frontiers | Frontiers in Computer Science

TYPE Original Research
PUBLISHED 01 October 2025
DOI 10.3389/fcomp.2025.1582206

OPEN ACCESS

EDITED BY
Stelvio Cimato,
University of Milan, Italy

REVIEWED BY
Muath Obaidat,
The City University of New York, United States
Abdur Rasool,
University of Hawaii at Manoa, United States

*CORRESPONDENCE
Mahmoud Murhej
mahmoudmrhej@gmail.com
G. Nallasivan
drnallasivang@veltech.edu.in

RECEIVED 25 February 2025
ACCEPTED 26 August 2025
PUBLISHED 01 October 2025

CITATION
Murhej M and Nallasivan G (2025) Component features based enhanced phishing website detection system using EfficientNet, FH-BERT, and SELU-CRNN methods.
Front. Comput. Sci. 7:1582206.
doi: 10.3389/fcomp.2025.1582206

Component features based enhanced phishing website detection system using EfficientNet, FH-BERT, and SELU-CRNN methods

Mahmoud Murhej * and G. Nallasivan *

Department of Computer Science & Engineering (CSE), Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology, Chennai, India

Introduction: Phishing is a type of cybercrime used by hackers to steal sensitive user information, making it essential to detect phishing attacks on websites. Many prevailing works have utilized Uniform Resource Locator (URL) links and Document Object Model (DOM) tree structures for Phishing Website Detection (PWD). However, since phishing websites imitate legitimate websites, these approaches often produce inaccurate detection results.

Methods: To enhance detection efficiency, we propose a PWD system that focuses on important website features and components. The process begins with collecting URL links from phishing website datasets, followed by the generation of Hypertext Markup Language (HTML) formats. A DOM tree structure is then constructed from the HTML, and components are extracted along with Natural Language Processing (NLP) features, credentials, URL, DOM tree similarity, and component features. The DOM-tree components are converted into score values using Feature Hasher-Bidirectional Encoder Representations from Transformers (FH-BERT). These score values are fused with component features, and significant features are selected using an Entropy-based Chameleon Swarm Algorithm (ECSA).

Results: The final classification is performed by Scaled Exponential Linear Unit Convolutional Recurrent Neural Network (SELU-CRNN). Simulation results demonstrate that the proposed technique improves PWD performance, achieving higher accuracy (98.42%) and reduced training time (63,003 ms) compared to prevailing methods.

Discussion: By integrating component, semantic, and structural features, the proposed model enhances both robustness and efficiency, making it an effective solution for phishing website detection.

KEYWORDS

phishing website detection, cybersecurity, component features, URL, DOM tree, EfficientNet, phishing attacks, SELU-CRNN

1 Introduction

With a recent advancement of digital services (social media, online gaming applications, and financial services; Alsariera et al., 2020), cyber security threats have increased (Alam et al., 2020; Kumar et al., 2020). Among various cyber threats, Phishing Attacks (PA) pose significant risks by targeting users through deceptive websites (Kumar et al., 2023; Rashid et al., 2020). Website phishing involves creating fake websites that

closely resemble legitimate ones to trick people into revealing private information (Balogun et al., 2021). The PA mainly focused on service blocking and illegal digital content (Korkmaz et al., 2020) to create privacy violation problems (Alkawaz et al., 2020). The phishing attack sends a deceptive message to Internet users via emails, which appear to be from legitimate sources (Safi and Singh, 2023; Nahapetyan et al., 2024). The attackers attempt to urge the users to enter their sensitive information if the users access the fraudulent websites (Wei et al., 2020). Owing to a lack of cybersecurity knowledge, people are deceived by these phishing websites quickly (Basit et al., 2020; Abedin et al., 2020). Further, the system's performance deteriorated due to the phished websites, thus influencing customer service. Hence, identifying these phishing websites and alerting internet users to protect themselves from being attacked is very important (Abuzuraiq et al., 2020). There are numerous countermoves available for detecting and escaping from PA (Lakshmanarao et al., 2021). However, they are less effective in detecting phishing websites because the phishing attackers utilize a secure server connection to do illegal activity (Do et al., 2022). Therefore, for detecting PA, several Machine Learning (ML) approaches, namely decision tree and random forest, and Deep Learning (DL) approaches, namely Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM), were utilized recently (Liu et al., 2021; Mughaid et al., 2022). Essentially, phishing websites increasingly utilize deceptive visual components like images, forms, and embedded media to mimic legitimate sites and evade detection. None of the existing studies focused on the key visual cues. Hence, a robust detection system that integrates visual, semantic, and structural features is essential to improve the detection efficacy and security level. Therefore, this work is proposed with the motive of developing a component-aware detection framework that utilizes EfficientNet and SELU-CRNN for enhanced robustness.

1.1 Problem statement

The conventional PWD systems have some common drawbacks, which are listed further. The prevailing works detected phishing websites by analyzing the URL and DOM tree structure without considering the important website components. Additionally, the prevailing approaches faced complexity in distinguishing phishing websites from legitimate websites as they were somewhat similar to each other and attained more false alarm rates. Also, some of the existing works concentrated only on a few features from the URL link and often overlooked the NLP features, increasing the differentiation problem's complexity.

1.2 Major contributions

The chief contributions of the proposed technique in detecting phishing websites are given further. To analyze component features (images, video links, input forms, and embedded objects) from the DOM tree structure, the proposed work utilizes EfficientNet, enhancing the recognition efficiency of the phishing websites. To mitigate the false alarm rate, the proposed system analyses a wide range of URL-based features. Furthermore, the proposed framework employs an SELU-CRNN to detect the PA with high training stability and enhanced convergence speed. Moreover, the proposed model performs an in-depth analysis of NLP-based features, DOM tree similarity features, and semantic embedding to improve the model's supremacy. The proposed framework ensures an improved detection rate by integrating these diverse feature types through a feature fusion mechanism and feature selection using ECSA.

The formation of the proposed work is described as: Section 2 explains the analysis of the related works; the proposed mechanism is exemplified in Section 3; the experimental analysis outcomes are demonstrated in Section 4; lastly, Section 5 concludes the paper with the conclusion, limitation, and future recommendations.

2 Literature survey

Tang and Mahmoud (2022) established an effective and real-time PWD system grounded on a DL approach. The experimental analysis showed a dynamic accuracy in PWD. Nevertheless, the system failed to detect phishing with more URL characters. So, Karim et al. (2023) established a hybrid ML technique grounded on PWD from URLs. Here, the canopy feature selection system centered on the Cross-fold validation with a grid search parameter was utilized, improving the model's performance. However, the model required a high computational cost. Furthermore, Somesha et al. (2020) propounded an efficient DL technique-based PWD system. By utilizing Deep Neural Network (DNN), LSTM, and CNN, the PWD was achieved. This model had better efficacy. Yet, there was a possibility for an increased false detection rate. Therefore, the appropriate features were selected by Sabahno and Safara (2021) using an Improved Spotted Hyena Optimization (ISHO) approach. Then, PWD was achieved by the Support Vector Machine (SVM) with high superiority. However, the system had insignificant detection results as it dropped into the local optima problem. In order to develop an efficient PWD, the deep optimizer approach named Swarm Intelligence-Binary Bat Algorithm (SI-BBA) was employed in Kumar et al. (2023) for classifying the website phishing URLs. But, it had considerable processing time. Furthermore, the need for feature extraction in PWD was eliminated by Purwanto et al. (2022) through the compression of two similar websites using Normalized Compression Distance (NCD). Nevertheless, the system had less adaptability. To detect phishing sites at different scales, three Multi-scale semantic deep fusion models were employed by Liu et al. (2022). It depicted the lesser error-proneness of the model. However, the credentials and NLP features were not concentrated, increasing the misclassification rate. Subsequently, the features of the various websites were optimized through the Particle Swarm Optimization (PSO)-centric feature weighting technique in Ali and Malebary (2020) for identifying phishing as well as legitimate websites. But, the computation time was higher due to the numerous network layers. Several features of the websites were analyzed through the Phishing Index Login Websites Dataset (PILWD) by Sanchez-Paniagua et al. (2022). Further, the Light Gradient Boosting Machine (LightGBM) classifier was employed to detect phishing websites. Even though it provided superior performance, the model was delayed in processing the URL links. Moreover, the impact of different features in PWD was assessed by Zhu et al. (2020). Here,

an Optimal Features-centric Artificial Neural Network was used for PWD. Thus, the framework obtained higher PWD performance. But, the system had high computational complexity. Various DL algorithms, including LSTM and Gated Recurrent Unit (GRU), were utilized by Benavides-Astudillo et al. (2023) for detecting phishing websites. But, the computational complexity was higher due to training with multiple DL networks. Also, Castano et al. (2023) focused on the HTML contents and the textual features of the website. Further, the embeddings were fused, and the phishing was detected using the Linear classifier. Yet, the important features were not chosen, increasing training time. Subsequently, a phishing website dataset was developed by Colhak et al. (2024) to detect phishing websites. However, the system failed to determine the language-based and credential features, thus resulting in insufficient performance. Further, the model demonstrated by Alqahtani et al. (2022) introduced a DL framework for detecting phishing websites. Here, a Deep Autoencoder (DAE) network was employed for phishing detection. However, the system did not concentrate on determining whether the URL was active, which reduced the system's performance. Also, the phishing detection framework presented by Feng et al. (2018) was based on neural network classification. This methodology utilized the design risk minimization principle along with the Monte Carlo algorithm for classification. But, this model had less generalizability. In addition, Aamir and Zaidi (2019) propounded a robust Distributed of Denial of Service (DDoS) attack detection framework based on feature engineering and ML approaches. Here, the feature engineering was done to improve the model's dominance. Thereafter, the selected features were fed into several ML approaches like K-Nearest Neighbor (KNN) and Artificial Neural Network (ANN) to predict the DDoS attack. Thus, the suggested strategic-level framework improved the security level. But, this framework was not generalized well enough to handle the different data distributions. Moreover, Aslam et al. (2024) implemented an optimal phishing URL detection model based on an LSTM-based stacked generalization approach. In phase 1, base classifiers like KNN, SVM, and NB were utilized to predict the presence of URL phishing. Also, stacked LSTM with adaptive optimizers were used in phase 2 to detect URL phishing effectively. This model obtained a noticeable detection rate. Nevertheless, this approach didn't focus on monitoring the website's URL active status. But, none of the existing phishing detection models mainly relied on analyzing URL features and DOM tree structures, ignoring critical visual and interactive components of websites (images, videos, and forms). Therefore, the proposed methodology was developed with the motive of eliminating the notable limitations in the existing works of PWD. Here, the proposed system establishes a robust phishing website detection framework that integrates visual and structural cues, improving the dependability and trustworthiness of the model.

2.1 Research novelty

To date, none of the prevailing PWD works have concentrated on the component features of the URL that provide detailed insight into URL characteristics like domain name, path, port number, and query string. Therefore, the proposed work incorporates the component features using the EfficientNet, capturing meaningful patterns from URL structures. Additionally, the prevailing works (Liu et al., 2022) and (Karim et al., 2023) had limited PWD accuracy and effectiveness due to insufficient consideration of key features like URL length, page rank, and NLP-based cues. In the proposed work, these features are extracted during the feature extraction phase to enhance detection performance. Further, the existing phishing detection model (Somesha et al., 2020) had a considerable false detection rate due to the insignificant feature representation. Hence, the similarity characters among the DOM tree features are analyzed for the reliable identification of the PWD. Subsequently, the analyzed features are fused prior to the selection of optimal features using the proposed ECSA, thereby selecting the most informative features. The proposed work reduces dimensionality while preserving important information, improving model efficiency. The proposed approach proficiently addresses the high time complexity issue observed in Ali and Malebary (2020). Hence, the proposed work provides an efficient and robust phishing website detection system grounded on multi-level feature analysis of the website URL.

3 Proposed methodology of PWD system based on SELU-CRNN

This work proposes an effective PWD system grounded on the efficient analysis of various features from the URLs, the created DOM tree, and the extracted important components. Afterward, the features are optimally selected by ECSA and classified by SELU-CRNN. Figure 1 displays the proposed PWD system's architecture.

