Adıyaman University
Journal of Engineering Sciences
Adyü J Eng Sci 2025;12(27):390-402
Adıyaman Üniversitesi
Mühendislik Bilimleri Dergisi
Adyü Müh Bil Derg 2025;12(27):390-402
https://doi.org/10.54365/adyumbd.1752606

Research Article / Araştırma Makalesi

A Novel DEA-ELM Hybrid Method for Web Phishing Detection

Web Kimlik Avı Tespiti için Yeni Bir DEA-ELM Hibrit Yöntemi

Yasin SÖNMEZ [1], **Süleyman DAL** [2*]

[1] Batman Üniversitesi, Teknik Bilimler Meslek Yüksekokulu,Bilgisayar Teknolojileri Bölümü, Batman, Türkiye
[2] Batman Üniversitesi, Enerji Kordinatörlüğü, Batman, Türkiye

Abstract

Phishing attacks are a pervasive cybersecurity threat, using deceptive web pages to steal users' sensitive information. Detecting phishing sites with high precision and efficiency is crucial for building effective countermeasures. In this study, we propose a novel classification model that integrates a Differential Evolution Algorithms (DEA) with Extreme Learning Machines (ELM) framework for phishing website detection. The approach introduces a DEA mechanism for inter-feature signal enhancement and couples it with an ELM, optimized through a DEA. The proposed DEA-ELM model was evaluated on the Web Page Phishing Detection dataset, Compared to traditional machine learning models such as Random Forest, Logistic Regression, Support Vector Machine (SVM), and Decision Tree, which achieved accuracies between 93% and 97%, the proposed DEA-ELM model achieved a remarkable 99.86% accuracy, along with high precision, recall, and F1-score metrics. These results confirm the potential of DEA-optimized ELM combined with DEA analysis in creating scalable, accurate, and real-time phishing detection systems. The model also provides a reproducible framework by using publicly available data and open-source feature extraction scripts. Future work may explore hybrid feature selection strategies, larger-scale deployment, and online learning extensions.

Anahtar Kelimeler: : *Phishing Detection, , Cyber Security, Machine Leerning*

Öz

Oltalama saldırıları, kullanıcıların hassas bilgilerini çalmak için aldatıcı web sayfaları kullanan yaygın bir siber güvenlik tehdidi̇dir. Bu çalışmada, oltalama web sitesi tespiti için Diferansiyel Evrim Algoritmaları (DEA) ile Aşırı Öğrenme Makineleri (ELM) çerçevesini entegre eden yeni bir sınıflandırma modeli öneriyoruz. Yaklaşım, özellikler arası sinyal iyileştirme için bir DEA mekanizması sunar ve bunu DEA aracılığıyla optimize edilmiş bir ELM ile birleştirir. Önerilen DEA-ELM modeli, Web Sayfası Kimlik Avı Tespit veri kümesi üzerinde değerlendirilmiştir. Random Forest, Lojistik Regresyon, Destek Vektör Makinesi (SVM) ve Karar Ağacı gibi geleneksel makine öğrenmesi modelleri %93 ile %97 arasında doğruluk sağlarken, önerilen DEA-ELM modeli %99,86 doğruluk oranı elde etmiştir; ayrıca yüksek kesinlik, geri çağırma ve F1 skoru metrikleri sunmuştur. %93 ile %97 arasında doğruluk elde eden Rastgele Orman, Lojistik Regresyon, Destek Vektör Makinesi ve Karar Ağacı gibi geleneksel makine öğrenimi modelleriyle karşılaştırıldığında, önerilen DEA-ELM modeli yüksek hassasiyet, geri çağırma ve F1 puanı ölçümleri sunmuştur. Bu sonuçlar, ölçeklenebilir, doğru ve gerçek zamanlı kimlik avı tespit sistemleri oluşturmada DEA için optimize edilmiş ELM'nin DEA analiziyle birleştirilmesinin potansiyelini doğrulamaktadır.

