NSS
Neutrosophic Sets and Systems, Vol. 83, 2025
University of New Mexico

Efficient Machine Learning for Prediction of Malicious URLs under Neutrosophic Uncertainty Framework

Mohamed eassa1, Ahmed Abdelhafeez2, Ahmed A. Metwaly3, Ahmed S. Salama4

1,2 Computer Science Department, Faculty of Information Systems and Computer Science, October 6th University, Giza, 12585, Egypt mohamed.eassa.cs@o6u.edu.eg; aahafeez.scis@o6u.edu.eg

3 Department of Computer Science, Faculty of Computers and Informatics, Zagazig University, Zagazig 44519, Egypt, a.metwaly23@fci.zu.edu.eg

4 Department of Computer Engineering and Electronics, Cairo Higher Institute for Engineering, Computer Science and Management, New Cairo, Egypt ahmadnagm@alazhar.edu.eg, A.salama@chi.edu.eg

Abstract:

With more than 5.44 billion users, the Internet is an essential component of everyday life, facilitating e-commerce, interaction, learning, and more. But with the proliferation of harmful Uniform Resource Locators (URLs), this widespread Internet access also raises questions about online security and privacy. Due to their significant advantages of lowering model variance, increasing predictive performance, raising prediction accuracy, and exhibiting strong generalization potential, traditional ensemble models have recently drawn interest. However, there is still work to be done on how to use it to combat rogue URLs. These URLs are dangerous to people and organizations because they frequently lurk behind static links in emails or web pages. Many malicious websites avoid detection despite blacklisting services because they are either newly created or not closely monitored. Hence, we use different machine learning (ML) models such as decision tree, AdaBoosting, Naïve Bayes, random forest, gradient boosting, and XGBoosting. Then these models are evaluated under the neutrosophic framework to deal with uncertainty. The WASPAS method is used to select the best ML model from different models. The results show that the random forest is the best ML model.

Keywords: Prediction of Malicious URLs; Neutrosophic Sets; Uncertainty; Machine Learning; Security.

1. Introduction

Today, about 5.44 billion people, or 67.1 percent of the world's population, have access to the Internet. The core value chain has been impacted by the Internet's revolutionary influence on business operations across all industries and enterprises. A growing quantity of digital traces is left in cyberspace because of people's increased daily interactions with the Internet. Every webpage that people visit is given a Uniform Resource Locator (URL), which acts as a distinct address for online content access. A URL is a combination of a domain name and protocol that acts as a special identification for finding and accessing information on the Internet. Unfortunately, dangerous URLs are becoming more and more common on the Internet.[1], [2].

On the Internet, malicious websites are a common problem that can result in a variety of online crimes, including malware, spam, phishing attacks, and vandalism. Online risks have evolved because hackers are using more complex strategies to trick people and organizations. The technique of imitating the URLs of trustworthy businesses is one that scammers frequently use. This dishonest tactic seeks to fool unwary users into disclosing important information or downloading harmful files. Fraudsters may install malware on the victim's computer, steal their private information, or launch more extensive cyberattacks when a user divulges sensitive data or clicks on a dangerous link.[3], [4].

Unfortunately, it's not always easy to tell if a website is dangerous. The fact that 1 in 10 rogue websites are housed on otherwise secure domains makes it much more difficult. The difficulty of distinguishing and stopping the spread of malicious information on an otherwise safe domain may be the cause of this trend. Additionally, search engine filters may not be able to detect dangerous information housed on HTTPS websites due to the encryption of HTTPS communication, which reduces its exposure. Furthermore, because malicious URLs can hide in static links in emails, Word documents, and websites, they are challenging to find or detect[5], [6].

URL blacklisting is a protection to prevent users from visiting fraudulent or dangerous websites. By identifying websites engaged in questionable activity, a comprehensive database is used to counter URL impersonation assaults. This database contains a comprehensive list of websites that trustworthy organizations, including search engines, hosting companies, and antivirus software, have identified as dangerous or dangerous. However, several techniques, including manual reporting, web crawlers, honeypots, and site analysis algorithms, are frequently used to generate these blacklists, and they are all vulnerable to assaults. Furthermore, traditional detection systems that rely on blacklists are ineffective, restricted in scope, and rigid in an environment where threats are constantly changing.[7], [8].

The use of artificial intelligence (AI) technologies has gained popularity recently; generative adversarial networks (GANs), machine learning (ML), reinforcement learning (RL), and machine learning (ML have all shown promise in improving the efficiency of harmful URL[9], [10] detection. Although ML algorithms have been useful in identifying and avoiding malicious URLs, their efficacy has been patchy and produced a range of outcomes. The constraints stem from

challenges encountered during the extraction of characteristics that differentiate benign URLs from dangerous URLs[11], [12].

Nonetheless, it is commonly known that the shortcomings in detection models' performance call for improvements in their efficacy. Any model changes should be a complete solution that has the potential to produce a highly positive result. Any newly presented machine learning framework should be able to manage big distributed connections, construct processes, and maintain efficient processing speed.[13], [14]. This study applies different ML models and then evaluates them using the Neutrosophic framework to show the best model.

As an extension of the classic set, fuzzy set, and intuitionistic fuzzy set, Wang et al. [15] Recently developed a single-valued neutrosophic set was recently developed, which is a subclass of a neutrosophic set established by Smarandache. [16]. The single-valued neutrosophic set may handle inconsistent, indeterminate, and incomplete information and separately represent truth-membership-degree, indeterminacy-membership-degree, and falsity-membership-degree. The single-value neutrosophic set was first presented by Ye. [17]. The imperfection of information that people get or perceive from the outside world makes all the variables indicated by the single-valued neutrosophic set highly appropriate for human thought.

