precision-recall performance.

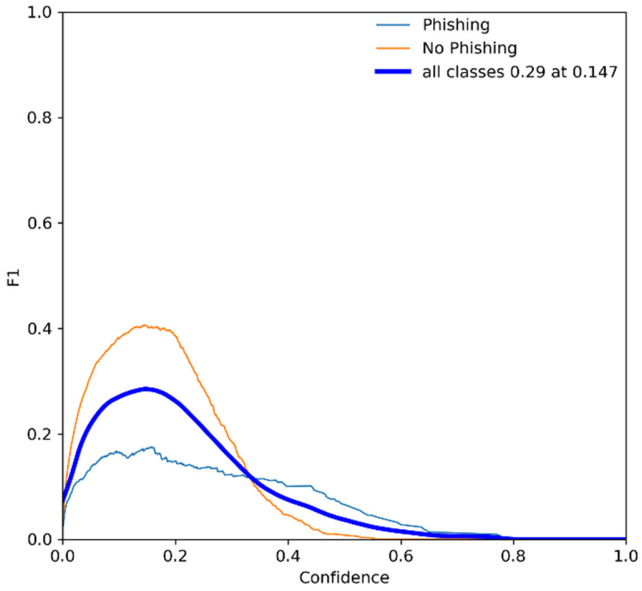

Fig. 7. F1-confidence curve.

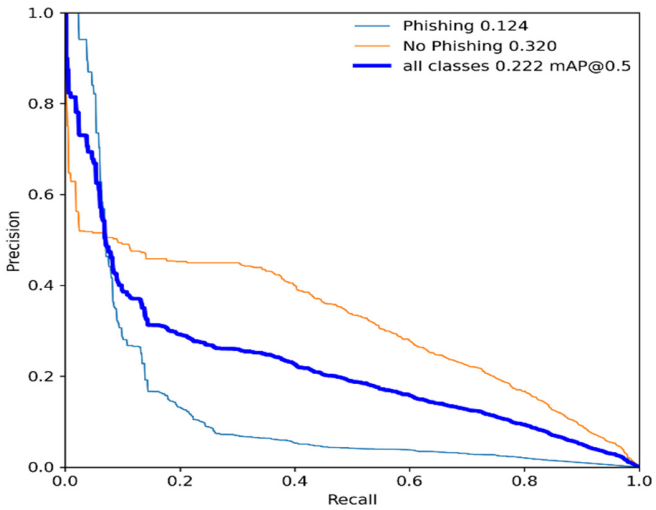

Fig. 8. Precision-recall curve.

Figure 9 depicts the precision-confidence curve, which evaluates how effectively the model maintains accuracy across different confidence levels. The results confirmed that the application performed consistently well when distinguishing between phishing and non-phishing screenshots. All classes refers/refer to the model's overall precision when considering both "Phishing" and "No Phishing" predictions together. A value of 1.00 indicated that, at this specific confidence threshold, every single prediction the model made was correct.

Figure 10 shows the recall-confidence curve, which measures how confidence thresholds affect the model's ability to detect true positives. The curve suggested that the application is well optimized, ensuring reliable detection performance. A value of 0.82 at 0.000 revealed that before any low-confidence predictions was/were filtered out (at a confidence threshold of 0), the model successfully identified 82% of all relevant instances across both classes.

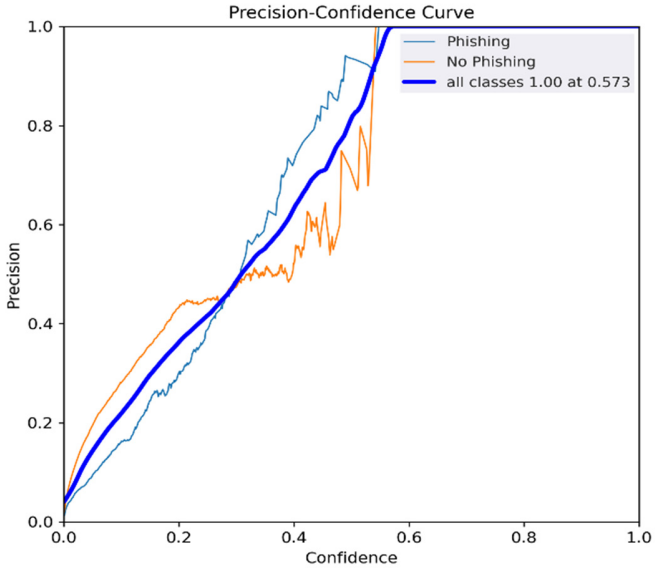

Fig. 9. Precision-confidence curve.