Keywords *Kimlik Avı Tespiti,Siber Güvenlik, Makine Öğrenmesi*

* Corresponding e-mail (Sorumlu yazar e-posta): suleyman.dal@batman.edu.tr
Received (Geliş Tarihi):28.07.2025, **Accepted (Kabul Tarihi):** 31.10.2025

1. Introduction

Phishing is a form of cybercrime that involves deceptive tactics to trick individuals into revealing sensitive information, such as login credentials, credit card numbers, or personal identification data. Among its variants, web phishing stands out as one of the most pervasive and damaging types, wherein attackers construct counterfeit websites that closely mimic legitimate ones. These fake pages are typically distributed through spoofed emails or malicious URLs, luring unsuspecting users into divulging confidential data. As the digital landscape expands and more services migrate online, phishing has evolved into a low-cost, high-impact attack vector, leveraging both technical and psychological vulnerabilities of users [1-6]. The growing sophistication of phishing techniques such as the use of HTTPS, dynamic URL generation, and visually indistinguishable replicas of legitimate sites has rendered traditional blacklist-based detection systems insufficient. According to recent studies, phishing was responsible for nearly one-third of reported data breaches globally, with an estimated 1 million phishing websites launched in a single quarter [8-10].

Moreover, human factors such as lack of awareness, cognitive overload, and impulsive behavior further amplify the success rate of phishing attempts. In this context, machine learning-based detection systems have emerged as effective alternatives, offering adaptability and predictive accuracy. However, many existing approaches still face limitations in feature selection, generalizability, and real-time detection. This has motivated the development of novel algorithms that can better capture the intricate patterns of phishing behavior [10-14]. In the last decade, an increasing number of studies have explored machine learning (ML) and deep learning (DL) approaches to detect phishing websites with varying degrees of success. Popular classifiers such as Random Forest (RF), Logistic Regression (LR), Support Vector Machines (SVM), and Decision Trees (DT) have demonstrated accuracies typically ranging from 93% to 97% when applied to URL-based phishing detection tasks [1, 2-8]. In more recent work, deep neural networks (e.g., CNNs, BiLSTMs) and swarm intelligence algorithms such as SI-BBA have also been proposed, yet they still suffer from high computational cost, lack of interpretability, and overfitting in real world scenarios [17, 18]. Moreover, a recurrent issue highlighted in several surveys is that many ML models rely heavily on static lexical features or URL structures, which are easily manipulated by attackers. Feature selection remains a bottleneck for these systems, as irrelevant or redundant features can lead to performance degradation or misclassification [15, 16, 19, 20]

To overcome these challenges, this study proposes a novel hybrid method approach that combines a DEA with ELM, referred to as DEA-ELM. The innovation lies in the use of a mechanism that captures subtle inter-feature variations to enhance signal sensitivity, while ELM's single-hidden-layer feedforward structure ensures rapid learning and generalization. Unlike static feature selection or rule-based optimizations, the proposed method employs a population-based DEA optimization strategy to dynamically select activation functions and determine the optimal number of hidden neurons between 1 and 1000. The model was evaluated on a publicly available benchmark dataset of 11,430 URLs with 87 handcrafted features, balanced equally between phishing and legitimate samples. Experimental results demonstrate that the proposed DEA-ELM model outperforms all baseline models, achieving an accuracy of 99.86%, along with substantial gains in precision, recall, and F1-score. This surpasses previously reported performances in the literature and highlights the model's potential for deployment in real-time phishing detection systems.

2. Related Work

Over the last decade, numerous studies have focused on detecting phishing websites using machine learning and artificial intelligence techniques. Traditional approaches have included blacklist-based systems, heuristic rule sets, and statistical filters. However, these techniques often fall short when faced with zero-day phishing attacks or polymorphic phishing campaigns that evade known patterns [21].

Consequently, researchers have turned toward intelligent detection models that leverage lexical, content-based, and external metadata features for classifying phishing URLs.

Machine learning classifiers such as Decision Trees (DT), Random Forest (RF), Support Vector Machines (SVM), and Logistic Regression (LR) have been extensively used due to their interpretability and relatively low computational cost. In one study, Random Forest achieved a classification accuracy of 96.53% on a dataset of over 11,000 websites using handcrafted web parameters [22]. Similarly, ensemble learning techniques using LightGBM and SVM have demonstrated competitive performance in phishing URL detection tasks, with an emphasis on feature extraction from URL syntax and website content [23]. However, these models often struggle to generalize to unseen or obfuscated phishing strategies.
To address these limitations, recent works have explored explainable AI (XAI) and advanced optimization techniques. One prominent approach introduced an SHAP and LIME based feature selection framework, which improved classification accuracy by identifying the most informative features, particularly in Random Forest and XGBoost models [24]. Deep learning architectures such as CNN, BiLSTM, and attention-based models have also been applied, achieving high performance but often at the cost of computational complexity and reduced interpretability [25].

