International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 14, No.1 (Sep-2023)

http://dx.doi.org/10.12785/ijcds/140179

PROCTOR: A Robust URL Protection System Against Fraudulent, Phishing, and Scam Activities

Abdul Azzam Ajhari[1], Dimas Febriyan Priambodo[2], Radifa Hilya Paradisa[3] and Henny Yulianti[4]

[1,4]*Department of Informatics, Universitas Siber Asia, Jakarta, 12550, Indonesia*
[2]*Cyber Security Engineering,National Cyber and Crypto Polytechnic, Bogor, 16120, Indonesia*
[3]*National Cyber and Crypto Agency, Depok, 16511, Indonesia*

Received 16 Jul. 2023, Revised 15 Sep. 2023, Accepted 18 Sep. 2023, Published 20 Sep. 2023

Abstract: Changes in internet usage patterns and behavior that have become increasingly massive since the COVID-19 pandemic have made hackers have various cybercrime ways to trick their victims. Some of the methods that are still used by hackers are fraud by utilizing user data with fake websites (phishing) that resemble the original website. The appearance and URL of the website that deceives the target or potential victim is a scam trick to gain the trust of the target. Therefore, we decided to research by building a URL detection system with the characteristics of fraud, phishing, and scam website-based using machine learning. Because this system is preventive in the form of protection, a user-friendly name was created, namely Protective URL Detector (PROCTOR). PROCTOR uses 52 standard features of website security protocols and is trained to leverage fraud, phishing, and scam data in Indonesia with random forest (RF) machine learning models. After training, the model is tested and evaluated with new data using the confusion matrix classification evaluation method. The most optimal model is achieved by the RF model with a training accuracy of 99.91

Keywords: Cybercrime, Machine Learning, Phishing, Random Forest, Scam

1. INTRODUCTION

In the phishing activity trend report, global phishing cases continued to increase during 2021 [1]. Indonesia Anti-Phishing Data Exchange [2] reports that at least 3,180 cases of phishing occurred in Indonesia in the first quarter of 2022. Hackers trick their victims to access data with short-lived, evasive, malicious URLs, usually hiding behind an automatically running redirect network. So that users as potential victims believe that the appearance of this website is safe (scam). The data is usually obtained by hackers in the form of financial data, personal information, usernames, and passwords which will then be used for illegal things (fraud). Indonesia is the biggest target for hackers because internet users in Indonesia in January 2022 reached 204.7 million or about 73.7% of the total population of Indonesia [3].

Therefore, we need a method to detect unsafe URLs (fraud, phishing, and scam) that can prevent and reduce the risk for users. There are several methods used, whitelist or blacklist by [4], [5], [6], Deep Learning with Deep Neural Network (DNN), Long Short-Term Memory (LSTM), and Convolution Neural Network (CNN) used in research [7]. CNN is also used by research [8], [9] and besides that, there is a Gated Recurrent Neural Network used in research [10]. In addition to whitelisting, blacklisting, and deep learning methods, current developments also lead to machine learning (ML), especially using URL-based, including research from [11] and using the random forest (RF) method [12] and more in-depth using feature extraction on URLs by [13] and additional 30 feature extraction by [14].

With so many ways to detect URLs, this research tries to use the point of view of an artificial intelligence (AI) system as a protective url detector (PROCTOR). Based also on the comparison of [15], [16] so to bring together the concept of AI with more comprehensive features this research was conducted. ML as a branch of AI has developed rapidly and has been used in various fields, including cybersecurity. Classification is one of the supervised machine learning techniques that identify classes by learning features in a dataset. It can be applied to classify the types of malicious or legitimate websites [17].

In a previous study [18] phishing detection of emails has also been using the K-Nearest Neighbor (KNN) algorithm model. This model can effectively distinguish normal, spam, and phishing emails, and has the highest accuracy of up to 95.27%. Its research was able to beat other studies [19] which only yielded an accuracy of 85.08% with the optimal number of neighbors being 10.

E-mail: abdulazzam@lecturer.unsia.ac.id, dimas.febriyan@poltekssn.ac.id, radifa.hilya@bssn.go.id

In the research conducted by [20], a combination of Support Vector Machine (SVM) and Logistic Regression (LR) was carried out using Lexical features based on predefined malicious URL detection. The model was able to work better with the highest accuracy rate reaching 98%. A similar study conducted by [21] proposed the use of Random Forest (RF) with 30 parameters which resulted in the greatest accuracy of 99.36% and is still more robust compared to existing research algorithms beating the combined RF study with BLSTM [14]. Thus, the results of this study make an important contribution to the development of more accurate and reliable malicious URL detection methods.

In this study, several algorithm models used were evaluated to detect unsafe website URLs. The purpose of this evaluation is to obtain an algorithm with optimal performance. This research compares the KNN, SVM, and RF [21] models that have been used in previous research with the proposed RF algorithm model plus the addition of 52 additional features or parameters. The addition of these features or parameters is expected to improve the ability to detect unsafe websites. By comparing the performance results of these models, this research aims to identify the most effective and accurate algorithm model for detecting malicious website URLs.

2. DATA

A. Data acquisition

Process of obtaining data to improve the accuracy of machine learning models. In this study, the data used consisted of a website address (URL) and a label stating whether the URL was valid (good) or unsafe (bad). The dataset is collected from several sources as follows.