Because indeterminacy is the area of ignorance regarding a statement's value between truth and untruth, the human brain is undoubtedly unable to produce accurate replies in the form of yes or no, for instance, for the claim "Movie X would be a hit." The intuitionistic fuzzy set is unable to manage and express indeterminacy and inconsistent information, whereas the neutrosophic components are better suited to do so. As a result, the single-valued neutrosophic set has developed quickly and has many uses[18], [19].

The following is a summary of this work's contributions to knowledge:
It offers a way to turn harmful URLs into characteristics that can be quantified and analyzed as cyberattacks.
It investigates machine learning methods that are applicable to the categorization of malevolent incursion.
Similarly, this work builds an ensemble model for the detection and classification of harmful URLs by dynamically selecting and combining models within a heterogeneous ensemble to detect dangerous URLs as efficiently as possible.

It assesses which model performs the best and may be applied to remove harmful URLs. Additionally, the proposed study would offer useful information for identifying network assaults, including distributed denial of service (DDoS), denial of service (DoD), and other specific online threat matrices.

We use the Neutrosophic Framework to select the best ML model based on evaluation matrices. The Neutrosophic set is used to deal with uncertainty.

The WASPAS method is used to select the model.

2. Proposed Methodology

This section shows the methodology of the proposed model using triangular Neutrosophic Numbers (TNNs) and the machine learning models. We show some definitions of the TNNs, and the steps of the WASPAS methodology to select the best alternative (ML model).

Definition 1

We can define the TNNs with three membership functions represented truth, indeterminacy, and falsity such as[20], [21]:

$$X = \left((x_1, x_2, x_3); T_x, I_x, F_x\right) \tag{1}$$

$$T_x(y) = \begin{cases} T_x\left(\frac{y-x_1}{x_2-x_1}\right) & if\ x_1 \le y \le x_2 \\ T_a & if\ y = x_2 \\ T_x\left(\frac{x_3-y}{x_3-x_2}\right) & if\ x_2 \le y \le x_3 \\ 0 & otherwise \end{cases} \tag{2}$$

$$I_x(y) = \begin{cases} \frac{(x_2-y+I_x(y-x_1))}{(x_2-x_1)} & if\ x_1 \le y \le x_2 \\ I_x & if\ y = x_2 \\ \frac{(y-x_2+I_x(x_3-y))}{(x_3-x_2)} & if\ x_2 \le y \le x_3 \\ 1 & otherwise \end{cases} \tag{3}$$

$$F_x(y) = \begin{cases} \frac{(x_2-y+F_x(y-x_1))}{(x_2-x_1)} & if\ x_1 \le y \le x_2 \\ F_x & if\ y = x_2 \\ \frac{(y-x_2+F_x(x_3-y))}{(x_3-x_2)} & if\ x_2 \le y \le x_3 \\ 1 & otherwise \end{cases} \tag{4}$$

We show the operations of the two TNNs such as:

$$x + z = \begin{pmatrix} (x_1 + z_1, x_2 + z_2, x_3 + z_3); \\ T_x \wedge T_z, I_x \vee I_z, F_x \vee F_z \end{pmatrix} \tag{5}$$

$$x - z = \begin{pmatrix} (x_1 - z_3, x_2 - z_2, x_3 - z_1); \\ T_x \wedge T_z, I_x \wedge I_z, F_x \wedge F_z \end{pmatrix} \tag{6}$$

$$x^{-1} = \left(\left(\frac{1}{x_3}, \frac{1}{x_2}, \frac{1}{x_1}\right);\ T_x\ , I_x\ , F_x\right) \tag{7}$$

$$\nabla x = \begin{cases} \left((\nabla x_1, \nabla x_2, \nabla x_3);\ T_x\ , I_x\ , F_x\right) & if\ \nabla > 0 \\ \left((\nabla x_3, \nabla x_2, \nabla x_1);\ T_x\ , I_x\ , F_x\right) & if\ \nabla < 0 \end{cases} \tag{8}$$

$$xz = \begin{cases} \begin{pmatrix} (x_1z_1, x_2z_2, x_3z_3); \\ T_x \wedge T_x, I_x \vee I_z, F_x \vee F_z \end{pmatrix} & if(x_3 > 0, z_3 > 0) \\ \begin{pmatrix} (x_1z_3, x_2z_2, x_3z_1); \\ T_x \wedge T_z, I_x \vee I_z, F_x \vee F_z \end{pmatrix} & if(x_3 < 0, z_3 > 0) \\ \begin{pmatrix} (x_3z_3, x_2z_2, x_1z_1); \\ T_x \wedge T_z, I_x \vee I_z, F_x \vee F_z \end{pmatrix} & if(x_3 < 0, z_3 < 0) \end{cases} \quad (9)$$

$$\frac{x}{z} = \begin{cases} \left(\left(\frac{x_1}{z_3}, \frac{x_2}{z_2}, \frac{x_3}{z_1} \right); T_x \wedge T_z, I_x \vee I_z, F_x \vee F_z \right) & if(x_3 > 0, z_3 > 0) \\ \left(\left(\left(\frac{x_3}{z_3}, \frac{x_2}{z_2}, \frac{x_1}{z_1} \right) \right); T_x \wedge T_z, I_x \vee I_z, F_x \vee F_z \right) & if(x_3 < 0, z_3 > 0) \\ \left(\left(\left(\frac{x_3}{z_1}, \frac{x_2}{z_2}, \frac{x_3}{z_3} \right) \right); T_x \wedge T_z, I_x \vee I_z, F_x \vee F_z \right) & if(x_3 < 0, z_3 < 0) \end{cases} \quad (10)$$

The steps of the WASPAS method to rank the ML models are shown as:

The decision matrix is created. Experts use the TNNs to evaluate the criteria and alternatives. The decision matrix is converted to crisp values and combined to single matrix. The criteria weights are computed using the average method.