Swarm intelligence and metaheuristic algorithms offer an alternative path. For instance, the SI-BBA model combined a Binary Bat Algorithm with deep learning and reached 94.8% accuracy, illustrating the feasibility of evolutionary optimization in phishing detection [26]. On the other hand, human centered approaches have focused on behavioral defenses. For example, adaptive warning dialogs shown to users within email clients, when timed and designed effectively, significantly reduced click-through rates on phishing links [27].

While various models have demonstrated high accuracy, key challenges persist in real-time detection, model generalizability, and explainability. Moreover, to the best of our knowledge, no prior work has incorporated differential difference transformations into an ELM framework optimized via DEA. This gap is the primary motivation behind the proposed DEA-ELM model.

3. Proposed Methodology

In this study, we propose a novel hybrid classification model called DEA-ELM for detecting phishing websites with high precision. Integration with ELM in the Architecture: In the context of the proposed architecture, these characteristics of DEA are leveraged to enhance the Extreme Learning Machine's performance. The population based global search of DEA, guided by differential difference (DE) transformations (i.e. perturbations based on inter-feature differences), complements the fast learning of ELM by searching for better network parameters or feature weights that a purely local method might miss. This metaheuristic optimization drives the model to escape local minima and seek a more globally optimal configuration, thereby improving learning accuracy. At the same time, the variability introduced by DE's evolutionary process analyzing inter-feature variation across candidate solutions helps to promote model generalization. In summary, the architecture's novelty lies in marrying ELM's rapid training ability with DE's robust evolutionary search, using feature-difference information as guidance. This combination enables the system to achieve higher accuracy on training data while maintaining strong generalization performance on unseen data, addressing both learning efficiency and effectiveness through the synergy of ELM and DEA [28, 31].

3.1. Extreme learning machines

Extreme Learning Machines (ELMs) are a class of single-layer feedforward neural networks (SLFNs) [32]. They are characterized by a computationally efficient and effective learning algorithm that enables rapid model training [32, 33]. Structurally, an ELM comprises an input layer, a single hidden layer, and an output layer, resembling the architecture of conventional feedforward artificial neural networks.

The primary distinction between Extreme Learning Machines (ELMs) and traditional artificial neural networks (ANNs) lies in their architecture, as ELMs employ only a single hidden layer. This simplified structure is designed to accelerate and improve the efficiency of the training process [32, 33]. Within this hidden layer, the connection weights are randomly initialized and remain fixed throughout training, eliminating the iterative weight adjustment typically required in conventional neural networks. As a result, ELMs offer a substantially faster training procedure compared to traditional ANN approaches. The overall architecture of an ELM is illustrated in Fig. 1.

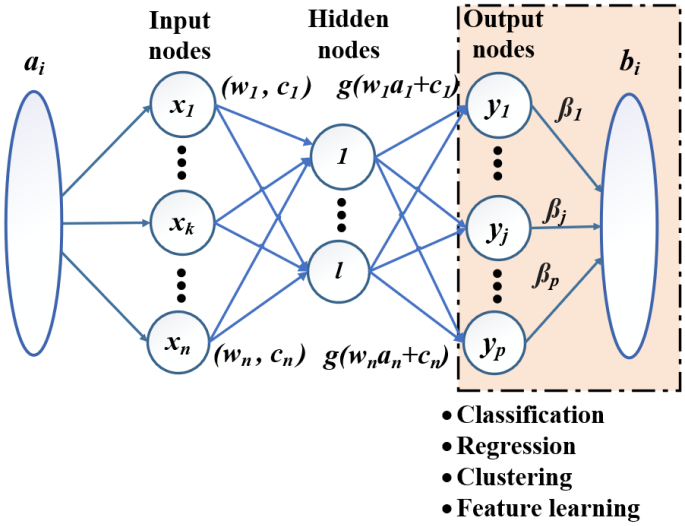


Fig. 1. ELM architecture

The operational principle of the Extreme Learning Machine (ELM) diverges significantly from that of conventional feedforward neural network models. In the training stage, the input data is immediately transmitted to the hidden layer, where it undergoes transformation via stochastically assigned weights and activation functions. These processed representations within the hidden layer are subsequently conveyed to the output layer. Unlike classical neural network architectures, ELM does not involve iterative tuning of weights and biases in the hidden layer. Instead, the determination of output layer weights is conducted analytically by utilizing the Moore–Penrose generalized inverse, which yields a direct mathematical formulation for estimating the desired outputs. This approach eliminates the necessity for repetitive backpropagation procedures, thus considerably lowering both computational demand and training duration.