The detailed overview of the proposed architecture is as follows: Initially, the URL links are collected. Next, the corresponding HTML formats are generated for each URL, followed by the creation of the DOM tree. Various features are then extracted, including URL features from the URL links, and then NLP, credential, and DOM tree similarity features from the DOM structure. Also, component features are extracted using EfficientNet from the identified components. Simultaneously, by using FH-BERT, word embeddings are performed on the generated DOM tree. These embedded words and extracted features are then fused through a feature fusion process. To enhance the model's effectiveness, ECSA is used for feature selection. Finally, the classification is performed using the SELU-CRNN model. The flowchart for the proposed PWD is represented in Figure 2.

3.1 URL link and HTML

The URL links are the reference sources for the web address, and they are used to detect phishing websites. The (w) number of URL links (U) collected from the phishing website dataset. Thereafter, the HTML (ι) format is created for the collected U. HTML provides the structural semantics for U, namely paragraphs, headings, quotes, links, etc.

$$U = \{U_1, U_2, ..., U_w\} \tag{1}$$

$$\iota\,(U) = \{\iota\,(U_1)\,, \iota\,(U_2)\,, ..., \iota_z\,(U_w)\} \tag{2}$$

For w number of U, the z number of ι formats is specified.

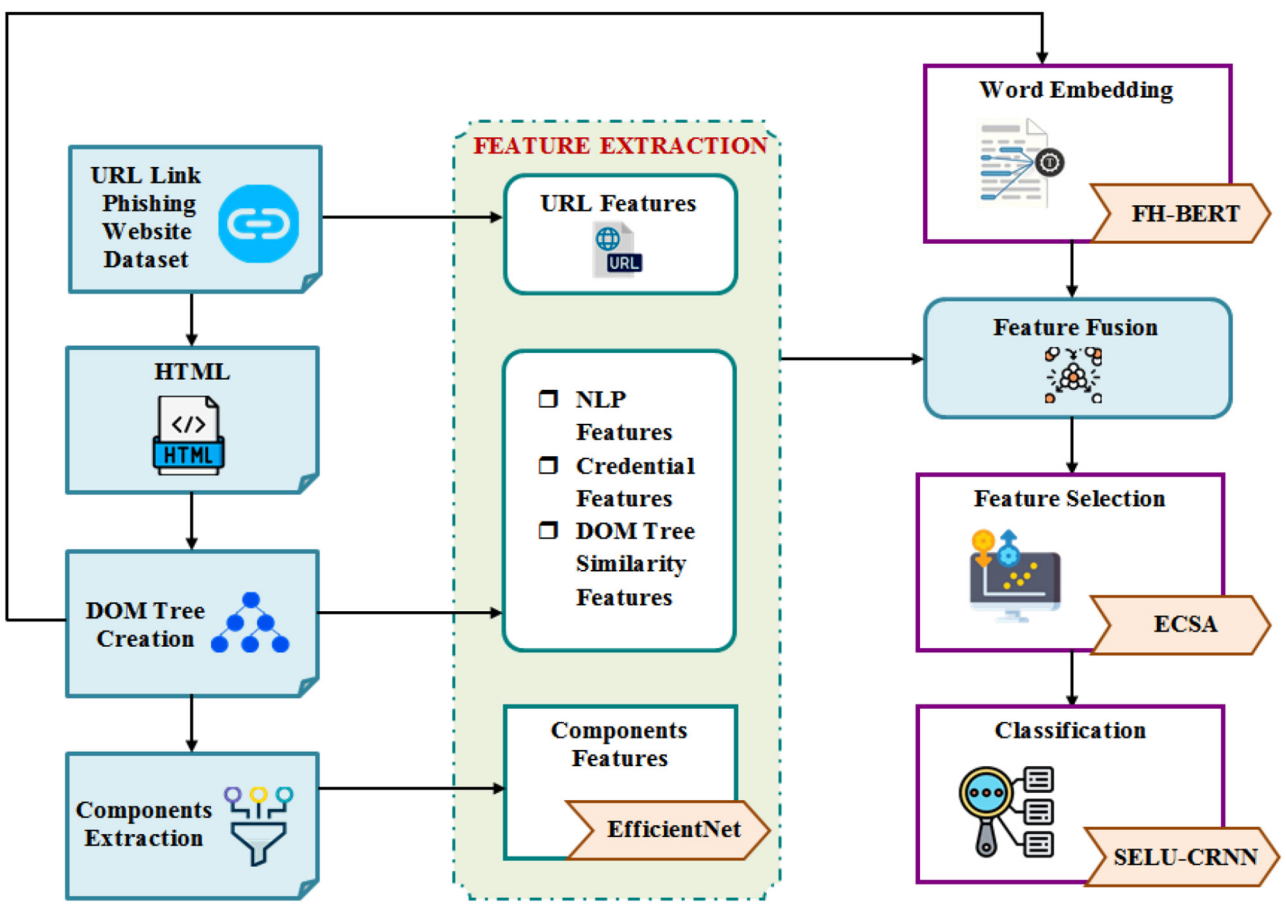

FIGURE 1
Architecture of the proposed PWD system.

3.2 DOM tree creation

The DOM tree (DT) structure was created for every element in ι. DOM tree is a language-independent interface that transforms HTML into a tree structure in which each node represents a tag ι. The browser (br) recovers ι from the server. The recovered ι is implied as ι_{rec} and the ι of U is described as,

$$br \rightarrow \iota\,(U) \tag{3}$$

Afterward, the br examines the ι_{rec} structure and creates the DOM tree for it. Thereafter, a number of elements (el) and b number of attributes (at) are added with the DT structure.

$$br \rightarrow DT \tag{4}$$

$$DT \Rightarrow \left[el_a, at_b\right] \tag{5}$$

Next, br will provide the web page based on the created DT structure.

3.3 Components extraction

The important components (C), such as pictures (C_1) and video links (C_2), are extracted from the created DT structures to train the classifier for PWD. It is expressed as,

$$C = \{C_1, C_2\} \tag{6}$$

Thus, these are the extracted components that help the classifier identify suspicious content.

3.4 Feature extraction

Next, the features are extracted from (U), (C), (DT) for determining whether the website is a phishing site or not, enhancing the accuracy of PWD.

3.4.1 URL features

Initially, the e number of URL features $\left(U_f\right)$ like length_url, ip, ratio_digits_host, and brand_in_path are extracted from U to accurately distinguish the PA.

$$U_f = \left\{U_{1f}, U_{2f}, \ldots\ldots, U_{ef}\right\} \tag{7}$$

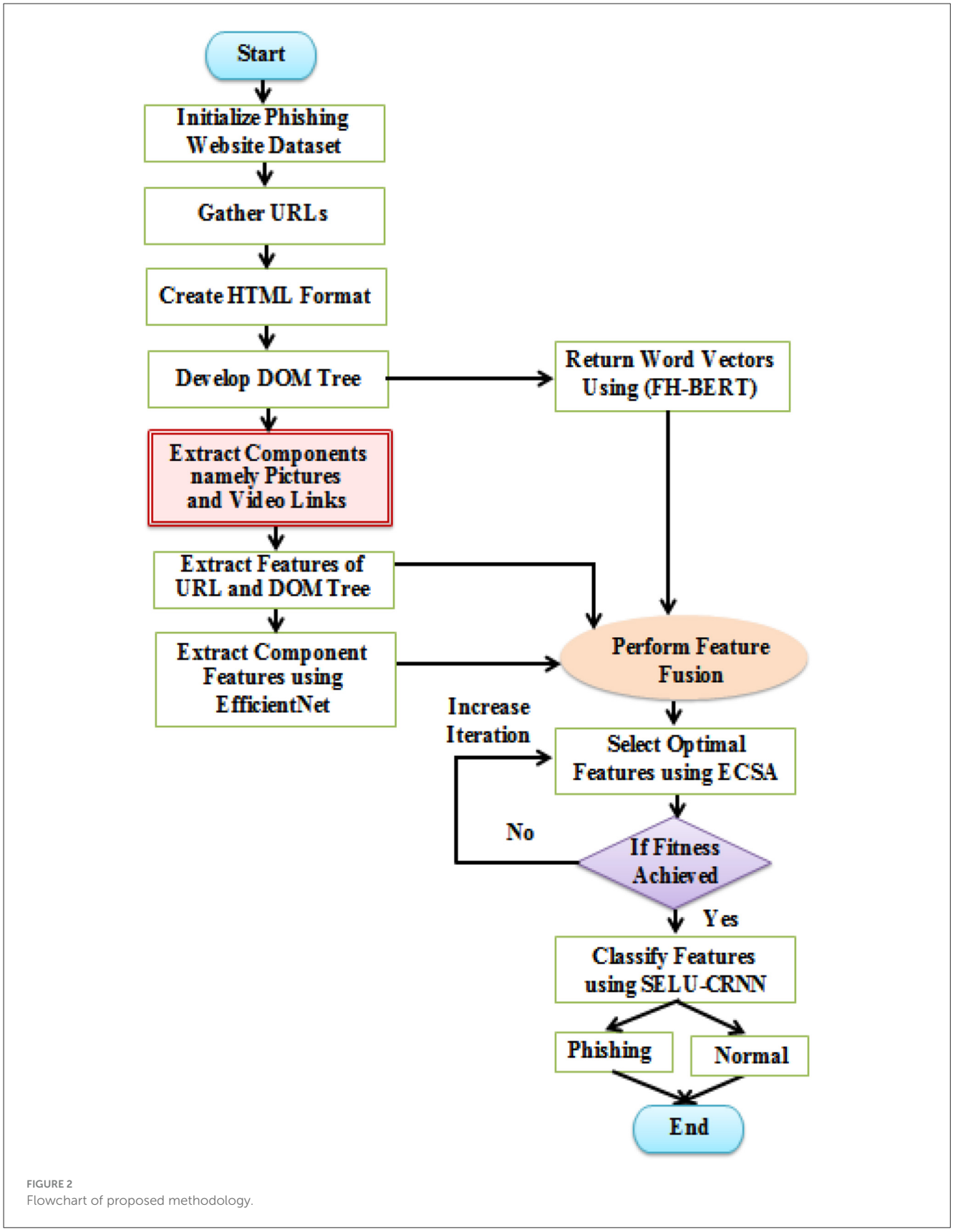

FIGURE 2
Flowchart of proposed methodology.

3.4.2 Component features

Next, the features from C are extracted utilizing EfficientNet. The EfficientNet is a pre-trained model that is trained by a large dataset for extracting video and image features. The EfficientNet (E) comprises 8 models within E0 and E7. It uniformly scaled the resolution (R), width (W), and depth (D) of the network with a

minimum number of parameters, thereby attaining better accuracy. The EfficientNet utilizes the swish activation function (ℓ) in every layer for retaining network information.

$$\ell(E) = E \times \nu(E) \tag{8}$$

Here, ν is the sigmoid function. In EfficientNet, primarily, the compound scaling method is utilized for grid search to evaluate the relationship among the network's various scaling dimensions with fixed resource constraints. Hence, the scaling factor from D, W, R is uniformly estimated in the below expressions with the compound coefficient (ϕ).

$$D = \Im^{\phi}, \quad \Im \geq 1 \tag{9}$$

$$W = \Re^{\phi}, \quad \Re \geq 1 \tag{10}$$

$$R = \aleph^{\phi}, \quad \aleph \geq 1 \tag{11}$$

Here, the constants that are determined by the small grid search are specified as $\Im$, $\Re$, $\aleph$ and ϕ signifies the user-defined coefficient. This equation is utilized to increase the Floating Point Operations Per Second (FLOPS) with only 2^{ϕ}.

$$\Im \cdot \Re^2 \cdot \aleph^2 \approx 2 \tag{12}$$

Beginning from the conventional EfficientNet-E0, the compound scaling technique is applied in two steps. Initially, fix the value of $\phi = 1$. It is assumed that the available resources are doubled, and the grid search is performed to find the best value of $\Im, \Re, \aleph$. $\Im, \Re, \aleph$ are fixed with constant values, and the baseline networks are scaled up with different ϕ to determine the EfficientNet-E1 to E7. Then, the feature maps f_M are formulated by each Convolutional block. Further, the f_M is moved to the pooling layer $\wp_{\circ}$, which compresses the spatial dimension of the feature map. Next, the features in $\wp_{\circ}$ are forwarded to the fully connected layer. Here, every neuron in the preceding layers is connected entirely with the corresponding neurons in the succeeding layers. At the output of the final Convolutional block, the component features (C_f) are obtained and are extracted as E.