The decision matrix is normalized for positive and cost criteria such as:

$$r_{ij} = \frac{y_{ij}}{\max y_{ij}}; i = 1, \dots, m; j = 1, \dots, n \quad (11)$$

$$r_{ij} = \frac{\min y_{ij}}{y_{ij}}; i = 1, \dots, m; j = 1, \dots, n \quad (12)$$

The additive and multiplication relative importance are computed such as:

$$Q_i^{(1)} = \sum_{j=1}^{n} r_{ij} w_j \quad (13)$$

$$Q_i^{(2)} = \prod_{j=1}^{n} (r_{ij})^{w_j} \quad (14)$$

The joint generalized criterion is computed such as:

$$Q_i = \rho Q_i^{(1)} + (1 - \rho) Q_i^{(2)} \quad (15)$$

Value of ρ between 0 and 1.

3. ML results

An extensively used template for creating machine learning models served as the basis for the design. Data collection, preprocessing and planning, algorithm construction, and outcome assessment are the four steps in the process. Data collection, preprocessing, extraction of features and choice, model training, and assessment were the steps involved in creating this structure.

The approach solves the binary classification issue by combining predictions from several classification models into an ensemble to assess if a URL is safe or dangerous. For classification

and prediction, machine learning methods such XGBoost (XGB), Naïve bayes (NB), Ada Boosting (Ada), Random Forest (RF), Gradient Boosting (GB), and Decision Tree (DT) were employed.

Data transformation and standardization, resolving unbalanced datasets, and feature selection were all part of pre-processing. Numerical representations of categorical values were made, noise was eliminated, and imbalance was fixed by resampling. Metrics such as Accuracy, Precision, Recall, and F1 score were used to assess the model's performance.

A malicious URL database (ISCX-URL2016) from the Canadian Institute for Cybersecurity was used in the study. Additionally, the dataset was accessible on Kaggle. This is significant since, rather than the method itself, the kind of dataset greatly influences the classification accuracy of the method. The present research used a dataset of 651,191 URLs to evaluate the effectiveness of the harmful URL forecasting system based on multi-ML. The collection contains 96,457 defacement URLs, 32,520 malicious, 94,111 phishing, and 428,103 benign URLs.

Fig 1 shows the distribution of the URLs per type. Fig 2 shows the length of the http. Fig 3 shows the length of the https. Fig 4 shows the length of URL. Fig 5 shows the length of abnormal URL. Fig 6 shows the heatmap.

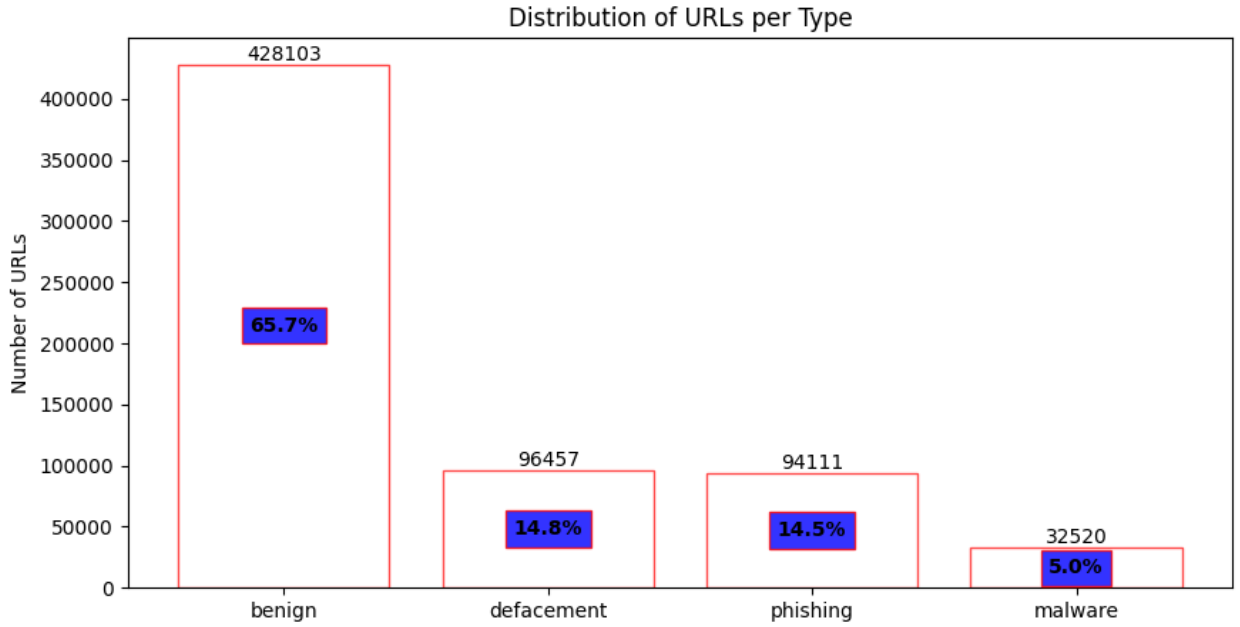

Fig 1. The distribution of the URLs.