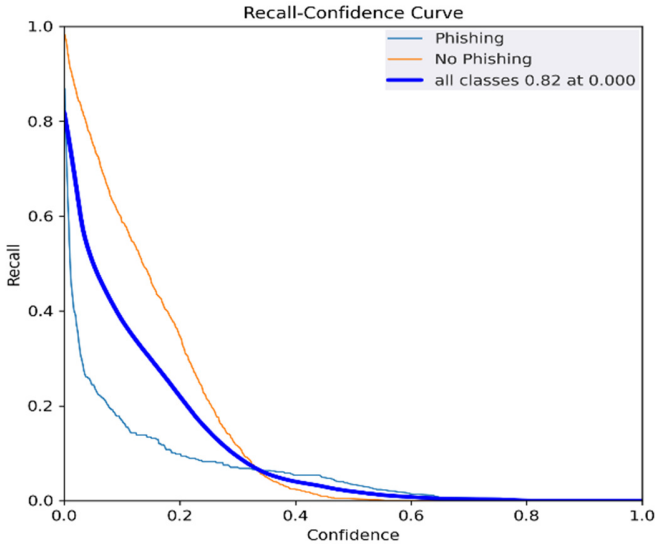

Fig. 10. Recall-confidence curve.

Figure 11 illustrates the confusion matrix, a key evaluation tool that summarizes the distribution of true positives, false positives, true negatives, and false negatives. The balance observed in the matrix indicates that the application performs effectively, minimizing misclassifications. All classes represented the overall recall, combining the "Phishing" and "No Phishing" results. Background is a class created by the model and represents all image regions where no ground-truth objects are present. It functions as an implicit negative class used exclusively for evaluation purposes, capturing instances in which the model produces detections in areas that should contain no labeled objects. Finally, Figure 12 displays an example of the model's output to the user after analysis. The system highlighted sections of the screenshot with bounding boxes, marking areas that may contain phishing indicators or safe content. The red box displayed on the screen represents the estimated percentage of potential phishing contained in that element. Leveraging the dataset and training process, the model is capable of detecting elements such as headers and webpage sections to identify potential threats.

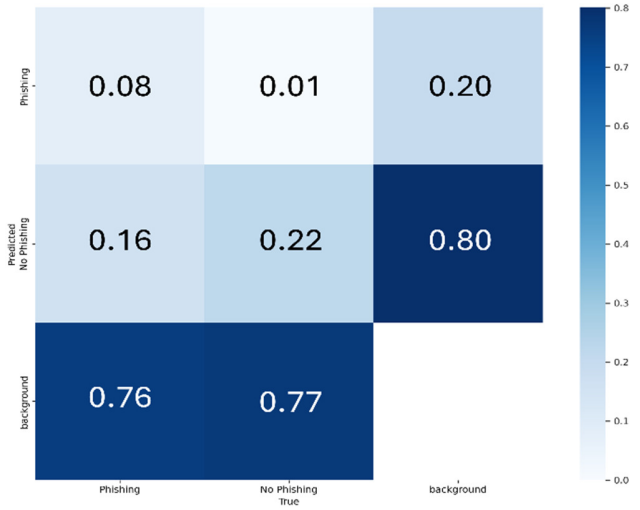

Fig. 11. Confusion matrix.

Fig. 12. Analysis result.

VI. DISCUSSION

The results of this study demonstrated the applicability of YOLOv5 for analyzing webpage screenshots in the context of phishing detection, broadening its application from the typical object recognition tasks to phishing detection. The model's precision of 85.80% was lower than that of some comparable works, but the higher recall (93.00%) highlighted its effectiveness in identifying true phishing instances.

Unlike many existing approaches that rely on URL-based detection - which attackers can bypass by frequently altering domain names - the present work emphasized the structural analysis of websites. Since modifying the visual composition of a site requires greater effort and resources, this strategy increases the robustness of phishing detection.

In addition, the integration of RPA streamlined the acquisition of training data by automating the screenshot capture process. This ensured consistency, accuracy, and

efficiency in collecting the visual inputs necessary for analysis, while reducing the likelihood of human error.

VII. CONCLUSION

This work proposed an application, called WillRo, capable of analyzing website screenshots to determine the presence of malicious content, thereby providing users the ability to detect phishing.

The system achieved a precision of 85.80% and a recall of 93.00% demonstrating strong performance. Moreover, the user-friendly interface and the automated analysis not only make it possible for technically unskilled users to use it, but they also make it suitable for non-technical users in everyday web browsing activities.

However, the performance evaluation metrics pointed out that the model needs optimization, particularly in the size and variability of the dataset. Future improvements should focus on employing larger and more diverse datasets from verified sources, which would enhance the model's generalization capacity. Additionally, monitoring the system's performance over time - by recording behaviors and outcomes during analysis - would provide valuable insights for iterative refinement. Working on these aspects will increase the reliability of detection and wider acceptance in educational and corporate environments.