3.4.3 DOM tree similarity features

Afterwards, the features that represent the hierarchical structure of the webpage elements, such as NLP (N_f) features, credential (CR_f) features, and DOM tree similarity (DT_f) features are extracted from for further analysis. Here, the DT_f features, namely the Shallow features, Deep features, and Tree Mapping Kernels features are extracted from DT to distinguish the features among each ι.

$$DT_f = \{N_f, CR_f, DT_f\} \tag{13}$$

Thus, various critical features that contribute to the accurate PWD, including U_f, $E\,B_f$ are extracted from the URL data, DOM, and its components.

3.5 Word embedding

For the word analysis DT, the word embedding ($\wp$) is utilized. The $\wp$ process is encoded with the real-valued vector for the DT words, and it is achieved by FH-BERT. The Bidirectional Encoder Representations from Transformers (BERT) is employed for its ability to train with a huge text collection, improving the semantic representation. Nevertheless, the BERT faced a problem in understanding the position control of the words due to the fixed positional embeddings. Therefore, the Feature Hasher technique is utilized for its space-efficient method of vectorizing features.

1. *BERT Architecture*: The BERT encompasses two model sizes, namely BERT Base and BERT Large, which are described below based on their certain encoders ($^{\circ}$), attention heads (Ξ), and millions of parameters (υ).

$$BERT_{Base} \rightarrow \{12^{\circ}, 12\Xi, 110\upsilon\} \tag{14}$$

$$BERT_{L\arg e} \rightarrow \{24^{\circ}, 16\Xi, 340\upsilon\} \tag{15}$$

2. *Text Processing*: Here, the embedding processes (position, segment, and token) are done.

(i) *Token Embedding*: Here, $[cls]$ indicates a classifying token that is added to the input word at the beginning of the sentence. Then, a separate token $[sep]$ is inserted at the end of the sentences to separate the various aspects of the tokens.

$$\wp_{tok} = \{\wp_{[cls]}, \wp_{11}, \wp_{12}, \wp_{[sep]}, \wp_{21}, \wp_{22}, \wp_{[sep]}\} \tag{16}$$

Here, $\wp_{tok}$ is the token embedding, $\wp_{11}$, $\wp_{12}$ specify the words in the first sentence, and $\wp_{21}, \wp_{22}$ imply the words in the second sentence.

(ii) *Segment Embedding*: Thereafter, an indicator 1 *and* 2 is added with each word of the sentences to distinguish them from the original text regarding punctuation or spaces.

$$\wp_{seg} = \{\wp_1, \wp_1, \wp_1, \wp_1, \wp_2, \wp_2, \wp_2\} \tag{17}$$

(iii) *Position Embedding*: The BERT learned and utilized the positional embeddings ($\wp_{pos}$) to estimate the position of the words in the sentence. But, the BERT could not efficiently manage the position control of the words to obtain the position information. Hence, a Feature Hasher technique is used, which hashed the categorical features of the words into fixed-sized vectors.

$$F_{\wp} = \kappa\left|\wp_{seg}\right| \tag{18}$$

Where, $F_{\wp}$ denotes the hashed outcome of the Feature Hasher technique and represents the hash function. Here, K creates the hash code and is mapped over the $\wp_{seg}$ through the modulo arithmetic operation to produce the fixed-sized vectors. Thus, the embedding of the words is represented as,

$$\wp_{pos} = \{\wp_0, \wp_1, \wp_2, \wp_3, \wp_4, \wp_5, \wp_6, \wp_7\} \tag{19}$$

$$\wp = (\wp_{tok} + \wp_{seg} + \wp_{pos}) \tag{20}$$

3. *Pre-training Tasks*: After processing the text, the training is done concurrently with two tasks: (1) MLM and (2) sentence prediction. In Mask Language Modeling (MLM), the pre-processed text sequences are randomly chosen and are masked with mask tokens. Next, during the sentence prediction task, the original vocabulary and vector values of the masked tokens are predicted by the FH-BERT model by producing a probability distribution of the overall vocabulary for the masked tokens. Afterward, the trained model gives the vector values ϕ for the tokens of DT strings, followed by a prediction.

3.6 Feature fusion

Here, for better classification of PWD, the extracted features in U_f and C_f and the transformed vectors (φ) for the strings in DT are concatenated.

$$\partial = U_f \oplus C_f \oplus \varphi \tag{21}$$

Here, $\oplus$ represents the fusion operation and ∂ denotes the fused features.

3.7 Feature selection

Here, the optimal features are selected from ∂ using ECSA for enhancing the classification accuracy. The conventional CSA was employed for its ability to locate the prey's position with the special rotational feature. Yet, there is an issue with its linear position update, leading to premature convergence. Hence, the Entropy technique is employed by effectively handling the weight determination process during position updates.

Initialization Phase: Here, the chameleon population (P) (fused features) is initialized.

$$P_m^n = \left\{P_1^1, P_2^2, ..., P_S^T\right\} \quad m = 1, 2, ..., S;\ n = 1, 2, ..., T \tag{22}$$

Here, S epitomizes the number of iterations, T exemplifies the number of chameleons' positions, and P_m^n specifies the n^{th} chameleon's position at m^{th} iteration. Thereafter, the initial position (optimal features) of the n^{th} chameleon is estimated by the upper bound (u) and lower bound (l) in the search place dimension (d) with a random number (r) within the range of [0,1]. The initialized position of the chameleon is P_d^n represented as,

$$P_d^n = L_d^n + r \times \left(u_d^n - l_d^n\right) \tag{23}$$

Fitness Evaluation: Next, the fitness function (τ) of the ECSA is calculated regarding the high classification accuracy (Ξ) to attain the optimal solution for feature selection. It is epitomized as,

$$\tau = \max\left(\Xi\right) \tag{24}$$

Prey Tracking Phase: For reaching the optimal features, the chameleon movement is estimated based on during the exploration phase with the position updation strategy.

$$P_{m+1,d}^n = \begin{cases} P_{m,d}^n + \varsigma_1 \left(B_{m,d}^n - G_m^n\right) r_2 + \varsigma_2 \left(G_m^n - P_{m,d}^n\right) r_1, & if\ \tau \geq \varsigma \\ P_{m,d}^n + \varpi_m \left(\left(u_m^n - l_m^n\right) r_3 + l_d\right) \left(sg\right)(r - 0.5), & if\ \tau < \varsigma \end{cases} \tag{25}$$

Here, $m+1$ specifies the next iterations, $P_{m,d}^n$, $P_{m+1,d}^n$ symbolize the current position and the new chameleon's position, respectively, $B_{m,d}^n$, G_m^n are the chameleon's best and global position, $r = r_1, r_2, r_3$ are the random numbers, ς is the perceiving probability of the chameleon perceiving prey, ς_1, ς_2 are two positive probabilities that control the ability of prey exploration, ϖ_m is the iteration parameter function utilized for reducing the number of iterations, sg is the sign function, and $r-0.5$ means random numbers between -0.5 *to* 0.5, which has an effect on exploration and exploitation directions. The B *and* G joined by a line (li). The chameleon's arbitrary position (ar) in li with B *and* G is specified as,

$$li\left(r\right) = rB + \left(1 - r\right) G \tag{26}$$

$$ar\left(r\right) = r_1 \cdot li + \left(1 - r_2\right) P_{m,d}^n \tag{27}$$

Hence, the prey is traced by the chameleon through its movement.

Eye Rotation Feature of Chameleon: After tracking the prey, the chameleon utilizes its special rotational eye feature to perceive the prey's location. The chameleon's position after updation $\left(P_{m+1}^n\right)$ is transformed to the center of gravity position $\left(\overline{P}_m^n\right)$ (origin) with the rotational center coordinates $(\hbar_m^n)$. It is given as,

$$P_{m+1}^p = \hbar_m^n + \overline{P}_m^n \tag{28}$$

Then, the rotation matrix ($\hbar$) is estimated along with the centering rotation matrix coordinates $\left(\hbar c_m^n\right)$ for identifying the position of the prey. It is expressed as,

$$\hbar_m^n = \hbar \times \hbar c_m^n \tag{29}$$

Next, the chameleon's positions are updated by utilizing $\hbar$ at $\overline{P}_m^n$. If the fitness is satisfied at the updated position, the further exploration process is carried out. Otherwise, the chameleon returns to its original position, and the position updation is iterated until the fitness function is fulfilled.

Hunting Prey: After updating the chameleon's position, the chameleon starts hunting if the prey's location is so close to it. The velocity (ϑ) of the chameleon's tongue to catch the prey with its inertia weight parameter (κ) is evaluated and is represented as,

$$\vartheta_{m+1}^n = \kappa \vartheta_m^n + \Gamma_1 \left(G_m^n - \overline{P}_{m,d}^n\right) r_1 + \Gamma_2 \left(B_{m,d}^n - \overline{P}_{m,d}^n\right) r_2 \tag{30}$$

Let ϑ_m^n, ϑ_{m+1}^n be the current and new velocities of the chameleon, Γ_1, Γ_2 be two positive constants that influence $B_{m,d}^n, G_m^n$

on dropping of the chameleon's tongue, and $\overline{P}^{n}_{m,d}$ be the position mean of the chameleon. When projected toward prey, the tongue position of the chameleon represents the chameleon's position with the acceleration rate (Ω) and the speed ($\Im$).

$$P^{n}_{m+1,d} = p^{n}_{m,d} + \frac{\log\left(\left(\Im^{n}_{m,d}\right)^{2} - \left(\Im^{n}_{m-1,d}\right)^{2}\right)}{2\Omega} \tag{31}$$

Here, $\Im^{n}_{m,d}, \Im^{n}_{m-1,d}$ signifies the current and the previous speed of n^{th} the chameleon. The Ω of chameleon's tongue projection gradually increases until it reaches the prey. It is the maximum value with the exponential (e) of the inverse log function of m, and it is expressed as,

$$\Omega = \Im\left(1 - e^{\left(\frac{1}{\log m}\right)}\right) \tag{32}$$

Therefore, the fastest chameleon that hunts the prey by satisfying the fitness function is determined as the best chameleon. Thus, the required optimal features are selected from the ECSA, and it is notated as ∂_{opt}. The pseudo-code of ECSA is presented below.

Pseudo code for ECSA

Input: Fused features(∂)
Output: Optimal features $\left(\partial_{opt}\right)$
Begin
Initialize Chameleon population(P), m, maximum iteration $m_{\max}$, u and l of d.
Set initial iteration
While $m \le m_{\max}$
For each P position **do**
Estimate initial position, $P^{n}_{d} = L^{n}_{d} + r \times \left(u^{n}_{d} - l^{n}_{d}\right)$
Compute fitness function, $\tau = \max(\Xi)$
Evaluate new position of P
If $(\tau \ge \varsigma)\{$
Update new position
$\}$ **Else** $\{$
Remains in previous position
$\}$
End if
Compute new position of P, $ar(r) = r_1 \cdot li + (1 - r_2) P^{n}_{m,d}$
Define inertia weight (κ), acceleration rate (Ω) and chameleon speed ($\Im$)
Compute the velocity of chameleon's tongue for hunting prey

$$\vartheta^{n}_{m+1} = \kappa \vartheta^{n}_{m} + \Gamma_1\left(G^{n}_{m} - \overline{P}^{n}_{m,d}\right) r_1 + \Gamma_2\left(B^{n}_{m,d} - \overline{P}^{n}_{m,d}\right) r_2$$

Estimate chameleon position,

$$\mathrm{P^{n}_{m+1,d}} = \mathrm{p^{n}_{m,d}} + \frac{\log\left(\left(\Im^{n}_{m,d}\right)^{2} - \left(\Im^{n}_{m-1,d}\right)^{2}\right)}{2\Omega}$$

Evaluate tongue projection of chameleon

$$\Omega = \Im\left(1 - e^{\left(\frac{1}{\log m}\right)}\right)$$

End for
$m = m + 1$
End while
Return ∂_{opt}
End

Hence, the proposed ECSA efficiently selected the optimal features.

3.9 Classification

Thereafter, by utilizing the SELU-CRNN, the optimal features ∂_{opt} are given to the classifier for PWD. The Convolutional Recurrent Neural Network (CRNN) is utilized for its ability to capture both the spatial and temporal features for the classification. However, the activation in the CRNN architecture makes the weight updation process weak by producing output values of more neurons as 0. Hence, Softplus Exponential Linear Unit (SELU) activation is employed to quickly converge the network with its speed internal normalization. Figure 3 displays the architecture of the SELU-based hybrid CNN and RNN.

3.9.1 Input layer

The input layer includes ∂_{opt}, which are transferred to the CNN layer for phishing detection.