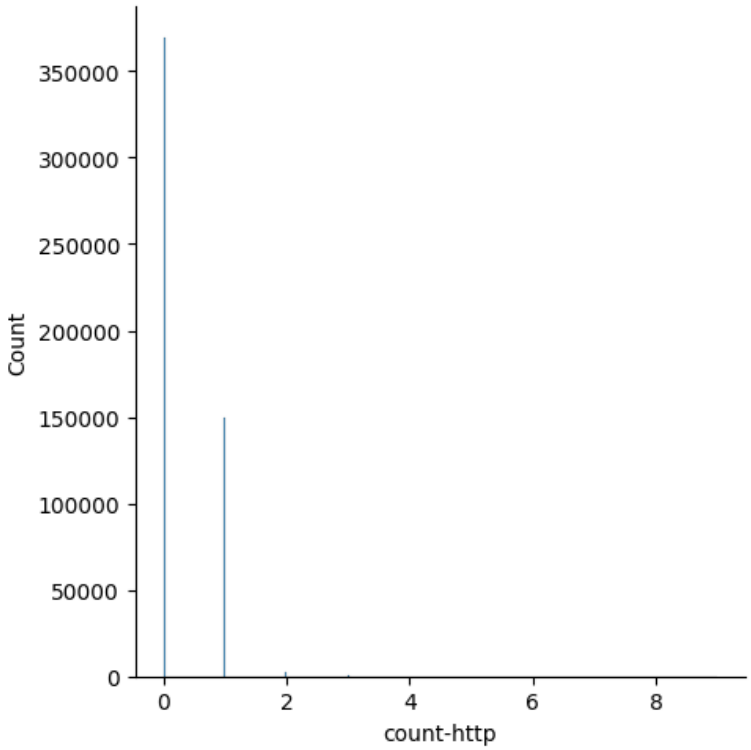

Fig 2. The length of the http.

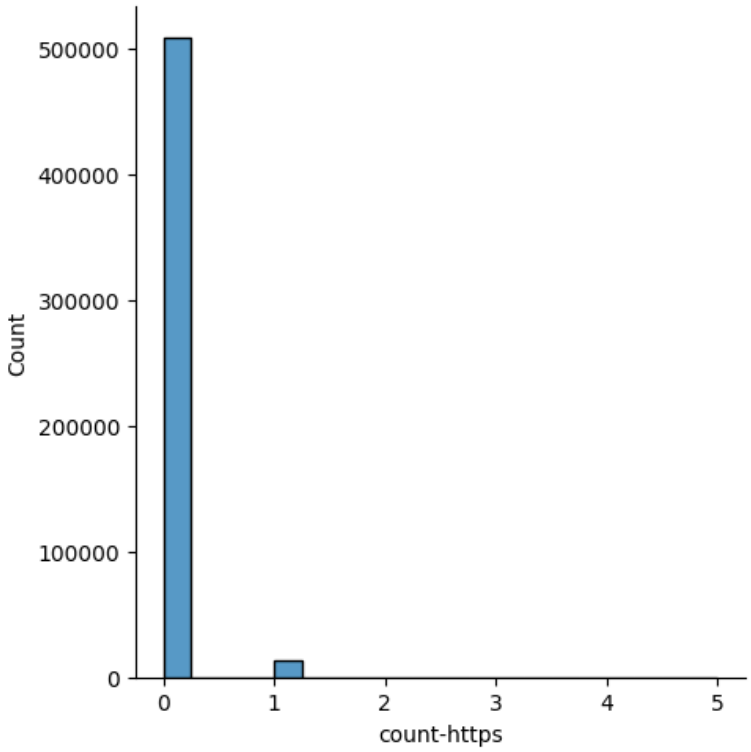

Fig 3. The length of https.

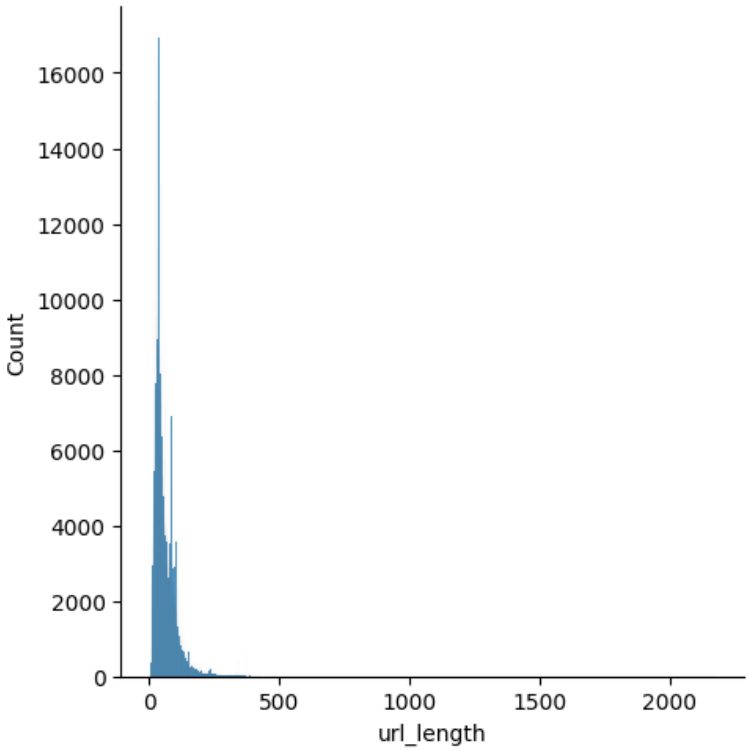

Fig 4. The length of URL.