1) Dataset of valid Indonesian government website URL links collected by the National Cyber and Crypto Agency (BSSN) in 2020;
2) Dataset valid URL link for payment system licensing and money management obtained from Bank Indonesia on October 15, 2021;
3) Dataset valid URL website of Electronic System Operator (PSE) obtained from the Directorate of Aptika Governance, Ministry of Communication and Information (Menkominfo) of the Republic of Indonesia on October 15, 2021;
4) Data collection indicated safe and unsafe obtained from SMS/Email of each research team;
5) Test data set jointly conducted by the community in December 2021.

The total dataset is 45,987 data, consisting of 9,325 good, 14,955 bad, and 21,667 malformed. The dataset labeled malformed is ignored because the website URL is no longer active, so the data used is 24,320 datasets with 9,325 labeled good and 14,995 labeled bad.

In machine learning, 2 data are needed to be used as training data and test data based on several references for dividing a dataset by [22], [23], [24], this study has similarities such as [21] uses the proportion of data split 80:20. The total dataset of 24,320 is then separated into two parts, for training data with a proportion of 80% or around 19,460 data. testing data using a proportion of 20% for the evaluation of the test or about 4,865 data.

B. Preprocessing

The process of getting data ready for a machine learning model to analyze. This paper is divided into three subprocesses, such as data cleaning by equating its characteristics and format using domain and top-level domain (TLD), feature extraction, and normalization.

Typically, website URLs generally consist of several website security protocols such as the use of SSL certificates and certificate authorities. These features may not reveal the legitimacy of a website, but combining multiple features increases the likelihood of detecting potentially malicious website URLs. There are at least 52 experimental parameters or features used in this study in Table I, with some previous parameters from malicious URL detection research [14], [20] and phishing site inspection [25], [26], [27].

The normalization standardization technique used is the Z-score normalization (ZN) technique, a simple transforming operation at the feature level, can offer an effective solution [28]. This results in a normally distributed feature with a mean value of 0 and a standard deviation of 1. The normalization formula is written mathematically in Formula 1, where Z is the normalized data value, X is the initial data value, ϕ the initial value of the feature average value, and θ the initial standard deviation of features.

$$Z = \frac{X - \phi}{\theta} \tag{1}$$

C. Classification techniques

1) K-Nearest Neighbors (KNN)
KNN is one of the machine learning algorithms used for classification and regression. The algorithm works by comparing new data with existing data in the training set. To find the close or far distance between points in class k is usually calculated using the Euclidean distance. In addition, choosing the right K value is very important as it affects the classification performance. This can be done by calculating the distance between the test set q and all training sets p, where n is the number of features. The distances are then sorted in ascending order to identify features based on K data and classify new data. The formula for calculating the Euclidean distance between two points in two dimensions is as Formula 2.