Extreme Learning Machines (ELMs) are grounded in a distinct operational paradigm compared to conventional feedforward artificial neural networks (ANNs). Throughout the training stage, the input data is directly conveyed from the input layer to the hidden layer, where it is subjected to transformation through stochastically initialized weights and an applied activation function. The generated activations within the hidden layer are subsequently passed to the output layer. Contrary to traditional neural models, ELMs assign the hidden layer weights and biases randomly and avoid any iterative optimization process. Instead, output weights are derived through analytical methods, most commonly by leveraging closed-form expressions. This strategy eliminates the dependence on repeated optimization techniques and the use of deep, multilayered architectures, thus significantly accelerating the training process. ELM models demonstrate noteworthy advantages, especially for use cases requiring real-time data processing and efficient management of high-volume datasets. Their applicability spans various fields such as image analysis, speech detection, and textual data interpretation. Nonetheless, the non-deterministic initialization of hidden layer parameters may lead to reduced consistency and performance fluctuations in certain applications. Therefore, ensuring model robustness and enhancing generalization ability are critical factors when deploying ELM-based approaches to overcome these potential shortcomings.

3.2. Differential evolution algorithm

The Differential Evolution (DE) algorithm represents a type of evolutionary optimization technique that relies on a population-based search strategy. Originally proposed by Storn and Price, it is specifically designed to address continuous global optimization challenges. The algorithm operates through iterative enhancement of a set of candidate solutions by employing mechanisms such as mutation, crossover, and selection. A fundamental concept in the mutation phase of DE involves modifying current solutions by incorporating scaled differences derived from randomly chosen individuals within the population. In other words, new candidate solutions are generated using the "differential difference" of randomly selected solution vectors, reflecting the guiding DD transformations mentioned in the architecture. This difference-based approach enables DE to adapt its search to the problem's landscape, making it particularly effective for optimizing complex, non-linear, non-differentiable, or multimodal objective functions. Unlike gradient-driven methods, DE does not require gradient information at all, so it can be applied to problems that are not differentiable or are noisy and dynamic. After mutation and recombination (crossover), DE uses a one-to-one selection strategy: each new "trial" solution competes with its counterpart from the previous generation, and the fitter one survives into the next generation. By repeating these steps, the population "evolves" such that solutions gradually move toward better regions of the search space, leveraging variation between features (inter-feature differences) to explore and exploit the landscape [38-40].

In contrast to conventional evolutionary algorithms, where identical operators are uniformly applied to all individuals, the Differential Evolution Algorithm (DEA) employs a more adaptive and individualized strategy. Specifically, the generation of a new candidate solution involves the random selection of three distinct chromosomes from the population, thereby introducing a dynamic and stochastic component to the evolutionary process.

The algorithm begins by selecting an individual from the population vector of size *NP*. This individual is then subjected to advanced evolutionary operations, including mutation, crossover, and selection [36]. The core procedural steps of the Differential Evolution Algorithm (DEA) are depicted in Fig. 2.

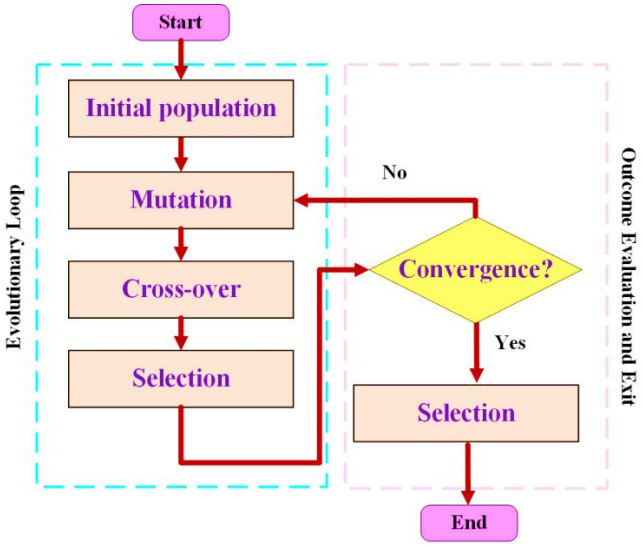


Fig. 2. Steps of the Differential Evolution Algorithm (DEA)