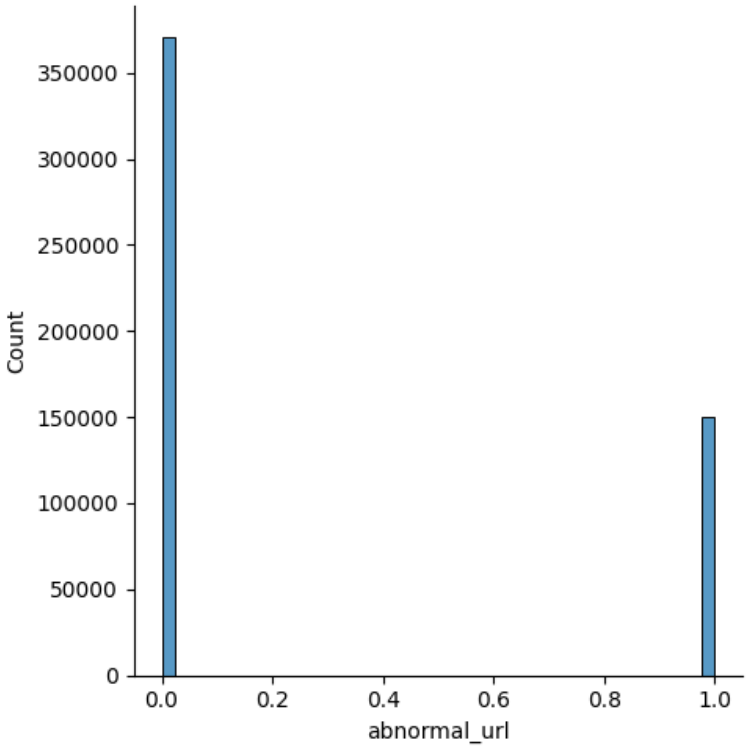

Fig 5. The length of abnormal URL.

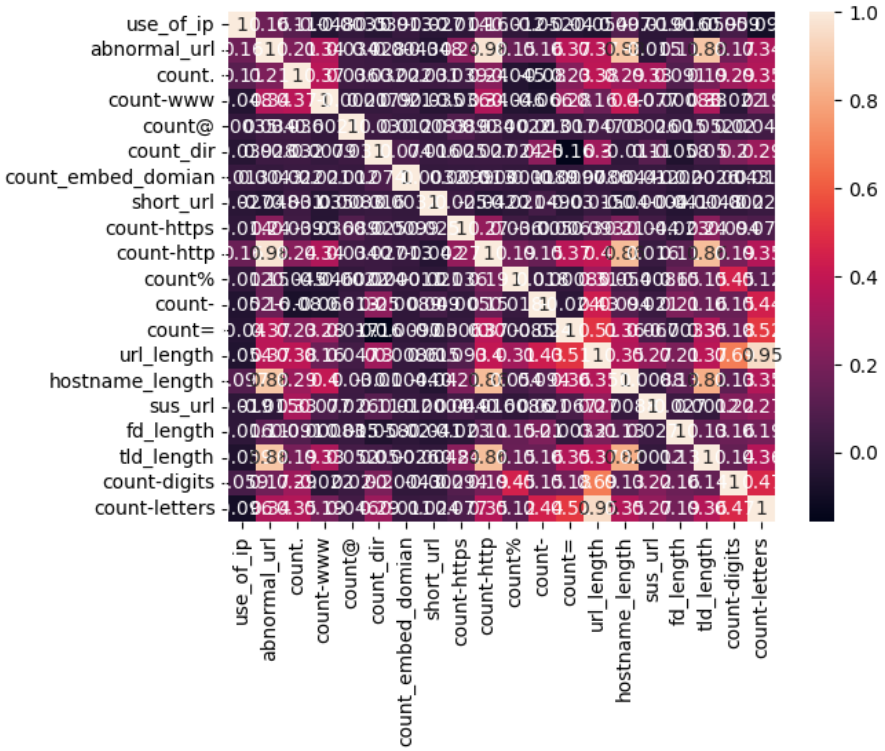

Fig 6. The heatmap of this study.

We trained six ML models on the URL dataset. Then we obtained the evaluation matrices as shown in Fig 7.

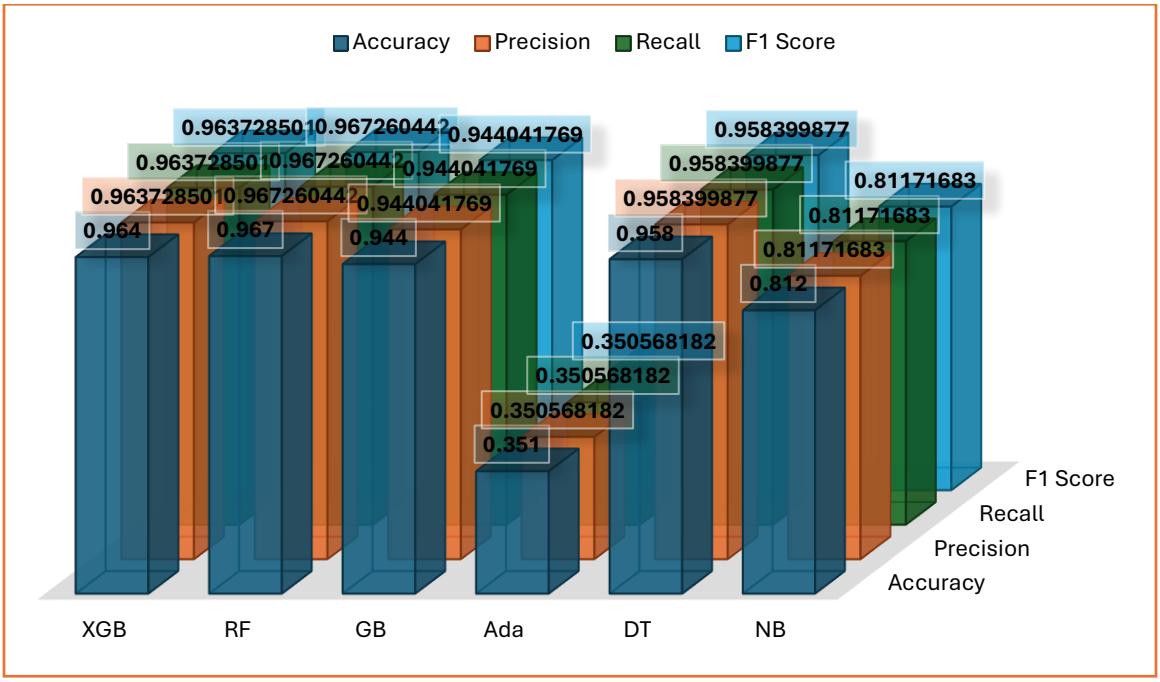

Fig 7. The results of ML models.