Convolutional Layer: Then, to convolve the input features, a multi-scaled convolutional kernel (Ψ) with height (h) is utilized for the accurate segmentation. In the feature convolution, local features in the input are evaluated by utilizing Ψ and are given as,

$$\chi = \frac{\left(\chi_{\partial_{opt}} - \Psi_h + 2p'\right) + B}{S + 1} \tag{33}$$

Here, χ symbolizes the convolution layer output, $\chi_{\partial_{opt}}$ epitomizes the input of CNN, p' specifies the size of padding, S is the stride, and B is the bias factor. After that, the j number of convolved outputs (χ_1) is expressed as,

$$\chi_1 = \left\{\chi(1), \chi(2), ..., \chi(j)\right\} \tag{34}$$

Activation Function: Further, the input scale (δ) and weight (g) are given to the SELU (γ) activation function to activate the neurons. It is defined as,

$$\gamma = \begin{cases} \log(1 + e^{g}), & if\ g > 0 \\ \delta(e^{g} - 1), & if\ g \le 0 \end{cases} \tag{35}$$

Pooling Layer: Next, χ_1 the is transferred to the max pooling layer ($\eta_{\max}$), and the important features are extracted for better classification of PWD. The size of $\eta_{\max}$ is changed according to

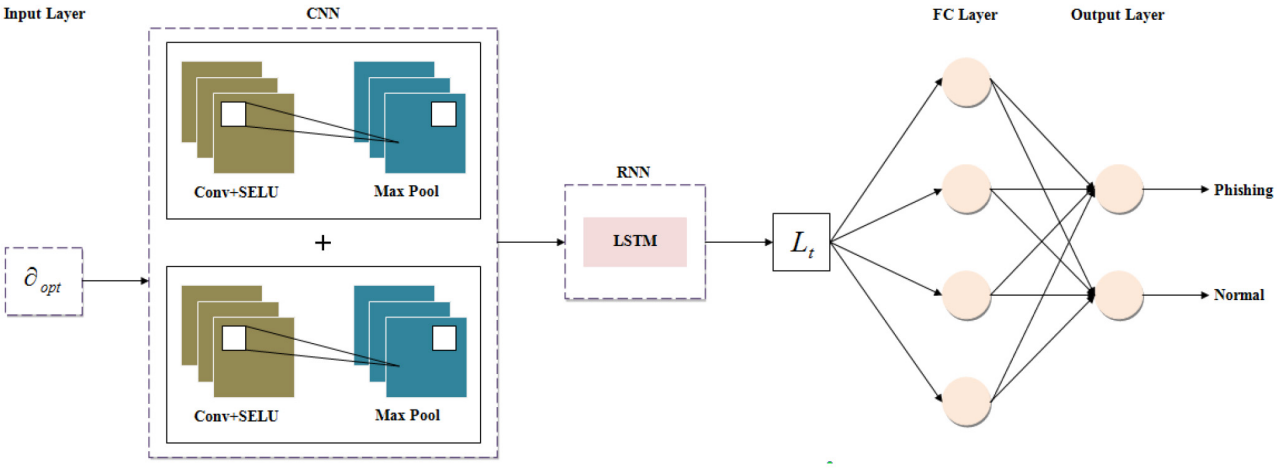

FIGURE 3
Architecture of the proposed SELU-CRNN.

the input features' size. The output of the max pooling layer η_{out} is represented as,

$$\eta_{out} = \eta_{\max}(\chi_1) \tag{36}$$

CNN Output: Afterward, the CNN output η is obtained by fusing features η_{out1}, η_{out2} of two max-pooling layers, and it is given as the input to the Recurrent Neural Network (RNN). The fused output (η) is presented as,

$$\eta = \eta_{out1} + \eta_{out2} \tag{37}$$

3.9.2 RNN structure

Next, η is fed to the LSTM. Here, the LSTM architecture is selected for its superior memory capability, allowing it to effectively capture long-term dependencies. The LSTM's layers are the forget layer (F_t), input layer (I_t), cell state (α_t), and output layer (O_t) with time (t). At first, F_t will remove the unused memory from the cell state, and it is expressed as,

$$F_t = \sigma\left(W_F \cdot \lfloor \eta_{t-1}, \eta_t \rfloor + B_F\right) \tag{38}$$

Here, W_F is the weight matrix of F_t, σ signifies the sigmoid function, η_{t-1}, η_t epitomizes the input from the pooled output at t and $t-1$, and B_F is the bias of F_t. Next, the useful information about the cell state is added with the I_t weight matrix (W_I) and bias factor B_I, and it is specified as,

$$I_t = \sigma\left(W_I \cdot \lfloor \eta_{t-1}, \eta_t \rfloor + B_I\right) \tag{39}$$

Afterward, the cell state is represented with the function, and it is determined grounded on the last output and current input. It determines and captures the long-term dependencies of the sequential features with the weight matrix (W_α) and bias factor (B_α) for processing complex data patterns. The current cell state α_t is computed as,

$$\ddot{\alpha}_t = \tanh\eta\left(W_\alpha \cdot \lfloor \eta_{t-1}, \eta_t \rfloor + B_\alpha\right) \tag{40}$$

$$\alpha_t = F_t \times \alpha_{t-1} + I_t \times \alpha_t \tag{41}$$

Here, $\ddot{\alpha}_t$ indicates the candidate cell state, (α_{t-1}) defines the cell state in the previous time, $(\tanh\eta)$ specifies the hyperbolic tangent function of (η). Lastly, the output layer O_t extracted the required information from the cell state, which is expressed below with its weight matrix (W_O) and bias factor (B_O).

$$O_t = \sigma\left(W_O \cdot \lfloor \eta_{t-1}, \eta_t \rfloor + B_O\right) \tag{42}$$

Then, the final output of the LSTM (L_t) is obtained based on the cell state and the output layer.

$$L_t = O_t \cdot \tanh\eta(\alpha_t) \tag{43}$$

Next, the LSTM output L_t is given to the fully connected layer for segmenting the features that have the output dimension grounded on the number of classes for the output. The probability of every single class is estimated by the softmax function (β) that is defined below,

$$\beta(L) = \frac{\exp(L)}{\sum_{H=1}^{q} \exp(L_H)} \tag{44}$$

Here, H epitomizes the output vector and symbolizes the standard exponential function. The $\beta(L)$ gives output classes as phishing or normal at the output layer of the proposed classifier. Hence, the phishing websites are effectively detected by the SELU-CRNN from the selected optimal features. The pseudo-code for the SELU-CRNN is depicted below.

Pseudo code for SELU-CRNN

Input: optimal features ∂_{opt}
Output: output classes $\beta\,(L)$
Begin
Initialize padding size p', stride, weight matrices W_F, W_I, W_α, W_0 and bias factors B_F, B_I, B_α, B_0, max (∂_{opt}).
While $\partial_{opt} \geq \max(\partial_{opt})$ **do**
For each ∂_{opt}
Compute convolutional function and max pooling function
Activate SELU activation function, $\gamma = \begin{cases} \log(1+e^g), & if\ g > 0 \\ \delta(e^g - 1), & if\ g \leq 0 \end{cases}$
Concatenate both max pooled outputs
Perform LSTM operation $L_t = O_t \cdot \tan\eta\,(\alpha_t)$,
Evaluate softmax, $\beta\,(L) = \frac{\exp(L)}{\sum_{H=1}^{q}\exp(L_H)}$
End for
End while
Return
End

Thus, the proposed SELU-CRNN efficiently classified the phishing websites.

4 Results and discussions

Here, the experimental outcomes for the proposed PWD system utilizing the ECSA and SELU-CRNN are presented. Also, the experiments are employed in the working platform of PYTHON.

4.1 Dataset description

For assessing the proposed PWD system's performance efficiency, the web page phishing detection dataset, Phishing URL dataset, PhishStorm dataset, and URL dataset are utilized. The Web page phishing detection dataset (Endnote 3) comprises 11,430 URLs and 87 different features. Then, the PhiUSIIL (Endnote 1) is used, which consists of 134,850 legitimate data and 100,945 phishing URL data. Further, the PhishStorm dataset (Endnote 2) comprised 96,018 URLs. Then, the the ISCXURL (Endnote 4) 2016 is utilized, which includes 35,300 benign and 10,000 phishing URLs. From each of these datasets, 80% and 20% are gathered for training and testing purposes, correspondingly.

Figure 4 represents the splitting of the dataset based on the K-fold method for training and validating the proposed technique.

4.2 Performance analysis of the proposed SELU-CRNN

Here, to show the efficiencies of the proposed SELU-CRNN in the PWD process, the evaluations of the proposed SELU-CRNN are

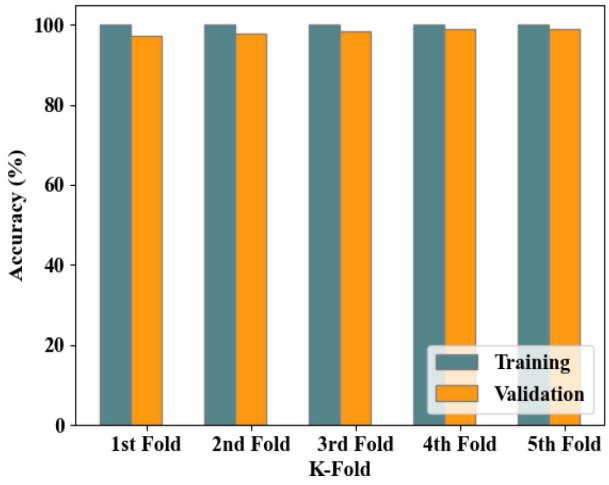

FIGURE 4
Graphical representation of cross-validation.

TABLE 1 Parameter settings of SELU-CRNN.

Training parameter	Value
Batch size and learning rate	32 and 0.001
Pooling type	Max pooling
Time step	7
Activation function of CNN	SELU activation function
Activation function of RNN	(Sigmoid, hyperbolic tangent function)
Number of output layers	2
Maximum epochs	100
Dropout rate	0.5

established. The configuration of the proposed SELU-CRNN model is given in Table 1.

Here, the batch size is tuned to enhance the stability of the model training. Further, the learning rate is defined to improve the learning efficiency. Subsequently, the dropout rate is fixed to rescue the model from being trapped in the overfitting problem. Moreover, the activation function is set to learn the complicated non-linear feature relationship of the model.

Figure 5 presents the comparison analysis of the SELU-CRNN with the prevailing RNN, LSTM, CNN, and DBN. The proposed SELU-CRNN attained 98.43% accuracy, 99.09% precision, 98.98% recall, 99.04% F-measure, 98.98% sensitivity, and 95.92% specificity. These effective metric values are attained as a result of the modified SELU with the hybrid CRNN since it takes both the spatial and the temporal features for the classification process. As these parameters are not improved in the prevailing works, the average metric values of accuracy (91.52%), precision (94.86%), recall (91.09%), F-measure (92.91%), sensitivity (91.09%), and specificity (90.33%) were achieved. Hence, the efficiency of the proposed SELU-CRNN is demonstrated.

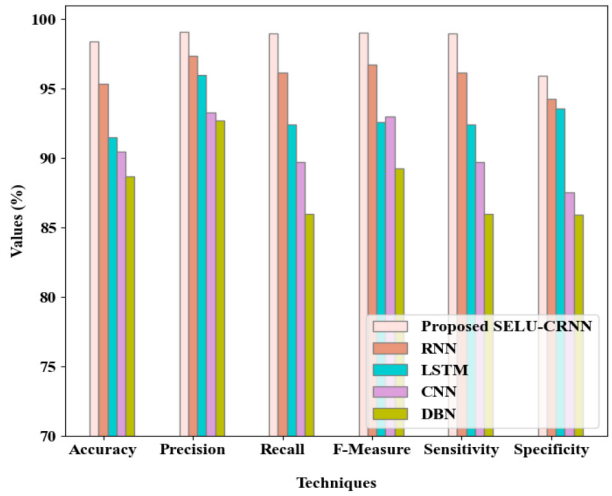

FIGURE 5
Comparison analysis of the proposed SELU-CRNN.