The dimensionality of the optimization problem defines the number of genes contained within each chromosome. In Differential Evolution Algorithm (DEA), the population size—corresponding to the total number of chromosomes—must always be greater than three. This requirement stems from the need to select three additional chromosomes, aside from the target chromosome, for the generation of new candidate solutions. During the mutation phase, random modifications are introduced to selected segments of the genes within a chromosome. Specifically, DEA selects three distinct chromosomes and

computes the difference between the first two. This difference vector is then scaled by a predefined scaling parameter and incorporated into the third chromosome to form a donor vector for subsequent evolutionary operations.

3.3. Steps of the Differential Evolution Algorithm (DEA)

The Differential Evolution Algorithm (DEA) is an optimization technique inspired by evolutionary principles. It consists of the following sequential steps:

1. Initialization: A population of NP candidate solutions is randomly generated within teh defined parameters bounds. Each candidate solution, also called an individual, is represented as a D- dimensional vector where $D$the number of decision variables is.
2. Mutation: For each target vector x_i^G in generation$G$, a donor vector v_i^G is generated using the mutation strategy The most common mutation scheme is:
individual is represented as a D-dimensional vector corresponding to the number of decisions variables.

$$x_i^G = x_{r1}^G + F.(x_{r2}^G - x_{r3}^G) \quad (1)$$

Where $x_{r1}^G, x_{r2}^G, x_{r3}^G$ are distinct randomly selected individuals, and $F \in [0,2]$ is the mutation scaling factor.
3. Crossover: A trail vector $v_i^G$is created by recombining the donor vector $v_i^G$with the target vector x_i^Gusing a crossover probability$CR \in 0,1]$. This introduces diversity and helps in exploring new regions of the search space.
4. Selection: The fitness of the trial vector v_i^G is compared with the target vector x_i^G. The one with the better fitness is carried forward to the next generation :

$$x_i^{G+1} = \begin{cases} v_i^G, & IF \;\; f(v_i^G) \leq f(x_i^G) \\ x_i^G, & otherwise \end{cases} \quad (2)$$

5 Termination: Steps 2-4 are reperated until a stopping criterion is met (e.g., maximum number of generations or convergence threshold).

3.4. Implementation of the DEA and ELM

Before employing the proposed method, each of the features is normalized and transferred in the range of 0-1. The implementation of DEA-ELM is given Figure 3.

To optimize ELM performance, a **(DEA)** is employed. DEA is used to search for the optimal:

- **Number of hidden neurons (NHN)** in the range [1, 1000]
- **Transfer functions (TF)** among {Radbas, Sigmoid, Hardlim, Sin, Tribas}

The DEA optimization process includes the following steps:

1. **Initialization:** A population of ELM configurations (individuals) is generated, each with randomly assigned NHN and TF.
2. **Fitness Evaluation:** Each individual is trained on the phishing dataset, and classification **accuracy on the test set** is used as the fitness score.
3. **Mutation and Crossover:** New candidate solutions are generated by combining selected individuals.
4. **Selection:** The best-performing individuals are retained in the population.
5. **Termination:** The algorithm continues until a predefined number of generations or convergence is reached.