4. Analysis of Neutrosophic ML models

This section shows the best ML model under different criteria such as evaluation matrices. We let seven experts evaluate the ML models based on four evaluation matrices as shown in Table 1. The decision matrix is converted to crisp values. Then we compute the criteria weights such as URLC1= 0.285030715, URLC1= 0.208988038, URLC1= 0.221694148, URLC1= 0.2842871.

Eq. (11) is used to normalize the decision matrix as shown in Table 2.

The additive and multiplication relative importance are computed using Eq. (13 and 14) as shown in Tables 3 and 4.

The joint generalized criterion is computed using Eq. (15). The rank of the alternatives is obtained such as: URLA4> URLA1>URLA6>URLA5> URLA2>URLA3. We show the RF ML model is the best model.

Table 1. The TNNs.

	$URLC_1$	$URLC_2$	$URLC_3$	$URLC_4$
$URLA_1$	((1,1,1);0.5,0.5,0.5)	((2,3,4);0.3,0.75,0.70)	((4,5,6);0.8,0.15,0.20)	((6,7,8);0.9,0.10,0.10)
$URLA_2$	((1,1,1);0.5,0.5,0.5)	((3,4,5);0.35,0.60,0.40)	((1,2,3);0.4,0.60,0.65)	((9,9,9);1.00,0.00,0.00)
$URLA_3$	((2,3,4);0.3,0.75,0.70)	((4,5,6);0.8,0.15,0.20)	((6,7,8);0.9,0.10,0.10)	((9,9,9);1.00,0.00,0.00)
$URLA_4$	((6,7,8);0.9,0.10,0.10)	((4,5,6);0.8,0.15,0.20)	((2,3,4);0.3,0.75,0.70)	((1,1,1);0.5,0.5,0.5)
$URLA_5$	((9,9,9);1.00,0.00,0.00)	((6,7,8);0.9,0.10,0.10)	((4,5,6);0.8,0.15,0.20)	((2,3,4);0.3,0.75,0.70)
$URLA_6$	((9,9,9);1.00,0.00,0.00)	((9,9,9);1.00,0.00,0.00)	((6,7,8);0.9,0.10,0.10)	((9,9,9);1.00,0.00,0.00)
	$URLC_1$	$URLC_2$	$URLC_3$	$URLC_4$
$URLA_1$	((9,9,9);1.00,0.00,0.00)	((1,2,3);0.4,0.60,0.65)	((3,4,5);0.35,0.60,0.40)	((1,1,1);0.5,0.5,0.5)
$URLA_2$	((6,7,8);0.9,0.10,0.10)	((4,5,6);0.8,0.15,0.20)	((6,7,8);0.9,0.10,0.10)	((2,3,4);0.3,0.75,0.70)
$URLA_3$	((6,7,8);0.9,0.10,0.10)	((2,3,4);0.3,0.75,0.70)	((4,5,6);0.8,0.15,0.20)	((4,5,6);0.8,0.15,0.20)
$URLA_4$	((4,5,6);0.8,0.15,0.20)	((1,1,1);0.5,0.5,0.5)	((2,3,4);0.3,0.75,0.70)	((6,7,8);0.9,0.10,0.10)
$URLA_5$	((2,3,4);0.3,0.75,0.70)	((3,4,5);0.35,0.60,0.40)	((1,1,1);0.5,0.5,0.5)	((9,9,9);1.00,0.00,0.00)
$URLA_6$	((1,1,1);0.5,0.5,0.5)	((6,7,8);0.9,0.10,0.10)	((3,4,5);0.35,0.60,0.40)	((3,4,5);0.35,0.60,0.40)
	$URLC_1$	$URLC_2$	$URLC_3$	$URLC_4$
$URLA_1$	((1,1,1);0.5,0.5,0.5)	((2,3,4);0.3,0.75,0.70)	((4,5,6);0.8,0.15,0.20)	((6,7,8);0.9,0.10,0.10)
$URLA_2$	((3,4,5);0.35,0.60,0.40)	((3,4,5);0.35,0.60,0.40)	((1,2,3);0.4,0.60,0.65)	((9,9,9);1.00,0.00,0.00)
$URLA_3$	((1,2,3);0.4,0.60,0.65)	((1,1,1);0.5,0.5,0.5)	((6,7,8);0.9,0.10,0.10)	((9,9,9);1.00,0.00,0.00)
$URLA_4$	((9,9,9);1.00,0.00,0.00)	((3,4,5);0.35,0.60,0.40)	((1,1,1);0.5,0.5,0.5)	((1,1,1);0.5,0.5,0.5)
$URLA_5$	((6,7,8);0.9,0.10,0.10)	((1,2,3);0.4,0.60,0.65)	((3,4,5);0.35,0.60,0.40)	((1,1,1);0.5,0.5,0.5)
$URLA_6$	((4,5,6);0.8,0.15,0.20)	((9,9,9);1.00,0.00,0.00)	((1,2,3);0.4,0.60,0.65)	((3,4,5);0.35,0.60,0.40)
	$URLC_1$	$URLC_2$	$URLC_3$	$URLC_4$
$URLA_1$	((9,9,9);1.00,0.00,0.00)	((1,2,3);0.4,0.60,0.65)	((3,4,5);0.35,0.60,0.40)	((9,9,9);1.00,0.00,0.00)
$URLA_2$	((6,7,8);0.9,0.10,0.10)	((4,5,6);0.8,0.15,0.20)	((1,1,1);0.5,0.5,0.5)	((6,7,8);0.9,0.10,0.10)
$URLA_3$	((1,2,3);0.4,0.60,0.65)	((2,3,4);0.3,0.75,0.70)	((3,4,5);0.35,0.60,0.40)	((1,2,3);0.4,0.60,0.65)
$URLA_4$	((1,1,1);0.5,0.5,0.5)	((1,1,1);0.5,0.5,0.5)	((1,2,3);0.4,0.60,0.65)	((3,4,5);0.35,0.60,0.40)
$URLA_5$	((3,4,5);0.35,0.60,0.40)	((1,1,1);0.5,0.5,0.5)	((9,9,9);1.00,0.00,0.00)	((1,1,1);0.5,0.5,0.5)
$URLA_6$	((1,2,3);0.4,0.60,0.65)	((3,4,5);0.35,0.60,0.40)	((6,7,8);0.9,0.10,0.10)	((3,4,5);0.35,0.60,0.40)
	$URLC_1$	$URLC_2$	$URLC_3$	$URLC_4$
$URLA_1$	((6,7,8);0.9,0.10,0.10)	((2,3,4);0.3,0.75,0.70)	((4,5,6);0.8,0.15,0.20)	((6,7,8);0.9,0.10,0.10)
$URLA_2$	((4,5,6);0.8,0.15,0.20)	((3,4,5);0.35,0.60,0.40)	((1,2,3);0.4,0.60,0.65)	((6,7,8);0.9,0.10,0.10)
$URLA_3$	((2,3,4);0.3,0.75,0.70)	((4,5,6);0.8,0.15,0.20)	((6,7,8);0.9,0.10,0.10)	((4,5,6);0.8,0.15,0.20)