$$d(x_1, x_2) = \sqrt{\sum_{i=1}^{n} (x_1 - y_1)^2} \tag{2}$$

TABLE I. Additional 52 Feature

No	Features Name	Description	Pseudo Code
1	Protocol	“http” or “https”	Feature 1-0: if $\left(\frac{\text{Shorten link}\in\{\text{find deep link}\}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$ Feature 1-1: if $\left(\frac{\text{HTTPS link}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$ Feature 1-2: if $\left(\frac{\text{SSL valid}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
2	Top-level domain (TLD)	the last part of the domain name such as “.com”, “co. id”, …	Feature 2: if $\left(\frac{\text{TLD valid}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
3	Len_URL_full	the length of the URL	Feature 3: if $\left(\frac{\text{(Length URL<50)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
4	Len_Alphabet_full	the no. of alphabet characters in the URL	Feature 4: if $\left(\frac{\text{(Alphabet URL>10)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
5	Len_Non-alphabet_full	the no. of non-alphabet characters in the URL	Feature 5: if $\left(\frac{\text{(Non-Alphabet URL<3)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
6	Len_Spec-character_full	the no. of special characters in the URL	Feature 6: if $\left(\frac{\text{(Special URL=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
7	Count_At_full	the no. of “@” in the URL	Feature 7: if $\left(\frac{\text{(At URL=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
8	Count_Dot_full	the no. of “.” in the URL	Feature 8: if $\left(\frac{\text{(Dot URL<3)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
9	Count_Dash_full	the no. of “-” in the URL	Feature 9: if $\left(\frac{\text{(Dash URL<3)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
10	Count_UnderScore_full	the no. of “_” in the URL	Feature 10: if $\left(\frac{\text{(Underscore URL=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
11	Count_Slash_full	the no. of “/” in the URL	Feature 11: if $\left(\frac{\text{(Slash URL<5)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
12	Count_Question_Mark_full	the no. of “?” in the URL	Feature 12: if $\left(\frac{\text{(QMark URL=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
13	Count_Equal_full	the no. of “=” in URL	Feature 13: if $\left(\frac{\text{(Equal URL=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
14	Count_Ampersand_full	the no. of “&” in the URL	Feature 14: if $\left(\frac{\text{(Ampers URL=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
15	Count_Comma_full	the no. of “,” in the URL	Feature 15: if $\left(\frac{\text{(Comma URL=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
16	Count_Asterisk_full	the no. of “*” in the URL	Feature 16: if $\left(\frac{\text{(Asterisk URL=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
17	Count_Hastag_full	the no. of “#” in the URL	Feature 17: if $\left(\frac{\text{(Hashtag URL=1)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
18	Count_Semicolon_full	the no. of “;” in the URL	Feature 18: if $\left(\frac{\text{(Semicolon URL=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
19	Len_domain	the length of the domain	Feature 19: if $\left(\frac{\text{(Length domain<20)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
20	Len_Alphabet_full	the no. of alphabet characters in the domain	Feature 20: if $\left(\frac{\text{(Alphabet domain>4)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
21	Len_Non-alphabet_domain	the no. of non-alphabet characters in the domain	Feature 21: if $\left(\frac{\text{(Non-Alphabet domain<3)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
22	Len_Spec-character_domain	the no. of special character in the domain	Feature 22: if $\left(\frac{\text{(Special domain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
23	Count_At_domain	the no. of “@” in the domain	Feature 23: if $\left(\frac{\text{(At domain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
24	Count_Dot_domain	the no. of “.” in the domain	Feature 24: if $\left(\frac{\text{(Dot domain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
25	Count_Dash_domain	the no. of “-” in the domain	Feature 25: if $\left(\frac{\text{(Dash domain<3)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
26	Count_UnderScore_domain	the no. of “_” in the domain	Feature 26: if $\left(\frac{\text{(Underscore domain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
27	Count_Slash_domain	the no. of “/” in the domain	Feature 27: if $\left(\frac{\text{(Slash domain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
28	Count_Question_Mark_domain	the no. of “?” in the domain	Feature 28: if $\left(\frac{\text{(QMark domain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
29	Count_Equal_domain	the no. of “=” in domain	Feature 29: if $\left(\frac{\text{(Equal domain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
30	Count_Ampersand_domain	the no. of “&” in the domain	Feature 30: if $\left(\frac{\text{(Ampers domain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
31	Count_Comma_domain	the no. of “,” in the domain	Feature 31: if $\left(\frac{\text{(Comma domain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
32	Count_Asterisk_domain	the no. of “*” in the domain	Feature 32: if $\left(\frac{\text{(Asterisk domain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
33	Count_Hastag_domain	the no. of “#” in the domain	Feature 33: if $\left(\frac{\text{(Hashtag domain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
34	Count_Semicolon_domain	the no. of “;” in the domain	Feature 34: if $\left(\frac{\text{(Semicolon domain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
35	Len_subdomain	the length of the subdomain	Feature 35: if $\left(\frac{\text{(Length subdomain<15)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
36	Len_Alphabet_subdomain	the no. of alphabet characters in the subdomain	Feature 36: if $\left(\frac{\text{(Alphabet subdomain>4)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
37	Len_Non-alphabet_subdomain	the no. of non-alphabet characters in the subdomain	Feature 37: if $\left(\frac{\text{(Non-Alphabet subdomain<2)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
38	Len_Spec-character_subdomain	the no. of special character in the subdomain	Feature 38: if $\left(\frac{\text{(Special subdomain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
39	Count_At_subdomain	the no. of “@” in the subdomain	Feature 39: if $\left(\frac{\text{(At subdomain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
40	Count_Dot_subdomain	the no. of “.” in the subdomain	Feature 40: if $\left(\frac{\text{(Dot subdomain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
41	Count_Dash_subdomain	the no. of “-” in the subdomain	Feature 41: if $\left(\frac{\text{(Dash subdomain<2)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
42	Count_UnderScore_subdomain	the no. of “_” in the subdomain	Feature 42: if $\left(\frac{\text{(Underscore subdomain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
43	Count_Slash_subdomain	the no. of “/” in the subdomain	Feature 43: if $\left(\frac{\text{(Slash subdomain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
44	Count_Question_Mark_subdomain	the no. of “?” in the subdomain	Feature 44: if $\left(\frac{\text{(QMark subdomain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
45	Count_Equal_subdomain	the no. of “=” in subdomain	Feature 45: if $\left(\frac{\text{(Equal subdomain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
46	Count_Ampersand_subdomain	the no. of “&” in the subdomain	Feature 46: if $\left(\frac{\text{(Ampers subdomain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
47	Count_Comma_subdomain	the no. of “,” in the subdomain	Feature 47: if $\left(\frac{\text{(Comma subdomain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
48	Count_Asterisk_subdomain	the no. of “*” in the subdomain	Feature 48: if $\left(\frac{\text{(Asterisk subdomain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
49	Count_Hastag_subdomain	the no. of “#” in the subdomain	Feature 49: if $\left(\frac{\text{(Hashtag subdomain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
50	Count_Semicolon_subdomain	the no. of “;” in the subdomain	Feature 50: if $\left(\frac{\text{(Semicolon subdomain=0)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
51	ratio_url_path	the ratio of URL and path	Feature 51: if $\left(\frac{\text{(ratio<3)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$
52	ratio_digit_url	the ratio of digits in URL and URL	Feature 52: if $\left(\frac{\text{(ratio<0.5)}\rightarrow\text{legitimate}}{\text{Otherwise}\rightarrow\text{suspicious}}\right)$

2) Support Vector Machine (SVM) SVM is a machine learning algorithm used for classification and regression [29]. It works by finding the best line or hyperplane that can separate two classes of data. Suppose the data set, D, is given as $\{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$, where x_i is the set of training tuples with corresponding classes labeled y_i. Each y_i can take one of two values, either $+1$ or -1, corresponding to the class 'malicious website' or 'legitimate website,' respectively. The SVM finds the best decision to separate these two classes by using a hyperplane, h, [30] which can be defined as Formula 3.

$$h(x) = W \cdot X + b = \sum_{i=1}^{N} \alpha_i y_i (x_i, x) + b \qquad (3)$$

Where W is the weight, b is the bias, N is the number of features in the dataset, x_i is the set of training tuples, and α_i is the Lagrange multiplier. When dealing with non-linear data, we can apply the RBF (Radial Basis Function) kernel to transform the data into a higher-dimensional space. This allows us to use a linear model to separate the transformed data.