ACKNOWLEDGMENT

The authors are grateful to the Dirección de Investigación of the Universidad Peruana de Ciencias Aplicadas for the support provided for this research work through the UPC-EXPOST-2025-2 incentive.

REFERENCES

[1] "2024 Data Breach Investigations Report," Verison Business, Ashburn, VR, USA, 2024, https://www.verizon.com/business/resources/reports/2024-dbir-data-breach-investigations-report.pdf.

[2] J. Rios. "Student emails are being used to spread fake news." Infobae, https://www.infobae.com/tecno/2023/10/21/correos-electronicos-de-los-estudiantes-son-usados-para-propagar-mensajes-falsos/.

[3] A. H. Aljammal, S. Taamneh, A. Qawasmeh, and H. B. Salameh, "Machine Learning Based Phishing Attacks Detection Using Multiple Datasets.," *International Journal of Interactive Mobile Technologies*, vol. 17, no. 5, Mar. 2023, Art. no. 71, https://doi.org/10.3991/ijim.v17i05.37575.

[4] U. A. Butt, R. Amin, H. Aldabbas, S. Mohan, B. Alouffi, and A. Ahmadian, "Cloud-based email phishing attack using machine and deep learning algorithm," *Complex & Intelligent Systems*, vol. 9, no. 3, pp. 3043–3070, 2023, https://doi.org/10.1007/s40747-022-00760-3.

[5] A. K. Jain, N. Debnath, and A. K. Jain, "APuML: An Efficient Approach to Detect Mobile Phishing Webpages using Machine Learning," *Wireless Personal Communications*, vol. 125, no. 4, pp. 3227–3248, May 2022, https://doi.org/10.1007/s11277-022-09707-w.

[6] S. Das Guptta, K. T. Shahriar, H. Alqahtani, D. Alsalman, and I. H. Sarker, "Modeling Hybrid Feature-Based Phishing Websites Detection Using Machine Learning Techniques," *Annals of Data Science*, vol. 11, no. 1, pp. 217–242, Feb. 2024, https://doi.org/10.1007/s40745-022-00379-8.

[7] A. Ozcan, C. Catal, E. Donmez, and B. Senturk, "A hybrid DNN–LSTM model for detecting phishing URLs," *Neural Computing and Applications*, vol. 35, no. 7, pp. 4957–4973, 2023, https://doi.org/10.1007/s00521-021-06401-z.

[8] N. Nagy *et al.*, "Phishing URLs Detection Using Sequential and Parallel ML Techniques: Comparative Analysis," *Sensors*, vol. 23, no. 7, Mar. 2023, Art. no. 3467, https://doi.org/10.3390/s23073467.

[9] S.-S. Shin, S.-G. Ji, and S.-S. Hong, "A Heterogeneous Machine Learning Ensemble Framework for Malicious Webpage Detection," *Applied Sciences*, vol. 12, no. 23, Nov. 2022, Art. no. 12070, https://doi.org/10.3390/app122312070.

[10] D. M. Linh, H. D. Hung, H. M. Chau, Q. S. Vu, and T.-N. Tran, "Real-time phishing detection using deep learning methods by extensions," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 14, no. 3, pp. 3021–3035, June 2024.

[11] O. K. Sahingoz, E. BUBEr, and E. Kugu, "DEPHIDES: Deep Learning Based Phishing Detection System," *IEEE Access*, vol. 12, pp. 8052–8070, 2024, https://doi.org/10.1109/ACCESS.2024.3352629.

[12] M. Elsadig *et al.*, "Intelligent Deep Machine Learning Cyber Phishing URL Detection Based on BERT Features Extraction," *Electronics*, vol. 11, no. 22, Nov. 2022, Art. no. 3647, https://doi.org/10.3390/electronics11223647.

[13] M. Sam'an and K. Imaddudin, "Hybrid deep learning model for YouTube spam comment detection," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 14, no. 3, pp. 3313–3319, June 2024.

[14] S. Aslam, H. Aslam, A. Manzoor, H. Chen, and A. Rasool, "AntiPhishStack: LSTM-Based Stacked Generalization Model for Optimized Phishing URL Detection," *Symmetry*, vol. 16, no. 2, Feb. 2024, Art. no. 248, https://doi.org/10.3390/sym16020248.

[15] S. A. Mohamed, M. A. Mahmoud, M. N. Mahdi, and S. A. Mostafa, "Improving Efficiency and Effectiveness of Robotic Process Automation in Human Resource Management," *Sustainability*, vol. 14, no. 7, Mar. 2022, Art. no. 3920, https://doi.org/10.3390/su14073920.

[16] B. Vajgel *et al.*, "Development of Intelligent Robotic Process Automation: A Utility Case Study in Brazil," *IEEE Access*, vol. 9, pp. 71222–71235, 2021, https://doi.org/10.1109/ACCESS.2021.3075693.