URLA4	((1,1,1);0.5,0.5,0.5)	((4,5,6);0.8,0.15,0.20)	((2,3,4);0.3,0.75,0.70)	((2,3,4);0.3,0.75,0.70)
URLA5	((3,4,5);0.35,0.60,0.40)	((6,7,8);0.9,0.10,0.10)	((4,5,6);0.8,0.15,0.20)	((1,1,1);0.5,0.5,0.5)
URLA6	((6,7,8);0.9,0.10,0.10)	((6,7,8);0.9,0.10,0.10)	((6,7,8);0.9,0.10,0.10)	((3,4,5);0.35,0.60,0.40)
	URLC1	URLC2	URLC3	URLC4
URLA1	((9,9,9);1.00,0.00,0.00)	((6,7,8);0.9,0.10,0.10)	((3,4,5);0.35,0.60,0.40)	((1,2,3);0.4,0.60,0.65)
URLA2	((6,7,8);0.9,0.10,0.10)	((4,5,6);0.8,0.15,0.20)	((1,1,1);0.5,0.5,0.5)	((4,5,6);0.8,0.15,0.20)
URLA3	((4,5,6);0.8,0.15,0.20)	((2,3,4);0.3,0.75,0.70)	((2,3,4);0.3,0.75,0.70)	((2,3,4);0.3,0.75,0.70)
URLA4	((4,5,6);0.8,0.15,0.20)	((2,3,4);0.3,0.75,0.70)	((9,9,9);1.00,0.00,0.00)	((1,1,1);0.5,0.5,0.5)
URLA5	((2,3,4);0.3,0.75,0.70)	((1,1,1);0.5,0.5,0.5)	((2,3,4);0.3,0.75,0.70)	((3,4,5);0.35,0.60,0.40)
URLA6	((1,1,1);0.5,0.5,0.5)	((3,4,5);0.35,0.60,0.40)	((1,1,1);0.5,0.5,0.5)	((1,2,3);0.4,0.60,0.65)
	URLC1	URLC2	URLC3	URLC4
URLA1	((1,2,3);0.4,0.60,0.65)	((3,4,5);0.35,0.60,0.40)	((2,3,4);0.3,0.75,0.70)	((3,4,5);0.35,0.60,0.40)
URLA2	((3,4,5);0.35,0.60,0.40)	((1,1,1);0.5,0.5,0.5)	((1,1,1);0.5,0.5,0.5)	((1,1,1);0.5,0.5,0.5)
URLA3	((3,4,5);0.35,0.60,0.40)	((2,3,4);0.3,0.75,0.70)	((3,4,5);0.35,0.60,0.40)	((2,3,4);0.3,0.75,0.70)
URLA4	((2,3,4);0.3,0.75,0.70)	((2,3,4);0.3,0.75,0.70)	((2,3,4);0.3,0.75,0.70)	((9,9,9);1.00,0.00,0.00)
URLA5	((1,1,1);0.5,0.5,0.5)	((1,1,1);0.5,0.5,0.5)	((1,1,1);0.5,0.5,0.5)	((1,1,1);0.5,0.5,0.5)
URLA6	((3,4,5);0.35,0.60,0.40)	((3,4,5);0.35,0.60,0.40)	((3,4,5);0.35,0.60,0.40)	((2,3,4);0.3,0.75,0.70)

Table 2. The normalized TNNs.

	URLC1	URLC2	URLC3	URLC4
URLA1	1	0.338426	0.674362	0.859389
URLA2	0.772338	0.504167	0.368165	1
URLA3	0.439639	0.335185	1	0.794635
URLA4	0.721958	0.351852	0.498177	0.539778
URLA5	0.601711	0.462963	0.758809	0.378816
URLA6	0.654468	1	0.866343	0.49445

Table 3. The additive relative importance.

	URLC1	URLC2	URLC3	URLC4
URLA1	0.285031	0.070727	0.149502	0.244313
URLA2	0.22014	0.105365	0.08162	0.284287
URLA3	0.125311	0.07005	0.221694	0.225904
URLA4	0.20578	0.073533	0.110443	0.153452
URLA5	0.171506	0.096754	0.168224	0.107692
URLA6	0.186543	0.208988	0.192063	0.140566

Table 4. The multiplication relative importance.