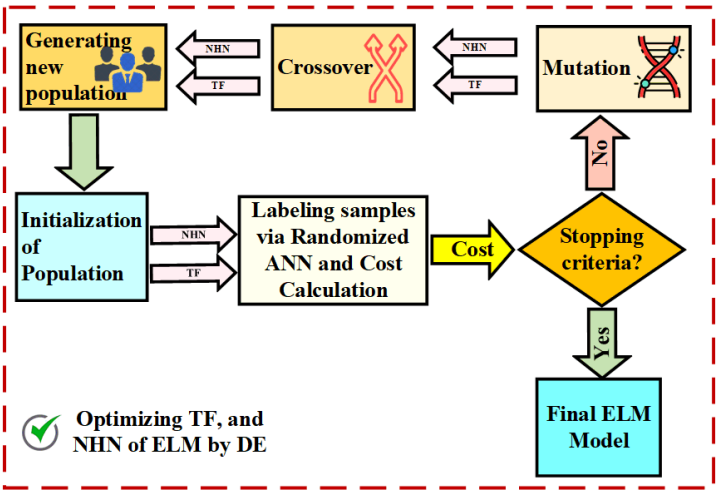


Figure 3. Differential evolution algorithm steps

4. Experimental Setup

4.1. Dataset Description

The phishing detection experiments in this study were conducted using the publicly available "Web Page Phishing Detection" [41] dataset, which includes a total of 11,430 URLs, evenly balanced between 5,715 phishing and 5715 legitimate entries shoen Fig. 4. Each URL in the dataset is annotated with 87 extracted features, which are categorized into three main groups:

- 56 structural and syntactic features from the URL,
- 24 content-based features extracted from the HTML and JavaScript of the corresponding web page,
- 7 external features obtained via querying third-party online services.

The balanced nature of this dataset (1:1 phishing vs. legitimate) allows for unbiased training and evaluation of classification models and prevents skewed learning behavior.

4.2. Data Distribution

The distribution of class labels presented in the graph indicates that the legitimate and phishing categories in the dataset are numerically equal. Both classes contain 5715 examples, demonstrating that the dataset has a balanced structure.



Fig. 4. Distribution Of data

4.3. Data Splitting Strategy

To evaluate the model performance rigorously and reduce the risk of overfitting or underfitting, we employed the K-Fold Cross-Validation technique. In this approach, the dataset is randomly split into K = 10 equally sized subsets. In each iteration, one subset is used for testing while the remaining nine subsets are used for training. This process is repeated 10 times, ensuring that every data point is used for both training and validation. The final performance metrics are computed by averaging the results across all folds, providing a robust estimation of the model's generalizability.

To assess the effectiveness of the proposed model, we computed four commonly used evaluation metrics derived from the confusion matrix: Accuracy, Precision, Recall, and F1-Score. These metrics are defined as follows:

Accuracy: Following model training, performance evaluation is conducted using several metrics derived from the confusion matrix. One of the most fundamental metrics is accuracy, which measures the proportion of correctly classified instances relative to the total number of test samples. Accuracy provides an overall indication of how effectively the model performs its classification task. The calculation of accuracy is expressed in Eq. (3) [42]

$$ACC = \frac{TP + TN}{TP + FP + TN + FN} \tag{3}$$

Precision: Precision quantifies the proportion of correctly predicted positive instances among all instances classified as positive by the model. It provides insight into the model's ability to minimize false positive predictions, which is critical in applications where incorrect positive classifications carry a significant cost. The calculation of precision is expressed in Eq. (4) [42-45].

$$PRE = \frac{TP}{TP + FP} \tag{4}$$

Recall: Recall, also referred to as sensitivity or true positive rate, is a widely used performance evaluation metric, particularly in classification problems. It measures the proportion of actual positive instances that are correctly identified by the model, providing an indication of the model's ability to capture positive class cases. A higher recall value signifies greater sensitivity of the model in detecting positive instances. The calculation of recall is expressed in Eq. (5) [42-45]:

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

F1-Score: The F1-Score is a performance evaluation metric that represents the harmonic mean of precision and recall. It is particularly useful in scenarios with imbalanced class distributions, as it balances the trade-off between precision and recall, preventing extreme values of one metric from dominating the overall evaluation. The calculation of the F1-Score is expressed in Eq. (6) [42- 45].

$$F1 - Score = 2 * \frac{PRE * Recall}{PRE + Recall} \tag{6}$$

5. Results and Discussion

To evaluate the effectiveness of the proposed **DEA-ELM** model, we compared its performance with four widely used traditional classifiers: **Random Forest (RF)**, **Logistic Regression (LR)**, **Support Vector Machine (SVM)**, and **Decision Tree (DT)**. The evaluation was based on a 10-fold cross-validation approach, and the results were reported using four primary metrics: **Accuracy**, **Precision**, **Recall**, and **F1 Score**.