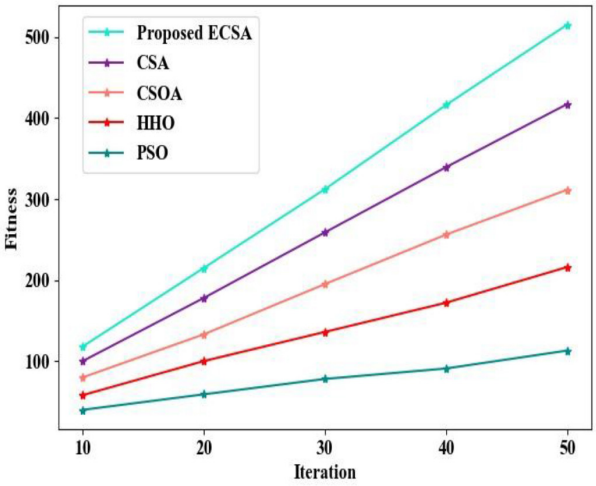

FIGURE 6
Fitness vs. Iteration.

TABLE 2 Time analysis for the process of PWD.

Process	Time (ms)
Dom tree creation	4,188
Credential features extraction	2,373
NLP features extraction	2,392

The time analysis for the processes, such as the Dom tree creation, credential features extraction, and the NLP features extraction of the PWD system, is explained in Table 2. The time taken for the dom tree creation, credential feature extraction, and NLP feature extraction is 4,188, 2,373, and 2,392 ms, respectively. Thus, the efficient classification process of the proposed PWD system is aided by time analysis.

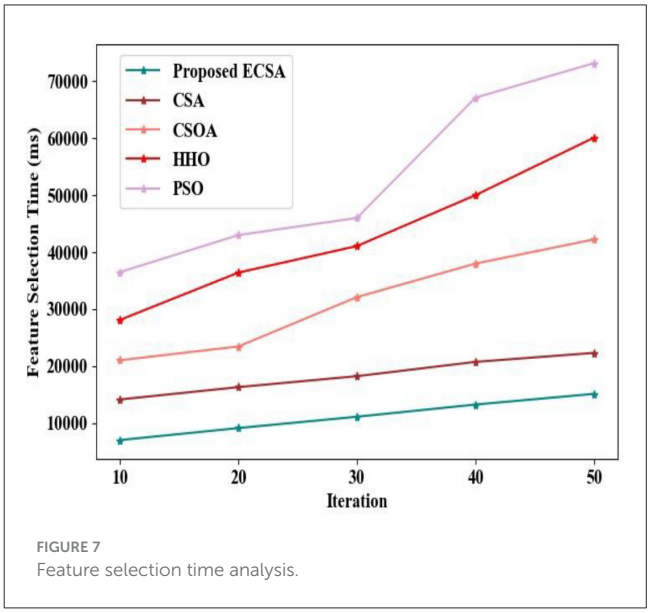

FIGURE 7
Feature selection time analysis.

4.3 Performance analysis of the proposed ECSA

Here, for validating the proposed ECSA's efficiency, the evaluation of the performance of the proposed ECSA was done and analogized with the prevailing Chameleon Swarm Algorithm (CSA), Crow Search Optimization Algorithm (CSOA), Harris Hawks Optimization (HHO), and PSO, grounded on the fitness vs. iteration and the feature selection time.

Figure 6 displays the different fitness values achieved at different iterations for the proposed ECSA and the other prevailing approaches. At a minimum of 10 iterations, the proposed ECSA achieved a fitness of 118.256. For 10 iterations, the prevailing approaches attained the fitness of 100.256, 80.125, 58.235, and 40.125 for CSA, CSOA, HHO, and PSO, respectively. As shown in Figure 6, the proposed ECSA achieves the highest fitness value compared to all other prevailing approaches. This improvement is attributed to the integration of the Entropy technique, which prevents premature convergence. Hence, it optimally chooses the features from the search space with effective weight updation in the selection process.

Figure 7 presents the comparative analysis of the time taken to select the optimal features by the proposed ECSA and the prevailing approaches. The proposed system underwent feature fusion of component features and word-embedded DOM structure prior to the feature selection process. The time taken for the proposed ECSA for selecting the features in a maximum of 50 iterations is 36,515 ms. However, the prevailing approaches randomly updated the position for the optimal solution, thus degrading their exploration ability. Hence, the existing algorithms, such as CSA, CSOA, HHO, and PSO, attained 43,012, 46,025, 67,145, and 73,214 ms of feature selection time in a maximum of 50 iterations, respectively. Hence, the lower time is attained by the proposed ECSA in the feature selection process.

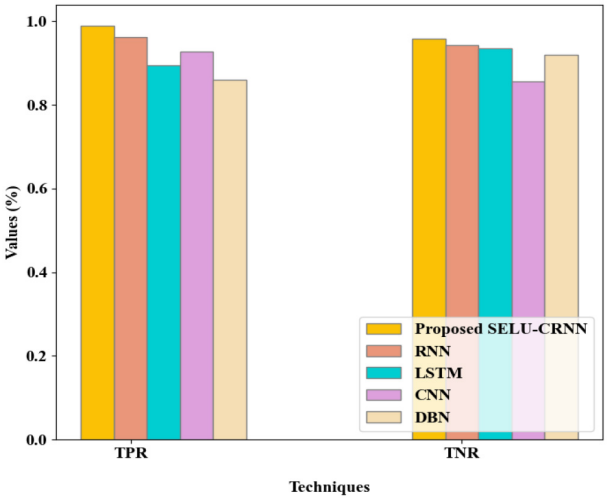

FIGURE 8
Comparative analysis of the proposed SELU-CRNN.

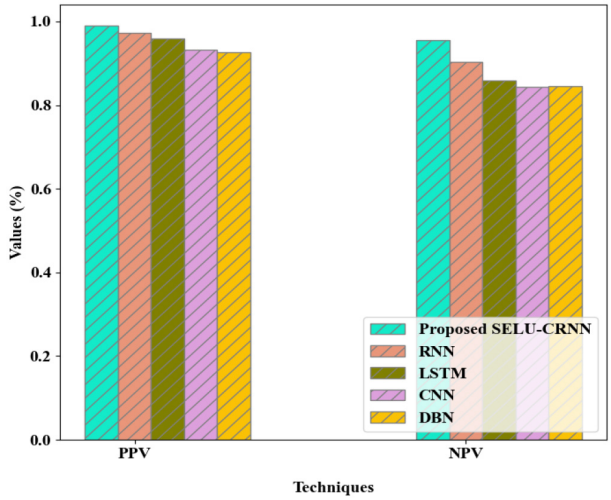

FIGURE 9
PPV and NPV analysis of the proposed SELU-CRNN and existing systems.

TABLE 3 Performance analysis of the proposed SELU-CRNN based on FPR and FNR.

Metrics	Proposed SELU-CRNN	RNN	LSTM	CNN	DBN
FPR (%)	0.040767	0.57504	0.64393	0.074444	0.080614
FNR (%)	0.010166	0.038205	0.054131	0.72797	0.099871
Training time (ms)	63,003	68,007	72,010	77,009	83,016

4.4 Comparative analysis of the performance of the proposed SELU-CRNN

Here, to exhibit the productivity of the proposed approach over the prevailing methods, the comparative assessment of the proposed SELU-CRNN and the prevailing RNN, LSTM, CNN, and DBN, grounded on the True Negative Rate (TNR), FNR, Positive Predictive Value (PPV), Training time, True Positive Rate (TPR), False Positive Rate (FPR), and NPV, is performed.

In Figure 8, the comparison of TPR and TNR of the proposed SELU-CRNN and the prevailing approaches is depicted. The proposed SELU-CRNN achieves the TPR and TNR of 0.989% and 0.959%, correspondingly. These true classification rates are achieved by the modification made with the SELU-CRNN technique that improves the classification process by considering the optimum features. So, the more accurate prediction is made by the proposed model, which resulted in improved detection of Phishing websites with higher TPR and TNR. The existing systems attained the TPR and TNR of 0.96% and 0.94% in RNN, 0.89% and 0.94% in LSTM, 0.93% and 0.856% in CNN, and 0.86% and 0.92% in DBN, correspondingly. Therefore, when analogized with the prevailing approaches, the highest true classification rate is attained by the proposed SELU-CRNN.

In Table 3, the performance assessment of the proposed SELU-CRNN and the prevailing systems, grounded on FPR, False Negative Rate (FNR), and training time, is presented. Here, the FPR and FNR exhibit the misclassified positive data classes and the incorrect classification of negative data classes by the proposed classifier, respectively. The proposed SELU-CRNN achieves an FPR and FNR of 0.040767% and 0.010166%, correspondingly. The existing systems attained an FPR and FNR of 0.57504% and 0.038205%, 0.64393% and 0.054131%, 0.074444% and 0.72797%, and 0.080614% and 0.099871% for RNN, LSTM, CNN, and DBN, correspondingly. The Training Time (TT) for the proposed SELU-CRNN is 63,003 ms, and the prevailing approaches like RNN, LSTM, CNN, and DBN consumed 68,007, 72,010, 77,009, and 83,016 ms, respectively. Thus, the lowest false classification rate and minimum TT are attained in the proposed SELU-CRNN.

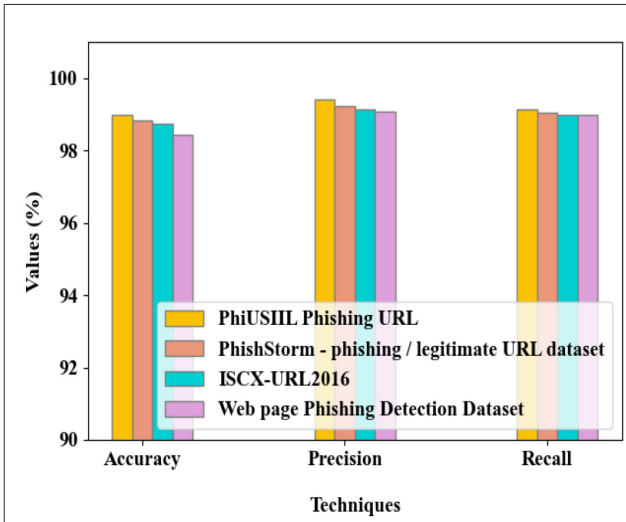

FIGURE 10
Evaluation of the proposed model over different datasets.

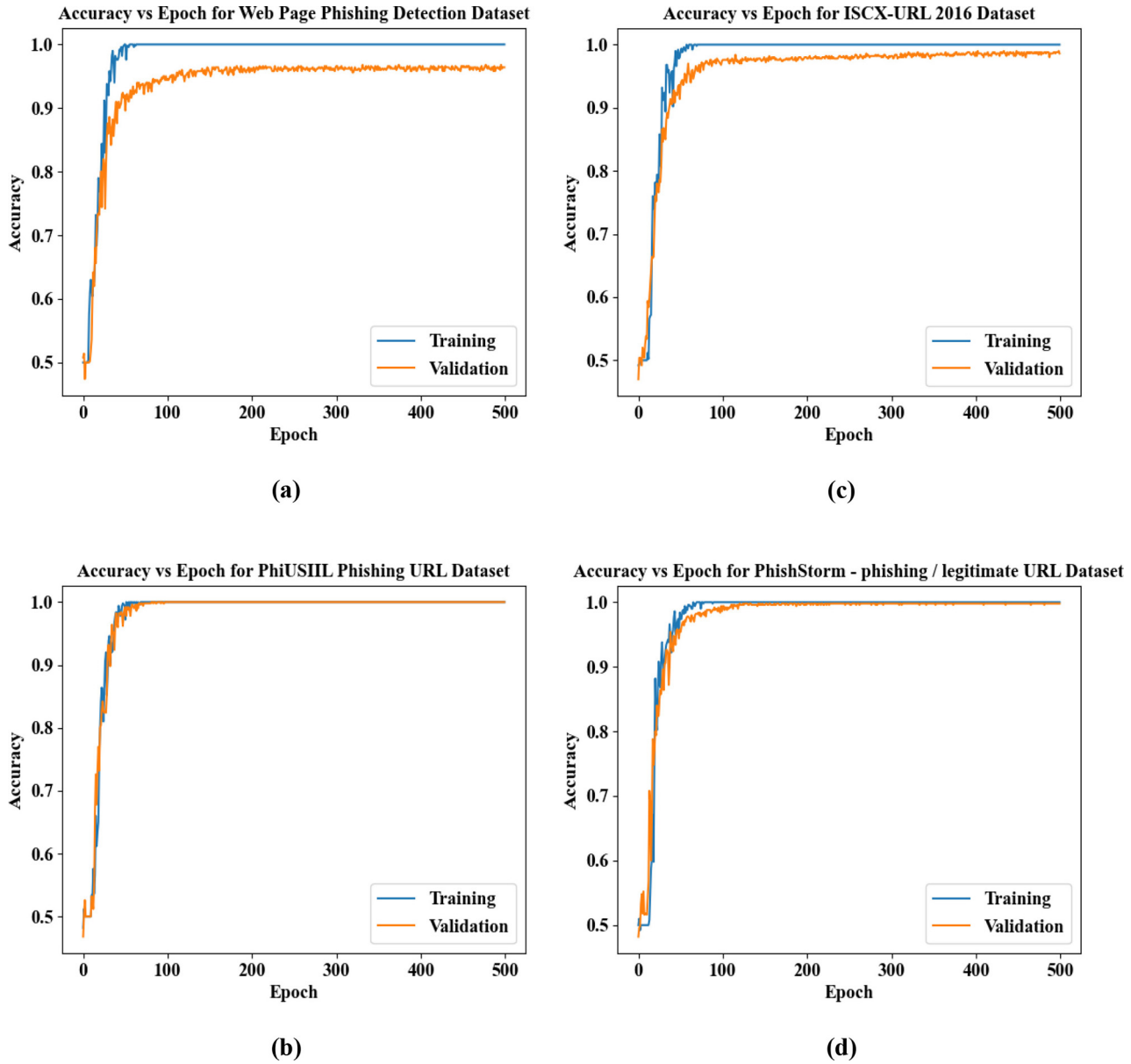

FIGURE 11
Analysis of accuracy vs. epochs for benchmark datasets.