3) Support Vector Machine (SVM) Random Forest is a machine learning algorithm used to classify or regress large data sets. Because of its functionality, it can be used for many dimensions with various scales and high performance. This classification is done by merging trees in a decision tree by training the dataset. Suppose RF(x) is the result of Random Forest prediction for data x, where $\{P_1(x), P_2(x), \ldots, P_n(x)\}$ are the predictions of each decision tree the Equation as Formula 4.

$$RF(x) = \text{mode}(P_1(x), P_2(x), \ldots, P_n(x)) \qquad (4)$$

3. MODELLING

Since some classifiers cannot be trained on categorical data, the dataset needed to be transformed and converted through pre-processing, where all nominal values were converted to numerical values. The same conversion model was used to map nominal data to numerical data across the dataset. In addition, the dataset went through a feature scaling process to make the data normally distributed with zero as the mean and standard deviation of 1. This process can reduce the processing time for some classifiers as well as avoid any mismatch issues that may arise. The performance evaluation of the algorithm is done using several metrics: confusion matrix and mean absolute error (MAE).

The confusion matrix is a two-dimensional matrix with rows indicating actual URL label values and columns indicating predicted URL label values. The confusion matrix is used to evaluate the performance of the classification model by comparing the model's predictions with the actual reality. In the confusion matrix, TP, FP, FN, and TN are used to calculate evaluation metrics such as accuracy, precision, sensitivity, and specificity [31]. The following steps are to formulate a confusion matrix in Table II for accuracy prediction in a standard way using only Positives and Negatives [32], [33] shown in Formula (5), (6), (7), and (8).

TABLE II. Confusion Matrix

		Predicted	
		Yes	No
Actual	Yes	True Positive (TP)	False Negative (FN)
	No	False Positive (FP)	True Negative (TN)

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \qquad (5)$$

$$Precision = \frac{TP}{TP + FP} \qquad (6)$$

$$Recall = \frac{TP}{TP + FN} \qquad (7)$$

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} \qquad (8)$$

In addition, the calculation of the mean absolute error (MAE) in equation (9) is used to compare the performance of several models to measure the accuracy of a model evaluation [34] in making predictions. MAE calculates the absolute difference between each pair of predicted and true values and then takes the average of all the differences. MAE gives an idea of the extent to which the model predictions deviate from the true values. The lower the MAE value, the smaller the prediction error, which indicates better performance. The most suitable and optimal model in evaluation is compiled into the pickle module (.pkl) to be applied at the deployment stage.

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i| \qquad (9)$$

In comparing the performance of RF with 52 additional parameters or features, we conducted several experiments with other algorithms, namely KNN and SVM. Three models were trained using the best hyperparameters and 80% data proportion in Table I. The accuracy of the proposed RF model shows better performance than the other algorithms, resulting in an optimal model result with a training accuracy of 99.91% and an MAE result of 0.0093 as seen in Table III. Thus, it can be concluded that RF is an effective choice in predicting outcomes with a high degree of accuracy.

The confusion matrix test is carried out after the training process from 20% of the data that is not used in the training and is presented in Table IV, Table V, and Table VI.

TABLE III. Train Evaluation Result

No	Models	Hyperparameter	Train accuracy	MAE
1	KNN	n-neighbors=4	99.19%	0.0081
2	SVM	kernel=rbf, degree=0.3	99.03%	0.0097
3	RF	n_tree=50	99.91%	0.0093

Table IV with the KNN model, explains that predictions that are misclassified should be bad into good is 25 data, while classifications that should be good into bad are 17 data. Then Table 5 with the SVM model, explains that the predictions that are misclassified should be bad into good are 29 data, while the classification that should be good into bad is 16 data. In Table 6 with the RF model 52 parameters or features explain that the predictions that are misclassified should be bad into good are 17 data, while for the classification that should be good into bad as much as 13 data. This model shows that RF is more optimal than other models.

TABLE IV. KNN Testing Evaluation Result

		Predicted Label	
		Bad	Good
True Label	Bad	2982	17
	Good	25	1840

TABLE V. SVM Testing Evaluation Result

		Predicted Label	
		Bad	Good
True Label	Bad	2983	16
	Good	29	1836

TABLE VI. RF Testing Evaluation Result

		Predicted Label	
		Bad	Good
True Label	Bad	2986	13
	Good	17	1848

Table VII presents the evaluation of the test performance on the confusion matrix and MAE. The RF model has the best performance among the other two models with a 99.38% accuracy and the smallest MAE of 0.0062. So, the RF model is compiled into a pickle module to be implemented at the model implementation stage. The proposed model has a slightly better improvement as shown in Table VIII with the addition of 52 features.

4. DEPLOYMENT

The best model, RF was then implemented using Python version 3.10.6 on a Docker container with a Flask API

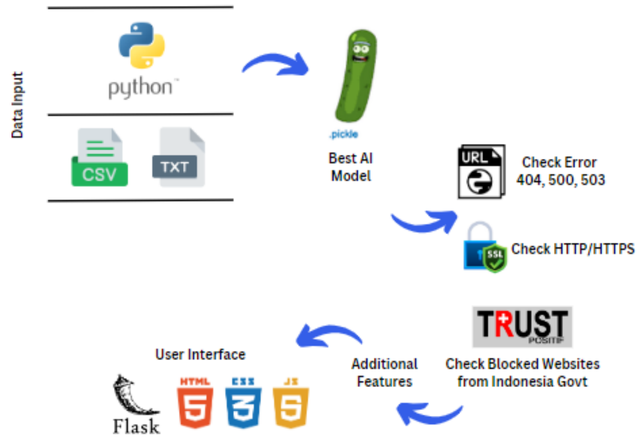


Figure 1. Deployment Method Process

web service as the user/client system website interface. Docker is a technology that combines applications, related dependencies, and organized system libraries to be built in containers. Flask is a micro web framework written in Python and based on the WSGI toolkit and Jinja 2 template engine that allows developers to build web applications quickly [35] as you can see in Figure 1. This combination is perfect for the stage of developing AI models for a web application as a PROtective url deteCTOR (PROCTOR).