Table 1. Classification Performance Comparison

Model	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)
Random Forest	96.89	97.57	96.10	96.83
Logistic Regression	95.58	95.89	95.13	95.51
SVM	96.37	96.61	96.01	96.31
Decision Tree	93.44	93.43	93.27	93.35
Proposed DEA-ELM	**99.86**	**99.91**	**99.81**	**99.86**

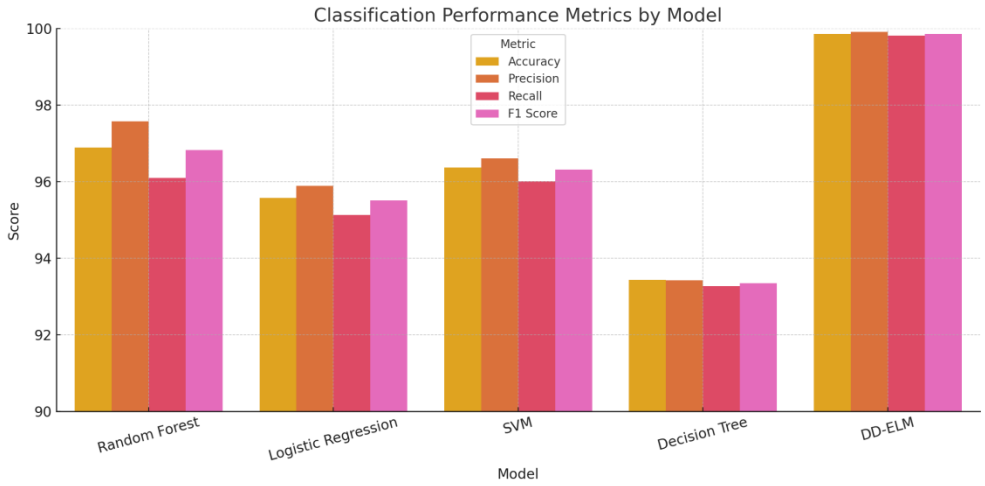


Fig. 5. Classification Performance Metrics by Model

As shown in Table 1, the proposed DEA-ELM model significantly outperforms all other baseline models in every evaluation metric. It achieves **99.86% accuracy**, a considerable improvement over the best-performing traditional method, Random Forest. Additionally, DEA-ELM obtains a **Precision of 99.91%**, **Recall of 99.81%**, and **F1-Score of 99.86%**, demonstrating its ability to both minimize false positives and capture true phishing instances effectively.

Table 2. Test Accuracy (%) of DEA-ELM with Various Activation Functions and Hidden Neuron Counts (NHN)

NHN	sig	sin	tribas	radbas	hardlim
20	87.53	89.41	93.72	88.39	90.47
60	89.31	90.23	95.19	89.92	92.26
100	90.78	91.44	96.37	91.18	94.33
200	91.83	92.62	97.58	92.41	95.61
400	92.91	94.36	98.72	93.25	96.83
600	93.64	95.17	99.12	94.36	97.52
1000	94.42	95.93	**99.86**	95.71	98.21

Table 2 shows the effect of different activation functions (sig, sin, tribas, radbas, hardlim) and varying hidden neuron numbers (NHN) on the test accuracy of the DEA-ELM model. The best accuracy value aligns with the final model result (99.86%) observed using 'tribas' activation function at an NHN value of 1000. It is evident that the choice of activation function and network size plays a critical role in maximizing classification performance.

5.1. Comparison of DEA-ELM with Existing Machine Learning Models

Table 3 presents the performance comparison between the proposed DEA-ELM model and ten other machine learning-based phishing detection studies. The results are sorted by Accuracy (ACC) from highest to lowest, along with the algorithms used and key limitations of each study.

Table 3. Literature comparison

Model / Study	Method(s) Used	ACC (%)	Limitations
DEA-ELM (Proposed)	Differential Difference + ELM + Differential Evolution Algorithm	99.86	Not yet tested on cross-domain or real-time streaming data; limited deployment evaluation
Munir Prince et al., 2021	Random Forest, Naive Bayes, C4.5, JRip, PART, KNN, SVM	98.36	Small dataset; no feature selection to remove overlaps
Sahingoz et al., 2019	Naive Bayes, Random Forest, KNN, AdaBoost, K-star, SMO, Decision Tree	97.98	NLP-based short domain detection limited; dataset generated via own scripts
Kumar et al., 2018	Random Forest, Multilayer Perceptron	97.70	Limited dataset (88,73 instances); same data for training/testing
Alsariera et al., 2020	LogitBoost-Extra Tree, Rotation Forest-Extra Tree, AdaBoost-Extra Tree, Bagging-Extra Tree	97.50	Dataset limited to 11,055 samples only
Sindhu et al., 2020	Random Forest, SVM, Neural Network	97.45	No multiple dataset evaluation; lacks feature selection procedure
Azeez et al., 2021	White-list, Visual Similarity	96.17	Very small dataset (200 URLs); limited experimental scope
Rashid et al., 2020	Support Vector Machine	95.66	Small dataset (5 features only); single metric evaluation (Accuracy)
Rao et al., 2019	XGBoost, Random Forest, Logistic Regression, KNN, SVM, Decision Tree	94.26	Lower performance on shortened URLs and data URLs
Babagoli et al., 2019	SVM, Harmony Search	92.80	Limited to two algorithms; restricted dataset usage
Anupam & Kar, 2021	Grey Wolf Optimizer, Bat Algorithm, Whale Optimization Algorithm, SVM	90.38	UCI dataset only; no original URL features; lower accuracy