In Figure 9, the comparison analysis of the proposed SELU-CRNN with the prevailing techniques grounded on the PPV and NPV metrics is depicted. The proposed SELU-CRNN achieves the PPV and Negative Predictive Value (NPV) of 0.990894% and 0.954654%, correspondingly. Meantime, the existing methods attained the PPV of 0.973603%, 0.960245%, 0.933162%, and 0.92721%, and NPV of 0.90411%, 0.85931%, 0.84386%, and 0.84629%. Therefore, owing to the proposed method's ability to identify the input features at any length and decrease unwanted parameter usage in the network, it achieved a high prediction rate.

The performance of the proposed SELU-SCRNN is evaluated in Figure 10 for the different datasets in terms of accuracy, precision, and recall. The proposed model attained nearly 99% accuracy, precision, and recall for all the utilized datasets. Here, the detection performance regarding accuracy, precision, and recall is similar among the different datasets. As the learning efficiency of the classifier is improved by integrating the CNN and RNN models with the SELU activation function, the performance of the proposed technique is better for the different datasets.

Figures 11a–d present the Accuracy vs. Epoch analysis for the proposed phishing detection framework evaluated on four different benchmark datasets: (a) Web Page Phishing Detection Dataset, (b) PhiUSIIL Phishing URL Dataset, (c) ISCX-URL 2016 Dataset, and (d) PhishStorm—Phishing/Legitimate URL Dataset. In each case, the proposed model shows a rapid rise in both training and validation accuracy during the initial epochs, with performance stabilizing near 1.0 after approximately 100 epochs. The minimal difference between training and validation curves across all datasets

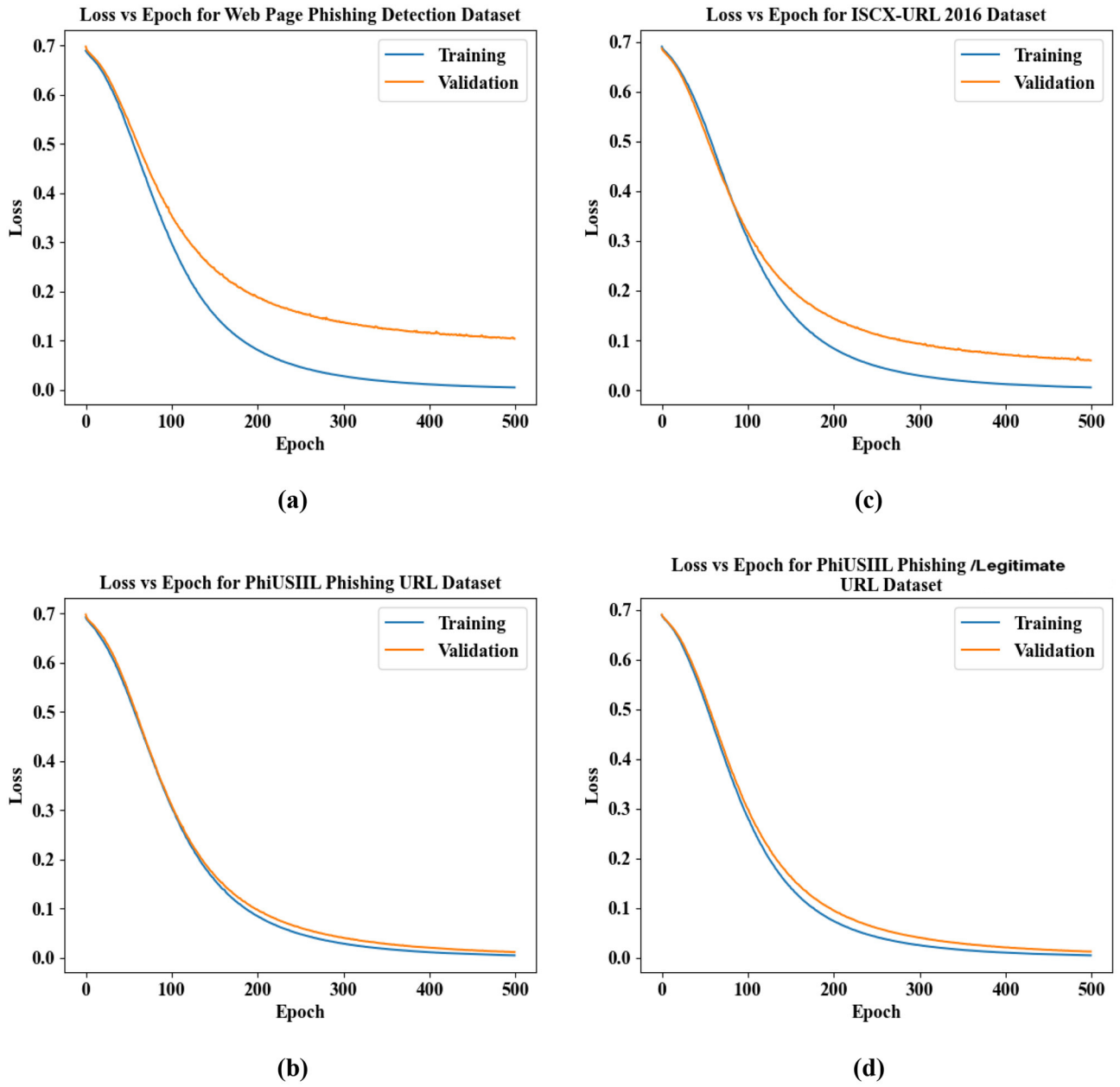

FIGURE 12
Analysis of loss vs. epochs for benchmark datasets.

indicates strong generalization capabilities. These results confirm the robustness of the proposed method in accurately detecting phishing websites across varied data sources.

Figures 12a–d illustrate the Loss vs. Epoch analysis for the proposed phishing detection framework across four benchmark datasets: (a) Web Page Phishing Detection Dataset, (b) PhiUSIIL Phishing URL Dataset, (c) ISCX-URL 2016 Dataset, and (d) PhishStorm—Phishing/Legitimate URL Dataset. In all cases, the training and validation loss curves show a consistent and steep decline from an initial value of approximately 0.7, rapidly decreasing below 0.1 within the first 150 epochs. The loss continues to approach zero as training progresses toward 500 epochs, with both training and validation curves closely aligned. This smooth convergence and the minimal gap between the curves indicate the model's stability and strong generalization capability, affirming the effectiveness of the proposed method in minimizing classification error across diverse phishing detection datasets.

The cross-dataset validation highlights the robustness and generalization capability of the proposed SELU-CRNN model. As shown in Table 4, when trained on the PhiUSIIL Phishing URL Dataset and tested on the PhishStorm—Phishing/Legitimate URL dataset, the model achieved high performance with 98.98% accuracy, 99.21% precision, 98.84% recall, and 99.23% F-measure, demonstrating its strong phishing detection ability even on unseen

TABLE 4 Evaluation of cross-dataset validation of training on PhiUSIIL phishing URL dataset and testing on PhishStorm—Phishing/Legitimate URL dataset.

Techniques	Precision (%)	Recall (%)	F-measure (%)	Accuracy (%)	Sensitivity (%)	Specificity (%)	TPR (%)	TNR (%)	PPV (%)	NPV (%)
Proposed SELU-CRNN	99.21	98.84	99.23	98.98	98.84	98.98	98.98	98.99	99.21	98.98
RNN	96.78	96.52	96.87	96.54	96.84	96.25	96.25	96.32	96.65	96.87
LSTM	93.94	94.78	93.65	93.65	94.78	93.62	93.54	93.52	94.54	93.65
CNN	91.26	91.78	90.63	90.65	91.54	90.78	90.23	90.54	91.45	90.65
DBN	88.74	88.65	88.45	88.65	87.96	88.74	87.32	87.45	87.85	88.45

TABLE 5 Efficacy analysis of cross-dataset on validation of training on ISCX-URL 2016 dataset and testing on web page phishing detection dataset.

Techniques	Precision (%)	Recall (%)	F-measure (%)	Accuracy (%)	Sensitivity (%)	Specificity (%)	TPR	TNR	PPV	NPV
Proposed SELU-CRNN	99.08	98.98	99.03	98.42	98.98	95.92	98.98	95.92	99.08	94.46
RNN	97.36	96.17	96.76	95.36	96.17	94.24	96.17	94.24	97.36	94.10
LSTM	96.02	92.45	92.62	91.52	92.45	93.56	92.45	93.56	96.02	85.93
CNN	93.31	89.72	93.01	94.63	89.72	87.55	89.72	87.55	93.31	84.38
DBN	92.72	86.01	89.24	88.71	86.01	85.93	86.01	85.93	92.72	84.62

data. Similarly, Table 5 presents the model's performance when it was trained on the ISCX-URL 2016 dataset and tested on the Web Page Phishing Detection dataset, where it achieved 98.42% accuracy, 99.08% precision, and 98.98% recall. These consistent results across datasets confirm that the proposed SELU-CRNN classifier maintains high detection efficiency in varying data distributions.

Figures 13a, b present the cross-dataset validation results for the proposed SELU-CRNN-based framework. In Figure 13a, the model trained on the PhiUSIIL Phishing URL Dataset is tested on the PhishStorm dataset, showing strong generalization. In Figure 13b, training on the ISCX-URL 2016 Dataset and testing on the Web Page Phishing Detection Dataset further confirms the model's robustness. In both cases, the proposed approach maintains high classification accuracy, demonstrating its effectiveness across heterogeneous phishing datasets.

Figure 14 illustrates the performance of the proposed SELU-CRNN in classifying the phishing and legitimate classes of the website data with respect to the confusion matrix. The higher number of phishing and legitimate classes recognized by the proposed model specifies the detection efficiency of the classifier amongst the different data classes.

Table 6 depicts the statistical performance of the proposed classifier in terms of MA and Var by comparing it with the existing models. Here, the MA and variance of the proposed and existing methods are determined for different numbers of epochs. The proposed method attained an MA of 98.43% and 97.64% for 100 and 60 epochs, respectively. Meanwhile, for 100 and 60 epochs, the existing RNN attained 97.56% and 95.71% and DBN attained 90.75% and 88.27% MA, respectively. Also, the abnormal or phishing of websites is detected more effectively by the proposed approach with higher MA and minimum variance. The existing techniques did not train with various important features, thus resulting in detection with lower MA and higher variance. Hence, the proposed model's performance is better than that of traditional networks.

The proposed model's performance for classifying phishing and legitimate website URLs is analyzed based on the *P*-value (Probability value) in Figure 15. The *p*-value determines the supremacy of the proposed technique based on the significance threshold value through a hypothesis test. As shown in the figure, the *p*-value of 0.050 is obtained, which is lower than the estimated threshold of 0.04. This depicts that the null hypothesis is effectively discarded by the proposed technique. Hence, the performance of the proposed model is statistically efficient.