	URLC1	URLC2	URLC3	URLC4
URLA1	1	0.797377	0.916361	0.957836
URLA2	0.929013	0.866646	0.801298	1
URLA3	0.791173	0.795775	1	0.93674
URLA4	0.911321	0.803886	0.856864	0.839213
URLA5	0.865205	0.85134	0.940646	0.758844
URLA6	0.886181	1	0.968693	0.818546

5. Conclusions

Because it facilitates extensive communication and trade, the internet is essential to modern life. However, it also exposes consumers to online dangers, such as bad URLs that are used for spam, phishing, malware, SCER, DoD, and performance degradation attacks. These URLs are frequently concealed in static links seen in emails and webpages, making them challenging to find. This work offers a machine learning strategy for recognizing and classifying dangerous URLs. Six ML models were among the ensemble models that were examined. These ML models are trained on the URL dataset. Triangular Neutrosophic Numbers (TNNs) framework is used to evaluate the different ML models under uncertainty. We used the WASPAS method to select the best ML model under four evaluation matrices. The results of this study show the random forest is the best ML model to predict bad URLs.

References

[1] A. E. Omolara and M. Alawida, "DaE2: Unmasking malicious URLs by leveraging diverse and efficient ensemble machine learning for online security," *Comput. Secur.*, vol. 148, p. 104170, 2025.

[2] H. V. S. Aalla, N. R. Dumpala, and M. Eliazer, "Malicious URL prediction using machine learning techniques," *Ann. Rom. Soc. Cell Biol.*, vol. 25, no. 5, pp. 2170–2176, 2021.

[3] F. Vanhoenshoven, G. Nápoles, R. Falcon, K. Vanhoof, and M. Köppen, "Detecting malicious URLs using machine learning techniques," in *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, IEEE, 2016, pp. 1–8.

[4] P. Wanda and H. J. Jie, "URLDeep: Continuous Prediction of Malicious URL with Dynamic Deep Learning in Social Networks.," *Int. J. Netw. Secur.*, vol. 21, no. 6, pp. 971–978, 2019.

[5] D. Sahoo, C. Liu, and S. C. H. Hoi, "Malicious URL detection using machine learning: A survey," *arXiv Prepr. arXiv1701.07179*, 2017.

[6] D. Vaishnavi, S. Suwetha, Y. B. Jinila, R. Subhashini, and S. P. Shyry, "A comparative analysis of machine learning algorithms on malicious URL prediction," in *2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS)*, IEEE, 2021, pp. 1398–1402.

[7] A. B. Sayamber and A. M. Dixit, "Malicious URL detection and identification," *Int. J. Comput. Appl.*, vol. 99, no. 17, pp. 17–23, 2014.

[8] D. K. Mondal, B. C. Singh, H. Hu, S. Biswas, Z. Alom, and M. A. Azim, "SeizeMaliciousURL: A novel learning approach to detect malicious URLs," *J. Inf. Secur. Appl.*, vol. 62, p. 102967, 2021.

[9] N. Reyes-Dorta, P. Caballero-Gil, and C. Rosa-Remedios, "Detection of malicious URLs using machine learning," *Wirel. Networks*, pp. 1–18, 2024.

[10] S. Mohanty, A. A. Acharya, T. Gaber, N. Panda, E. Eldesouky, and I. A. Hameed, "An

efficient hybrid feature selection technique towards prediction of suspicious URLs in IoT environment," *IEEE Access*, 2024.

[11] R. B. Hani, M. Amoura, M. Ammourah, Y. A. Khalil, and M. Swailm, "Malicious URL detection using machine learning," in *2024 15th International Conference on Information and Communication Systems (ICICS)*, IEEE, 2024, pp. 1–5.

[12] A. S. Rafsanjani, N. B. Kamaruddin, M. Behjati, S. Aslam, A. Sarfaraz, and A. Amphawan, "Enhancing malicious URL detection: A novel framework leveraging priority coefficient and feature evaluation," *IEEE Access*, 2024.

[13] S. Abad, H. Gholamy, and M. Aslani, "Classification of malicious URLs using machine learning," *Sensors*, vol. 23, no. 18, p. 7760, 2023.

[14] G. Wejinya and S. Bhatia, "Machine learning for malicious URL detection," in *ICT Systems and Sustainability: Proceedings of ICT4SD 2020, Volume 1*, Springer, 2021, pp. 463–472.

[15] H. Wang, F. Smarandache, Y. Zhang, and R. Sunderraman, "Single valued neutrosophic sets," *Infin. study*, vol. 12, 2010.

[16] F. Smarandache, "Neutrosophy: neutrosophic probability, set, and logic: analytic synthesis & synthetic analysis," 1998.

[17] J. Ye, "Single-valued neutrosophic minimum spanning tree and its clustering method," *J. Intell. Syst.*, vol. 23, no. 3, pp. 311–324, 2014.

[18] S. A. Edalatpanah, "Data envelopment analysis based on triangular neutrosophic numbers," *CAAI Trans. Intell. Technol.*, vol. 5, no. 2, pp. 94–98, 2020.

[19] S. K. Das and S. A. Edalatpanah, "A new ranking function of triangular neutrosophic number and its application in integer programming," *Int. J. Neutrosophic Sci.*, vol. 4, no. 2, pp. 82–92, 2020.

[20] A. Chakraborty, S. P. Mondal, A. Ahmadian, N. Senu, S. Alam, and S. Salahshour, "Different forms of triangular neutrosophic numbers, de-neutrosophication techniques, and their applications," *Symmetry (Basel).*, vol. 10, no. 8, p. 327, 2018.

[21] Q. Wang *et al.*, "A novel method for solving multiobjective linear programming problems with triangular neutrosophic numbers," *J. Math.*, vol. 2021, no. 1, p. 6631762, 2021.

Received: Nov. 7, 2024. Accepted: April 2, 2025