At this stage, the system uses the Flask method process and several additional features. Before implementing the best AI model into the user interface, we need to convert the model into a library named pickle (.pkl). The additional features describe as follows.

A. Algorithm

There are several algorithms in the program created. Several algorithms are made to get plain URLs so that they can be analyzed further. Algorithm 1 is intended to process URLs with the https prefix and algorithm 2 is intended for URL processing with the http format. Algorithm 3 is intended to perform URL checks as in additional feature 1 as the initial stage of checking. Subsequent checks use more url plans so a parsing algorithm is needed as in algorithm 4. The next feature up to 52 features can be exemplified in algorithm 5. Algorithm 6 is a new feature for connecting TrustPositive "internet positif" from The Indonesia Ministry of Communication and Informatics. This algorithm also gave validation this application suitable for Indonesian.

B. Scenario

1) The user receives a short message in this case in Bahasa or Indonesian language via social media or email accompanied by a website URL as shown in Figure 2. No need to open a suspicious URL, the user just copies the website URL https://bit.ly/30BXEOu.
2) Paste the previously copied link into the input box in the interface PROCTOR website system as shown in

TABLE VII. The Best Testing Evaluation Result

No	Models Name	Overall Accuracy	Precision	Recall	F1-score	MAE
1	KNN	99.14%	Bad = 99.17% Good = 99.08%	Bad = 99.43% Good = 98.66%	Bad = 99.30% Good = 98.87%	0.0086
2	SVM	99.07%	Bad = 99.04% Good = 99.14%	Bad = 99.47% Good = 98.45%	Bad = 99.25% Good = 98.79%	0.0093
3	RF	99.38%	Bad = 99.43% Good = 99.30%	Bad = 99.57% Good = 99.09%	Bad = 99.50% Good = 99.19%	0.0062

Algorithm 1: Check Error

```
def check_url(url):
ERROR_URL = f'https://url'
try:
ERROR_URL = urlparse(ERROR_URL)
connection = HTTPConnection(ERROR_URL.netloc, timeout=2)
connection.request('HEAD', ERROR_URL.path)
if connection.getresponse():
return True
else:
return False
except:
return False
```

Algorithm 2: check_http_url

```
def check_http_url(url):
HTTP_URL = f'http://url'
try:
HTTP_URL = urlparse(HTTP_URL)
connection = HTTPConnection(HTTP_URL.netloc)
connection.request('HEAD', HTTP_URL.path)
if connection.getresponse():
return True
else:
return False
except:
return False
```

Algorithm 3: check_https_url

```
def check_https_url(url):
HTTPS_URL = f'https://url'
try:
HTTPS_URL = urlparse(HTTPS_URL)
connection = HTTPSConnection(HTTPS_URL.netloc)
connection.request('HEAD', HTTPS_URL.path)
if connection.getresponse():
return True
else:
return False
except:
return False
```

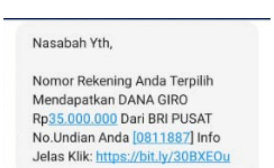


Figure 2. Deployment Method Process

TABLE VIII. Research Accuracy Comparison

Research	Models	Accuracy
Previous	RF with additional 30 features [14]	99.38%
Proposed	RF with additional 52 features	99.36%

Algorithm 4: base_url

```
def base_url(url, with_path=False):
parsed = urlparse(url)
path = '/'.join(parsed.path.split('/')[:-1]) if with_path else ''
parsed = parsed._replace(path=path)
parsed = parsed._replace(params='')
parsed = parsed._replace(query='')
parsed = parsed._replace(fragment='')
return parsed.geturl()
```

Figure 3 and press predict button to get the website URL classification result.

3) The PROCTOR system will analyze website URLs by checking HTTP error codes, SSL certificates, and sites blocked by the Indonesian government on TrustPositive. After being analyzed, the website URL will be processed by ML by studying its 52 characteristics.
4) The prediction and classification results will bring up two classifications, namely indicated safe

Algorithm 5: feature

```
def fitur(df):
sy = ['','.','-','_','/','?','=','&',',','%','hastag',';']
sl = ['','A-Za-z','0-9','.-_/?=&,%hastag;']
nm = ['At','Dot','Dash','UnderScore', 'Slash', 'Question', 'Equal',
 'Ampersand', 'Comma', 'Percent', 'Hastag', 'Semicolon']
ln = ['URL', 'Alphabet', 'Non-alphabet', 'Spec-character']
en = ['Website','domain','subdomain']
for j in range(len(en)):
for k in range(len(sl)):
if k==0:
df['Len_'+ln[k]+'_'+en[j]] = df[en[j]].str.len()
else:
df['Len_'+ln[k]+'_'+en[j]] = df[en[j]].str.count('['+sl[k]+']')
for i in range(len(sy)):
df['Count_'+nm[i]+'_'+en[j]] = df[en[j]].str.count('['+sy[i]+']')
prediksi.append(en.inverse_transform([np.argmax(y_pred)])[0])
print(prediksi)
run_program(['http://google.com'], 'model')
```