Table 7 presents the proposed work's comparative analysis with the related works regarding accuracy, precision, recall, and FPR metrics. To classify the phishing sites, the prevailing works utilize a TWSVM, Deep Neural Network-Long Short-Term Memory (DNN-LSTM), ML, and ADAptive Moment (ADAM) estimation. Owing to the lack of consideration of the internal features of the websites, the prevailing Twin Support Vector Machine Classifier (TWSVM) achieved a recall of 98.33%. Since the efficiency of the system deviated from the occurrence of noisy instances, the DNN-LSTM, ML, and ADAM achieved an approximate accuracy of 97%. Then, as the system was processed only with a few URLs at a time, the CNN-LSTM grasped an FPR of 1.20%. Lastly, the proposed SELU-CRNN attained accuracy, precision, recall, and FPR of 98.43%, 99.095%, 98.98%, and 0.04%, respectively, which are considerably higher. Hence,

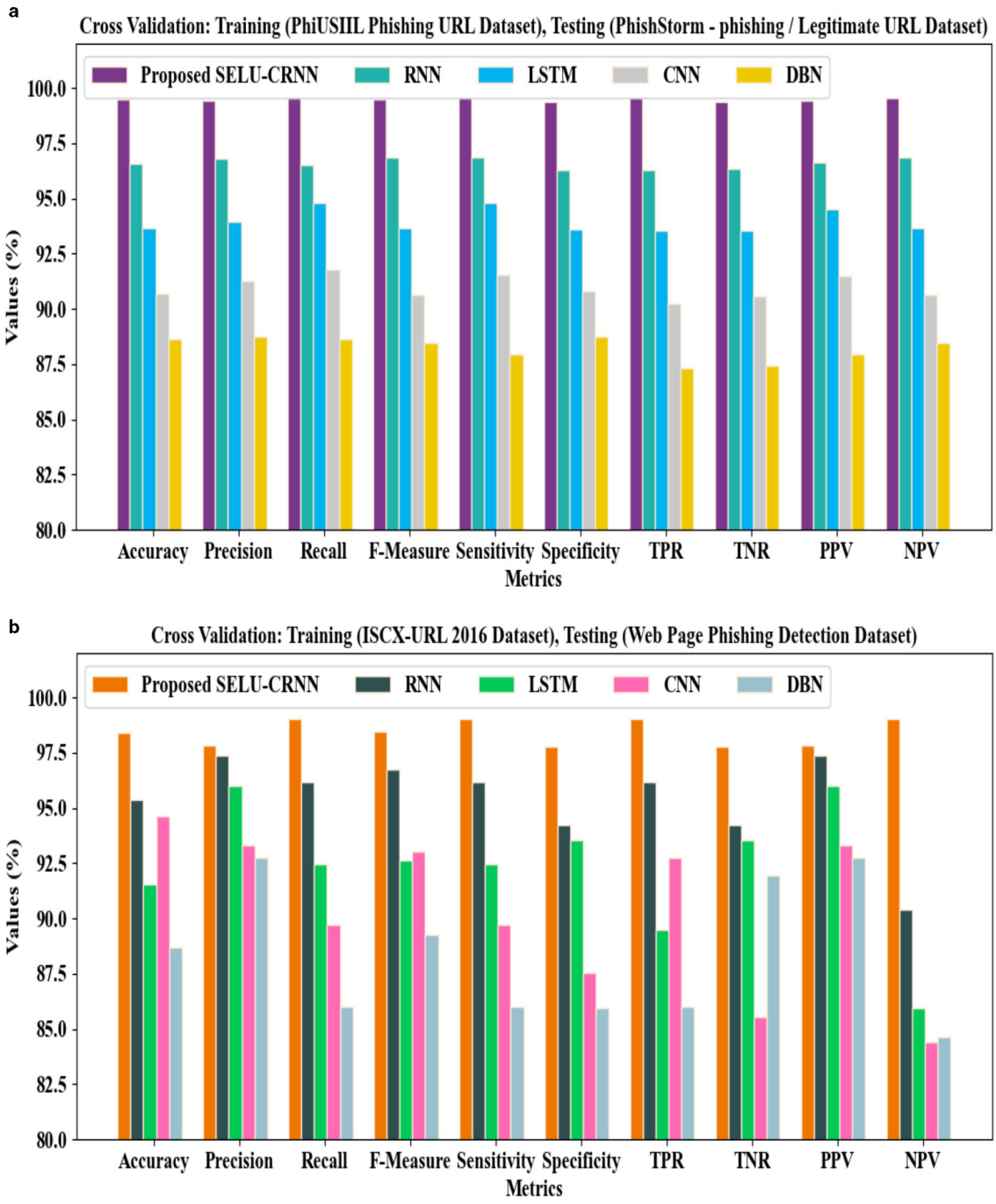

FIGURE 13
Analysis of cross-dataset validation regarding **(a)** validation of training on PhiUSIIL phishing URL dataset and testing on PhishStorm-phishing/legitimate URL dataset and **(b)** validation of training on ISCX-URL 2016 dataset and testing on web page phishing detection dataset.

effective outcomes are achieved by the proposed model in the PWD system.

The performance of the proposed technique is assessed in Table 8 by comparing it with the recent works that are related to the proposed approach. As per the experiment, the proposed SELU-CRNN attained 98.43% accuracy and 99.04% F-measure for the PWD. In the meantime, the existing techniques, such as NLP and DL algorithms in Benavides-Astudillo et al. (2023) attained 97.39% accuracy, and the Linear classifier-based multi-modal analysis of Castano et al. (2023) attained 96.80% F-measure. Subsequently, the Message Digest 5 (MD5) hash and DOM algorithm utilized by Colhak et al. (2024) attained 92.50% accuracy, and the Hybrid LSD algorithm, including Linear Regression (LR), Support Vector Classifier (SVC), and Decision Tree, attained

95.23% accuracy and 95.77% F-measure. Thus, the proposed model made a deep analysis of the URLs, HTML contents, and website content features for the detection of phishing websites. Therefore, the proposed model provides more effective PWD than the other related approaches.

4.5 Real-world applicability and challenges of the proposed methodology

As the proposed PWD model identified the phishing attack on websites with improved performance, it is well-suited for real-world applications. The deep learning-based proposed model can be embedded in various domains, such as financial institutions, information technology organizations, and public platforms. By deeply analyzing and learning the patterns of the website URLs, any harmful malicious actions on the website are detected in advance using the proposed model, thus protecting the website from PA. These are exposed on the website due to phishing, which is analyzed and detected by the proposed model. So, the vulnerability is prevented by displaying warnings and enabling website security with two-factor authentication. Although the proposed model can be executed for real-time scenarios, some limitations restrict its practical applicability. The proposed Deep learning-based detection model required more resources for deployment. Furthermore, the user must have advanced knowledge regarding machine learning and deep learning-based algorithms to relish the models' advantages. The embedded system that holds the detection model should have a larger storage capacity and a super-effective processor. Also, the requirement for more resources causes higher computational costs and time for the practical deployment of the proposed approach.

5 Conclusion

This framework proposes an effective PWD system grounded on the SELU-CRNN along with the FH-BERT and EfficientNet. From the URLs and DOM tree structure, the important features are extracted, and the optimal features are selected by utilizing ECSA.

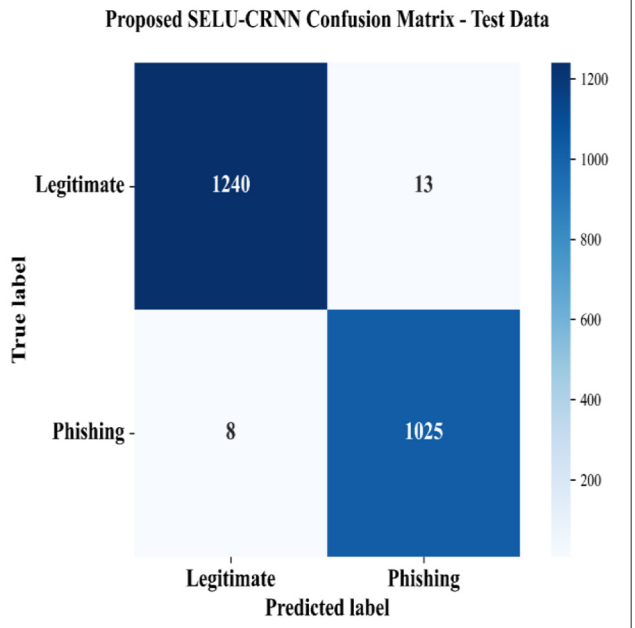

FIGURE 14
Confusion matrix of the proposed classifier.

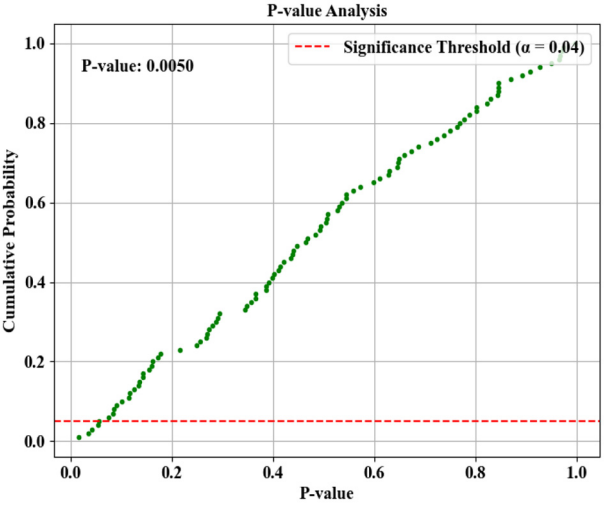

FIGURE 15
P-value analysis of proposed SELU-CRNN.

TABLE 6 Statistical analysis of the proposed SELU-CRNN.

Techniques	Number of epochs									
	20		40		60		80		100	
	Var	MA (%)	Var	MA (%)	Var	MA (%)	Var	MA (%)	Var	MA (%)
Proposed SELU-CRNN	0.33	95.53	0.25	96.51	0.22	97.64	0.10	98.07	0.04	98.43
RNN	0.44	93.05	0.37	94.18	0.32	95.71	0.18	96.55	0.15	97.56
LSTM	0.50	91.86	0.48	92.49	0.36	93.23	0.25	94.38	0.23	95.41
CNN	0.58	88.45	0.53	89.27	0.44	91.39	0.30	92.81	0.31	94.16
DBN	0.65	87.92	0.61	88.16	0.52	88.27	0.39	89.03	0.36	90.75

TABLE 7 Comparative assessment with the related works.

Study	Methods	Datasets	Classification			
			Accuracy (%)	Precision (%)	Recall (%)	FPR (%)
Proposed	SELU-CRNN	Web page phishing detection + Phishing URL dataset + PhishStorm-Phishing/Legitimate URL dataset + URL dataset	98.43	99.090	98.98	0.04
Rao et al. (2021)	TWSVM	Phishtank and Alexa	98.05	–	98.33	–
Ozcan et al. (2023)	DNN–LSTM	Ebbu2017	97.79	–	–	–
Gupta et al. (2021)	Machine learning classifiers	ISCXURL-2016	97.57	–	–	–
Xiao et al. (2021)	CNN-LSTM	Phishtank	97.2	98.76	95.6	1.20
Lakshmi et al. (2021)	ADAM	Phishing website	98	–	–	–

TABLE 8 Comparative assessment of the proposed method with recent literature.

Authors	Technique used	Accuracy (%)	F-measure (%)
Proposed model	SELU-CRNN	98.43	99.04
Benavides-Astudillo et al. (2023)	NLP and DL algorithms	97.39	–
Castano et al. (2023)	Multi-modal analysis with a linear classifier	97.18	96.80
Colhak et al. (2024)	MD5 hash and DOM algorithms	92.50	–
Karim et al. (2023)	Hybrid LSD model	95.23	95.77

Lastly, the proposed SELU-CRNN classifies the phishing websites. For the optimal feature selection, the modified ECSA technique takes 7,015 ms. The proposed SELU-CRNN achieves classification accuracy, precision, recall, and F-measure of 98.43%, 99.09%, 98.98%, and 99.03%, respectively. The proposed work has the highest TPR (0.99%) and TNR (0.96%) values and the lowest FPR (0.40%) and FNR (0.10%) values. Owing to this approach's dynamic true and false prediction rates, the proposed model outperformed the prevailing works.

6 Limitations of the study and future recommendations

The proposed technique's training is only grounded on the data structure and does not concentrate on the actions given in the phishing URLs. The actions of the phishing URLs are predefined, which causes the user to reach an irrelevant or fake webpage and steal the user's information. This is because of the scripts behind the webpage links, such as redirecting to malicious sites, displaying fake login forms, tracking mouse movements, and so on. Hence, in future work, the actions after clicking on the URLs are examined by analyzing the features related to the URLs' actions. Then, the system will be trained based on the written script of the webpage and user behavior to develop an effective Phishing website detection system with advanced techniques in the future.