[17] M. Yao, "RPA Technology Enables Highly Automated Development of Corporate Financial Accounting Processes," *Applied Mathematics and Nonlinear Sciences*, vol. 9, no. 1, 2024, Art. no. 20240541.

[18] S. Cho, J. Moon, J. Bae, J. Kang, and S. Lee, "A Framework for Understanding Unstructured Financial Documents Using RPA and Multimodal Approach," *Electronics*, vol. 12, no. 4, Feb. 2023, Art. no. 939, https://doi.org/10.3390/electronics12040939.

[19] D. Choi, H. R'bigui, and C. Cho, "Candidate Digital Tasks Selection Methodology for Automation with Robotic Process Automation," *Sustainability*, vol. 13, no. 16, Aug. 2021, Art. no. 8980, https://doi.org/10.3390/su13168980.

[20] I. Skula and M. Kvet, "A Framework for Preparing a Balanced and Comprehensive Phishing Dataset," *IEEE Access*, vol. 12, pp. 53610–53622, Apr. 2024, https://doi.org/10.1109/ACCESS.2024.3387437.

[21] A. A. Albishri and M. M. Dessouky, "A Comparative Analysis of Machine Learning Techniques for URL Phishing Detection," *Engineering, Technology & Applied Science Research*, vol. 14, no. 6, pp. 18495–18501, Dec. 2024, https://doi.org/10.48084/etasr.8920.

[22] L. S. H. Colin, P. M. Mohan, J. Pan, and P. L. K. Keong, "An Integrated Smart Contract Vulnerability Detection Tool Using Multi-Layer Perceptron on Real-Time Solidity Smart Contracts," *IEEE Access*, vol. 12, pp. 23549–23567, Feb. 2024, https://doi.org/10.1109/ACCESS.2024.3364351.

[23] W. Wang, J. Song, G. Xu, Y. Li, H. Wang, and C. Su, "ContractWard: Automated Vulnerability Detection Models for Ethereum Smart Contracts," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 2, pp. 1133–1144, 2021, https://doi.org/10.1109/TNSE.2020.2968505.

[24] Y. A. Alsariera, M. H. Alanazi, Y. Said, and F. Allan, "An Investigation of AI-Based Ensemble Methods for the Detection of Phishing Attacks," *Engineering, Technology & Applied Science Research*, vol. 14, no. 3, pp. 14266–14274, June 2024, https://doi.org/10.48084/etasr.7267.

[25] *Kaggle*, https://www.kaggle.com/.

[26] *Phish-Iris Dataset*, https://www.kaggle.com/datasets/saurabhshahane/phishiris.

[27] *Counterfeit Affiliate Programs (CAP)*, https://www.kaggle.com/datasets/claudiocarpineto/counterfeit-affiliate-programs.

[28] *Phishing dataset*, https://www.kaggle.com/datasets/fajaritafebriani/phishing-dataset.

[29] "Train YOLOv5 on custom data." Ultralytics, https://docs.ultralytics.com/es/yolov5/tutorials/train_custom_data/.

[30] "Performance Metrics Deep Dive." Ultralytics, https://docs.ultralytics.com/guides/yolo-performance-metrics/.

AUTHORS PROFILE

Alvaro Roque is a Systems Information Engineering student at the Peruvian University of Applied Sciences in Lima, Peru, who focuses on the analysis and development of solutions to business problems, information security, and project management (email: u202013176@upc.edu.pe).

Willy Sotelo is a Systems Information Engineering student at the Peruvian University of Applied Sciences in Lima, Peru, who focuses on the analysis of data and development of digital solutions for business problems (email: u202015236@upc.edu.pe).

Sandra Wong-Durand has a master's degree in Artificial Intelligence, a master's degree in Business Administration from ESAN with mention in Advanced Project Management, Systems Engineer from UNIFE, with specialization studies in Innovation and Leadership at the Escuela Superior de Administración y Dirección de Empresas (ESADE) – Spain. (email: pcsiswon@upc.edu.pe, ORCID: https://orcid.org/0000-0002-6154-2124.

Pedro Castaneda is a RENACYT Researcher and holds a PhD in Systems Engineering, a master's degree in management and information technology management from UNMSM, and a master's degree in business administration (MBA) - ESAN. (email: pedro.castaneda@untrm.edu.pe, ORCID: https://orcid.org/0000-0003-1865-1293).

Alejandra Onate-Andino. She holds a PhD in Systems Engineering and Computer Science from Universidad Mayor de San Marcos (Peru). She is the author of several scientific articles in the area of Information Technology Governance, Business Intelligence, and Information Technology Management, among others. (email: monate@espoch.edu.ec).