Algorithm 6: Trust Positive/Internet positif feature

```
for i in range(len(df_test))::
link.append(protocol_check((df_test['Website'])[i])[0])
#print(link)
protocol.append(protocol_check((df_test['Website'])[i])[1])
try:
link[i] = requests.get(link[i], timeout=10).url
deep_url = requests.get(link[i], timeout=10).raise_for_status()
#print(link[i])
print('1')
print(link[i])
if 'mercusuar' in requests.get(link[i], timeout=10).url:
nb = 'URL tidak ditemukan'
base.append(nb)
deep.append(link[i])
elif 'aduankonten' in requests.get(link[i], timeout=10).url or
  'internetpositif' in requests.get(link[i], timeout=10).url:
print('ii')
nb = 'AKSES KE SITUS INI DIBLOKIR OLEH PEMERINTAH
  INDONESIA'
base.append(nb)
deep.append((df_test['Website'])[i])
else:
#link[i] = requests.get(link[i], timeout=10).url
base.append(base_url(link[i]))
deep.append(link[i])
```

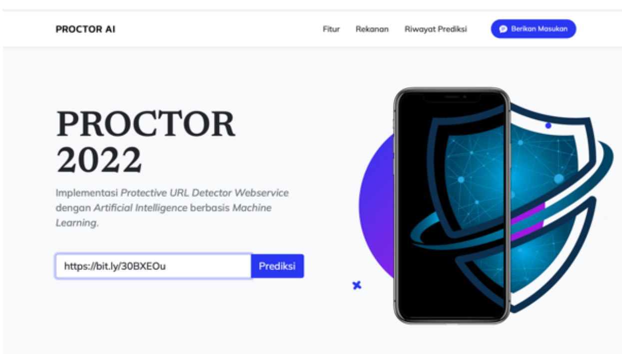


Figure 3. Interface PROCTOR website system

"terindikasi aman" and indicated unsafe "terindikasi tidak aman" with a probability accuracy. But, if the website URL is inactive, it will display the results of the site not being found "situs tidak ditemukan".

5) Another problem is the SSL certificate error, when the website is damaged in the SSL certificate like https://bit.ly/30BXEOu with the original URL https://programhadiah-bripoin.blogspot.com, the information "SSL Certificate Verification Error" will appear as shown in Figure 4 .

Different test scenarios were carried out on another short URL, bit.ly/shopeebigsale662, and displays the information as shown in Figure 5 . The classification also shows "terindikasi tidak aman" with the original URL, https://shopeebigsale662.blogspot.com, but with the addition of 99.98% prediction probability information.

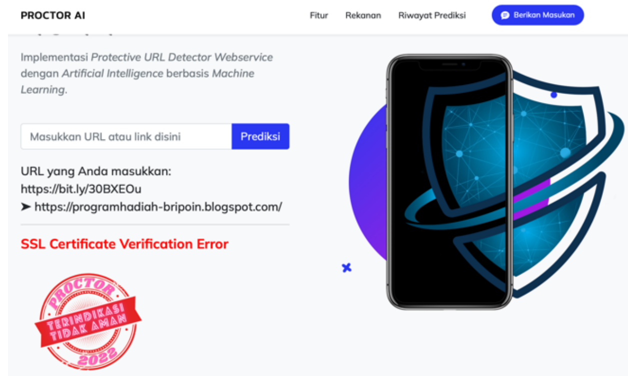


Figure 4. PROCTOR prediction and classification SSL Verification Error

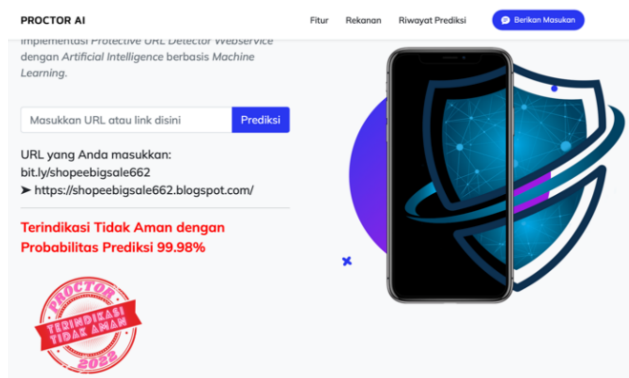


Figure 5. PROCTOR prediction and classification SSL Verification Error

5. Conclusions and Future Work

Three models are trained and tested on the dataset. A parametric study is performed for each model, and the best results are forwarded for evaluation. For KNN, high accuracy was given by the neighbor parameter corresponding to 4, resulting in a training accuracy of 99.19% and a testing accuracy of 99.14%. For SVM, high accuracy was reported with the radial basis function (RBF) kernel with 99.03% training accuracy and 99.07% testing accuracy. As for the proposed model, RF with 52 parameters, high accuracy was achieved resulting in a training accuracy of 99.91% and a testing accuracy of 99.38%.

Therefore, the proposed model enables fast and accurate detection of malicious website URL attacks. In addition, this system can be additional security from sites that are not registered on trustpositif / 'internet positif' owned by the Ministry of Communication and Informatics of the Republic of Indonesia. There are still many shortcomings in this study and it is hoped that improvements can be made in future research. Several improvements can be made by adding a Term Frequency Inverse Document Frequency (TF-IDF) or word vector to determine the frequency value of a word in a URL. Use of the time factor to analyze the efficiency and performance of an algorithm. The last thing that can be added is to use the latest model algorithm for processing.