Endnotes

1. https://archive.ics.uci.edu/dataset/967/phiusiil+phishing+url+dataset
2. https://research.aalto.fi/en/datasets/phishstorm-phishing-legitimate-url-dataset
3. https://www.kaggle.com/datasets/shashwatwork/web-page-phishing-detection-dataset
4. https://www.unb.ca/cic/datasets/url-2016.html

Data availability statement

The original contributions presented in the study are included in the article/Supplementary material, further inquiries can be directed to the corresponding authors.

Author contributions

MM: Conceptualization, Data curation, Formal analysis, Methodology, Project administration, Software, Validation, Writing – original draft. GN: Supervision, Writing – review & editing.

Funding

The author(s) declare that no financial support was received for the research and/or publication of this article.

Acknowledgments

We thank the referees for their useful suggestions.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Generative AI statement

The author(s) declare that no Gen AI was used in the creation of this manuscript.

Any alternative text (alt text) provided alongside figures in this article has been generated by Frontiers with the support of artificial intelligence and reasonable efforts have been made to ensure accuracy, including review by the authors wherever possible. If you identify any issues, please contact us.

Publisher's note

Supplementary material

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fcomp.2025.1582206/full#supplementary-material

References

Aamir, M., and Zaidi, S. M. A. (2019). DDoS attack detection with feature engineering and machine learning: the framework and performance evaluation. *Int. J. Inf. Secur.* 18, 761–785. doi: 10.1007/s10207-019-00434-1

Abedin, N. F., Bawm, R., Sarwar, T., Saifuddin, M., Rahman, M. A., Hossain, S., et al. (2020). "Phishing attack detection using machine learning classification techniques," in *3rd International Conference on Intelligent Sustainable Systems (ICISS)* (Thoothukudi: IEEE), 1125–1130. doi: 10.1109/ICISS49785.2020.9315895

Abuzuraiq, A., Alkasassbeh, M., and Almseidin, M. (2020). "Intelligent methods for accurately detecting phishing websites," in *11th International Conference on Information and Communication Systems, (ICICS)* (Irbid: IEEE), 85–90. doi: 10.1109/ICICS49469.2020.239509

Alam, M. N., Sarma, D., Lima, F. F., Saha, I., Ulfath, R. E., Hossain, S., et al. (2020). "Phishing attacks detection using machine learning approach," in *Proceedings of the 3rd International Conference on Smart Systems and Inventive Technology (ICSSIT)* (Tirunelveli: IEEE), 1173–1179. doi: 10.1109/ICSSIT48917.2020.9214225

Ali, W., and Malebary, S. (2020). Particle swarm optimization-based feature weighting for improving intelligent phishing website detection. *IEEE Access* 8, 116766–116780. doi: 10.1109/ACCESS.2020.3003569

Alkawaz, M. H., Steven, S. J., and Hajamydeen, A. I. (2020). "Detecting phishing website using machine learning," in *Proceedings of the 16th IEEE International Colloquium on Signal Processing and Its Applications (CSPA)* (Langkawi: IEEE), 111–114. doi: 10.1109/CSPA48992.2020.9068728

Alqahtani, H., Alotaibi, S. S., Alrayes, F. S., Al-Turaiki, I., Alissa, K. A., Aziz, A. S., et al. (2022). Evolutionary algorithm with deep auto encoder network based website phishing detection and classification. *Appl. Sci.* 12, 1–16. doi: 10.3390/app12157441

Alsariera, Y. A., Elijah, A. V., and Balogun, A. O. (2020). Phishing website detection: forest by penalizing attributes algorithm and its enhanced variations. *Arab. J. Sci. Eng.* 45, 10459–10470. doi: 10.1007/s13369-020-04802-1

Aslam, S., Aslam, H., Manzoor, A., Chen, H., and Rasool, A. (2024). AntiPhishStack: LSTM-based stacked generalization model for optimized phishing URL detection. *Symmetry* 16, 1–25. doi: 10.3390/sym16020248

Balogun, A. O., Mojeed, H. A., Adewole, K. S., Akintola, A. G., Salihu, S. A., Bajeh, A., et al. (2021). Optimized decision forest for website phishing detection. *Data Sci. Intell. Syst. Proc. 5th Comput. Methods Syst. Softw.* 2, 568–582. doi: 10.1007/978-3-030-90321-3_47

Basit, A., Zafar, M., Javed, A. R., and Jalil, Z. (2020). "A novel ensemble machine learning method to detect phishing attack," in *Proceedings of the 23rd International MultiTopic Conference (INMIC)* (Bahawalpur: IEEE), 1–5. doi: 10.1109/INMIC50486.2020.9318210

Benavides-Astudillo, E., Fuertes, W., Sanchez-Gordon, S., Nunez-Agurto, D., and Rodriguez-Galan, G. (2023). A phishing-attack-detection model using natural language processing and deep learning. *Appl. Sci.* 13, 1–23. doi: 10.3390/app13095275

Castano, F., Fernandez, E. F., Alaiz-Rodriguez, R., and Alegre, E. (2023). PhiKitA: phishing kit attacks dataset for phishing websites identification. *IEEE Access* 11, 40779–40789. doi: 10.1109/ACCESS.2023.3268027

Colhak, F., Ecevit, M. I., Ucar, B. E., Creutzburg, R., and Dag, H. (2024). Phishing website detection through multi-model analysis of HTML content. *ArXiv* 1–15. Available online at: http://arxiv.org/abs/2401.04820 (Accessed September 10, 2025).

Do, N. Q., Selamat, A., Krejcar, O., Herrera-Viedma, E., and Fujita, H. (2022). Deep learning for phishing detection: taxonomy, current challenges and future directions. *IEEE Access* 10, 36429–36463. doi: 10.1109/ACCESS.2022.3151903

Feng, F., Zhou, Q., Shen, Z., Yang, X., Han, L., Wang, J., et al. (2018). The application of a novel neural network in the detection of phishing websites. *J. Ambient Intell. Humaniz. Comput.* 15, 1865–1879. doi: 10.1007/s12652-018-0786-3

Gupta, B. B., Yadav, K., Razzak, I., Psannis, K., Castiglione, A., Chang, X., et al. (2021). A novel approach for phishing URLs detection using lexical based machine learning in a real-time environment. *Comput. Commun.* 175, 47–57. doi: 10.1016/j.comcom.2021.04.023

Karim, A., Shahroz, M., Mustofa, K., Belhaouari, S. B., and Joga, S. R. K. (2023). Phishing detection system through hybrid machine learning based on URL. *IEEE Access* 11, 36805–36822. doi: 10.1109/ACCESS.2023.3252366

Korkmaz, M., Sahingoz, O. K., and Diri, B. (2020). "Detection of Phishing websites by using machine learning-based URL analysis," in *11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)* (Kharagpur: IEEE), 1–7. doi: 10.1109/ICCCNT49239.2020.9225561

Kumar, J., Santhanavijayan, A., Janet, B., Rajendran, B., and Bindhumadhava, B. S. (2020). "Phishing website classification and detection using machine learning," in *International Conference on Computer Communication and Informatics (ICCCI)* (Coimbatore: IEEE), 1–6. doi: 10.1109/ICCCI48352.2020.9104161

Kumar, P. P., Jaya, T., and Rajendran, V. (2023). SI-BBA – a novel phishing website detection based on swarm intelligence with deep learning. *Mater. Today Proc.* 80, 3129–3139. doi: 10.1016/j.matpr.2021.07.178

Lakshmanarao, A., Rao, P. S. P., and Krishna, M. B. (2021). "Phishing website detection using novel machine learning fusion approach," in *Proceedings of the International Conference on Artificial Intelligence and Smart Systems (ICAIS)* (Coimbatore: IEEE), 1164–1169. doi: 10.1109/ICAIS50930.2021.9395810

Lakshmi, L., Reddy, M. P., Santhaiah, C., and Reddy, U. J. (2021). Smart phishing detection in web pages using supervised deep learning classification and optimization technique ADAM. *Wireless Personal Commun.* 118, 3549–3564. doi: 10.1007/s11277-021-08196-7

Liu, D. J., Geng, G. G., Jin, X. B., and Wang, W. (2021). An efficient multistage phishing website detection model based on the CASE feature framework: aiming at the real web environment. *Comput. Secur.* 110:102421. doi: 10.1016/j.cose.2021.102421

Liu, D. J., Geng, G. G., and Zhang, X. C. (2022). Multi-scale semantic deep fusion models for phishing website detection. *Expert Syst. Appl.* 209, 1–13. doi: 10.1016/j.eswa.2022.118305

Mughaid, A., Al Zubi, S., Hnaif, A., Taamneh, S., Alnajjar, A., and Elsoud, E. A. (2022). An intelligent cyber security phishing detection system using deep learning techniques. *Cluster Comput.* 25, 3819–3828. doi: 10.1007/s10586-022-03604-4

Nahapetyan, A., Prasad, S., Childs, K., Oest, A., Ladwig, Y., Kapravelos, A., et al. (2024). "On sms phishing tactics and infrastructure," in *Proceeding of Symposium on Security and Privacy (SP)* (San Francisco, CA: IEEE), 1–16. doi: 10.1109/SP54263.2024.00169

Ozcan, A., Catal, C., Donmez, E., and Senturk, B. (2023). A hybrid DNN–LSTM model for detecting phishing URLs. *Neural Comput. Appl.* 35, 4957–4973. doi: 10.1007/s00521-021-06401-z

Purwanto, R. W., Pal, A., Blair, A., and Jha, S. (2022). PhishSim: aiding phishing website detection with a feature-free tool. *IEEE Trans. Inf. Forensics Secur.* 17, 1497–1512. doi: 10.1109/TIFS.2022.3164212

Rao, R. S., Pais, A. R., and Anand, P. (2021). A heuristic technique to detect phishing websites using TWSVM classifier. *Neural Comput. Appl.* 33, 5733–5752. doi: 10.1007/s00521-020-05354-z

Rashid, J., Mahmood, T., Nisar, M. W., and Nazir, T. (2020). "Phishing detection using machine learning technique," in *Proceedings of the 1st International Conference of Smart Systems and Emerging Technologies (SMARTTECH)* (Riyadh: IEEE), 43–46. doi: 10.1109/SMART-TECH49988.2020.00026

Sabahno, M., and Safara, F. (2021). ISHO: improved spotted hyena optimization algorithm for phishing website detection. *Multimed. Tools Appl.* 81, 34677–34696. doi: 10.1007/s11042-021-10678-6

Safi, A., and Singh, S. (2023). A systematic literature review on phishing website detection techniques. *J. King Saud Univ. Comput. Inf. Sci.* 35, 590–611. doi: 10.1016/j.jksuci.2023.01.004

Sanchez-Paniagua, M., Fidalgo, E., Alegre, E., and Alaiz-Rodriguez, R. (2022). Phishing websites detection using a novel multipurpose dataset and web technologies features. *Expert Syst. Appl.* 207, 1–16. doi: 10.1016/j.eswa.2022.118010

Somesha, M., Pais, A. R., Rao, R. S., and Rathour, V. S. (2020). Efficient deep learning techniques for the detection of phishing websites. *Sadhana* 45, 1–18. doi: 10.1007/s12046-020-01392-4

Tang, L., and Mahmoud, Q. H. (2022). A deep learning-based framework for phishing website detection. *IEEE Access* 10, 1509–1521. doi: 10.1109/ACCESS.2021.3137636

Wei, W., Ke, Q., Nowak, J., Korytkowski, M., Scherer, R., Wozniak, M., et al. (2020). Accurate and fast URL phishing detector: a convolutional neural network approach. *Comput. Netw.* 178, 1–9. doi: 10.1016/j.comnet.2020.107275

Xiao, X., Xiao, W., Zhang, D., Zhang, B., Hu, G., Li, Q., et al. (2021). Phishing websites detection via CNN and multi-head self-attention on imbalanced datasets. *Comput. Secur.* 108, 1–14. doi: 10.1016/j.cose.2021.102372

Zhu, E., Ju, Y., Chen, Z., Liu, F., and Fang, X. (2020). DTOF-ANN: an artificial neural network phishing detection model based on decision tree and optimal features. *Appl. Soft Comput. J.* 95, 1–14. doi: 10.1016/j.asoc.2020.106505