References

[1] Anti-Phishing Working Group (APWG), "Phishing Activity Trends Report 4th Quarter 2021," Tech. Rep. February, 2022. [Online]. Available: https://apwg.org/trendsreports.

[2] Indonesian Anti-Phising Data Exchange, "Laporan Aktivitas Phising Domain .id," Indonesian Anti-Phising Data Exchange, Tech. Rep. 1, 2022.

[3] Global Digital Insights, "DataReportal – Global Digital Insights." [Online]. Available: https://datareportal.com/

[4] Y. Cao, W. Han, and Y. Le, "Anti-Phishing Based on Automated Individual White-List," in *Proceedings of the 4th ACM Workshop on Digital Identity Management*, ser. DIM '08. New York, NY, USA: Association for Computing Machinery, 2008, pp. 51–60.

[5] P. Prakash, M. Kumar, R. R. Kompella, and M. Gupta, "PhishNet: Predictive Blacklisting to Detect Phishing Attacks," in *2010 Proceedings IEEE INFOCOM*, 2010, pp. 1–5.

[6] A. K. Jain and B. B. Gupta, "A novel approach to protect against phishing attacks at client side using auto-updated white-list," *EURASIP Journal on Information Security*, vol. 2016, no. 1, p. 9, 2016. [Online]. Available: https://doi.org/10.1186/s13635-016-0034-3

[7] M. Somesha, A. R. Pais, R. S. Rao, and V. S. Rathour, "Efficient deep learning techniques for the detection of phishing websites," *Sādhanā*, vol. 45, no. 1, p. 165, 2020. [Online]. Available: https://doi.org/10.1007/s12046-020-01392-4

[8] A. Al-Alyan and S. Al-Ahmadi, "Robust URL phishing detection based on deep learning," *KSII Transactions on Internet and Information Systems*, vol. 14, no. 7, pp. 2752–2768, 2020.

[9] A. Aljofey, Q. Jiang, Q. Qu, M. Huang, and J. P. Niyigena, "An effective phishing detection model based on character level convolutional neural network from URL," *Electronics (Switzerland)*, vol. 9, no. 9, pp. 1–24, 2020.

[10] J. Zhao, N. Wang, Q. Ma, and Z. Cheng, "Classifying Malicious URLs Using Gated Recurrent Neural Networks BT - Innovative Mobile and Internet Services in Ubiquitous Computing," L. Barolli, F. Xhafa, N. Javaid, and T. Enokido, Eds. Cham: Springer International Publishing, 2019, pp. 385–394.

[11] H. Alaskar and T. Saba, "Machine Learning and Deep Learning: A Comparative Review BT - Proceedings of Integrated Intelligence Enable Networks and Computing," K. K. Singh Mer, V. B. Semwal, V. Bijalwan, and R. G. Crespo, Eds. Singapore: Springer Singapore, 2021, pp. 143–150.

[12] O. K. Sahingoz, E. Buber, O. Demir, and B. Diri, "Machine learning based phishing detection from URLs," *Expert Systems with Applications*, vol. 117, pp. 345–357, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417418306067

[13] R. S. Rao and A. R. Pais, "Detection of phishing websites using an efficient feature-based machine learning framework," *Neural Computing and Applications*, vol. 31, no. 8, pp. 3851–3873, 2019. [Online]. Available: https://doi.org/10.1007/s00521-017-3305-0

[14] M. G. Hr, A. Mv, S. Gunesh Prasad, and S. Vinay, "Development of anti-phishing browser based on random forest and rule of extraction framework," *Cybersecurity*, vol. 3, no. 1, pp. 1–14, 2020.

[15] N. Sharma, R. Sharma, and N. Jindal, "Machine Learning and Deep Learning Applications-A Vision," *Global Transitions Proceedings*, vol. 2, no. 1, pp. 24–28, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666285X21000042

[16] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning," *Electronic Markets*, vol. 31, no. 3, pp. 685–695, 2021. [Online]. Available: https://doi.org/10.1007/s12525-021-00475-2

[17] R. D. Prayogo and S. A. Karimah, "Optimization of Phishing Website Classification Based on Synthetic Minority Oversampling Technique and Feature Selection," in *2020 International Workshop on Big Data and Information Security (IWBIS)*, 2020, pp. 121–126.

[18] D. Xiao and M. Jiang, "Malicious Mail Filtering and Tracing System Based on KNN and Improved LSTM Algorithm," *Proceedings - IEEE 18th International Conference on Dependable, Autonomic and Secure Computing, IEEE 18th International Conference on Pervasive Intelligence and Computing, IEEE 6th International Conference on Cloud and Big Data Computing and IEEE 5th Cybe*, pp. 222–229, 2020.

[19] T. A. Assegie, "K-Nearest Neighbor Based URL Identification Model for Phishing Attack Detection," *Indian Journal of Artificial Intelligence and Neural Networking*, vol. 1, no. 2, pp. 18–21, 2021.

[20] A. Saleem Raja, R. Vinodini, and A. Kavitha, "Lexical features based malicious URL detection using machine learning techniques," *Materials Today: Proceedings*, vol. 47, no. xxxx, pp. 163–166, 2021. [Online]. Available: https://doi.org/10.1016/j.matpr.2021.04.041

[21] A. K. Jain and B. B. Gupta, *PHISH-SAFE: URL features-based phishing detection system using machine learning*. Springer Singapore, 2018, vol. 729.

[22] K. M. Kahloot and P. Ekler, "Algorithmic Splitting: A Method for Dataset Preparation," *IEEE Access*, vol. 9, pp. 125 229–125 237, 2021.

[23] V. R. Joseph, "Optimal ratio for data splitting," *Statistical Analysis and Data Mining*, vol. 15, no. 4, pp. 531–538, 2022.

[24] A. Nurhopipah and U. Hasanah, "Dataset Splitting Techniques Comparison For Face Classification on CCTV Images," *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*, vol. 14, no. 4, p. 341, 2020.

[25] R. S. Rao, T. Vaishnavi, and A. R. Pais, "CatchPhish: detection of phishing websites by inspecting URLs," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 2, pp. 813–825, 2020. [Online]. Available: https://doi.org/10.1007/s12652-019-01311-4

[26] K. L. Chiew, C. L. Tan, K. Wong, K. S. C. Yong, and W. K. Tiong, "A new hybrid ensemble feature selection framework for machine learning-based phishing detection system," *Information Sciences*, vol. 484, pp. 153–166, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025519300763

[27] M. Korkmaz, O. K. Sahingoz, and B. DIri, "Detection of Phishing Websites by Using Machine Learning-Based URL Analysis," *2020*

11th International Conference on Computing, Communication and Networking Technologies, ICCCNT 2020, 2020.

[28] N. Fei, Y. Gao, Z. Lu, and T. Xiang, "Z-Score Normalization, Hubness, and Few-Shot Learning," *Proceedings of the IEEE International Conference on Computer Vision*, pp. 142–151, 2021.

[29] Y. Arjoune, F. Salahdine, M. S. Islam, E. Ghribi, and N. Kaabouch, "A Novel Jamming Attacks Detection Approach Based on Machine Learning for Wireless Communication," in *2020 International Conference on Information Networking (ICOIN)*, 2020, pp. 459–464.

[30] F. Salahdine, Z. E. Mrabet, and N. Kaabouch, "Phishing Attacks Detection A Machine Learning-Based Approach," in *2021 IEEE 12th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, 2021, pp. 250–255.

[31] A. Luque, A. Carrasco, A. Martín, and A. de las Heras, "The impact of class imbalance in classification performance metrics based on the binary confusion matrix," *Pattern Recognition*, vol. 91, pp. 216–231, 2019. [Online]. Available: https://doi.org/10.1016/j.patcog.2019.02.023

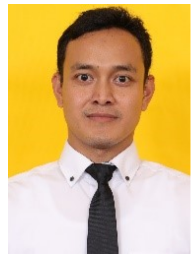
[32] G. Zeng, "On the confusion matrix in credit scoring and its analytical properties," *Communications in Statistics - Theory and Methods*, vol. 49, no. 9, pp. 2080–2093, 2020. [Online]. Available: https://doi.org/10.1080/03610926.2019.1568485

[33] D. Krstinić, M. Braović, L. Šerić, and D. Božić-Štulić, "Multi-label Classifier Performance Evaluation with Confusion Matrix," pp. 01–14, 2020.

[34] T. O. Hodson, "Root-mean-square error (RMSE) or mean absolute error (MAE): when to use them or not," *Geoscientific Model Development*, vol. 15, no. 14, pp. 5481–5487, 2022.

[35] A. M. Potdar, D. G. Narayan, S. Kengond, and M. M. Mulla, "Performance Evaluation of Docker Container and Virtual Machine," *Procedia Computer Science*, vol. 171, no. 2019, pp. 1419–1428, 2020. [Online]. Available: https://doi.org/10.1016/j.procs.2020.04.152

Abdul Azzam Ajhari is a Lecturer at Universitas Siber Asia (UNSIA) and an Informatics Expert at the National Cyber and Crypto Agency in the Republic of Indonesia. He received his master's degree from Bina Nusantara University majoring in computer science in 2022 with a thesis about Aircraft Flight Movement Anomaly Detection using Automatic Dependent Surveillance Broadcast. He has been certified ISO 27001 Lead Auditor Information Security Management Systems (ISMS). He also has certification from NVIDIA Deep Learning Institute. He can be contacted at email: abdulazzam@lecturer.unsia.ac.id.



Dimas Febriyan Priambodo is a lecturer at National Cyber and Crypto Polytechnic. He received his master's degree from Gajah Mada University majoring in computer science in 2018 with a thesis about resource utilization on multicore servers. His research concentrates mainly on operating systems, cybersecurity, electronics, and IT education. He is a Certified Ethical Hacker or CEH from EC Council. He is one of a joint research team in 5G security with the latest book "Tinjauan Strategis Keamanan 5G" as a reference for National Cyber and Crypto Agency. His research on the topic cross border payment receives a grant from the Central Bank of Indonesia. He can be contacted at email: dimas.febriyan@poltekssn.ac.id.

Radifa Hilya Paradisa is an employee of the National Cyber and Crypto Agency. Her educational background is a bachelor's and master's degree in mathematics from Universitas Indonesia. The basis of her research during college was in the fields of computational mathematics, data science, and artificial intelligence, especially machine learning or deep learning. She has international papers that have been published with more than 200 citations since 2021. Now her area of interest is starting to branch out into the areas of cybersecurity and cryptography. She can be contacted at email: radifa.hilya@bssn.go.id.

Henny Yulianti is the Head Department of the Informatics program and a Lecturer of Data Science at Universitas Siber Asia (UNSIA). She has two master's degrees from Budi Luhur University majoring in management in 2004 and computer science in 2022. She also has a lot of experience in computer science areas such as Data Science and Artificial Intelligence. She can be contacted at email: hennyyulia@lecturer.unsia.ac.id.