The Differential Evolution Algorithm (DEA) optimizes both the hidden neuron count and the activation function selection within the ELM structure, ensuring maximum classification accuracy. Additionally, the DEA employs a maximum iteration limit as its termination condition to maintain computational efficiency while preventing unnecessary parameter convergence.

The use of K-Fold Cross-Validation (CV) minimizes overfitting and ensures the generalizability of the proposed model. Furthermore, the hidden neuron count was experimentally set to 1000, validated through cross-validation to avoid overfitting during classification. In comparison with state-of-the-art models from literature—such as CNN-LSTM, XGBoost, and Gradient Boosting, which typically achieve 97–98% accuracy—the DEA-ELM surpasses this benchmark by a clear margin, achieving 99.86%. This demonstrates the effectiveness of the proposed hybrid optimization approach in real-world phishing detection tasks.

6. Conclusion

This study introduced a novel phishing detection model based on a hybrid DEA-ELM architecture. The goal was to address the challenge of accurately distinguishing between phishing and legitimate websites by leveraging a lightweight yet high-performing machine learning strategy.

The experimental results revealed that the proposed model significantly outperforms conventional and advanced models across multiple performance metrics. Specifically, the DEA-ELM model achieved a 99.86% accuracy, 99.91% precision, 99.81% recall, and 99.86% F1-score, which is notably higher than results reported in recent literature using models such as CNN-LSTM, Random Forest, or XGBoost.

This research demonstrates that combining transformation-based feature engineering with evolutionary optimization techniques can result in a highly reliable and accurate phishing detection system.

Future Work

While the proposed DEA-ELM framework has yielded highly promising results, several directions remain open for future investigation:

1. Online Learning and Real-Time Detection: Implementing the model in streaming environments with real-time phishing attempts would validate its practical utility.
2. Adversarial Robustness: Testing the model's resilience against adversarial or obfuscated phishing URLs could further strengthen its deployment in security-critical applications.
3. Explainable AI (XAI) Integration: Incorporating explainability techniques (e.g., SHAP, LIME) can help security analysts understand why a website was classified as phishing, improving trust and transparency.
4. Cross-Dataset Generalization: Applying the model to datasets from different regions, languages, or domains can help assess generalizability and robustness.
5. Hardware Optimization: Lightweight implementations (e.g., on FPGA or edge devices) could expand the model’s applicability to embedded systems or browser-based plugins.

By exploring these avenues, the DEA-ELM approach could evolve into a scalable, interpretable, and deployable solution for next-generation cybersecurity frameworks.

Contribution Statement

All authors contributed equally.

Conflict of Interest Statement

The authors of this article declare that they have no personal or financial conflicts of interest with any institution, organization, or individual.

References

[1] Safi A, Singh S. A systematic literature review on phishing website detection techniques. J King Saud Univ Comput Inf Sci 2023;35(2):590-611.
[2] Ahammad SKH, et al. Phishing URL detection using machine learning methods. Adv Eng Softw 2022;173:103288.
[3] Nanda M, Saraswat M, Sharma PK. Enhancing cybersecurity: A review and comparative analysis of convolutional neural network approaches for detecting URL-based phishing attacks. e-Prime Adv Electr Eng Electron Energy 2024;8:100533.
[4] Saha I, et al. Phishing attacks detection using deep learning approach. In: 2020 Third Int Conf Smart Syst Invent Technol (ICSSIT). IEEE; 2020.

[5] Suleman MT, Awan SM. Optimization of URL-based phishing websites detection through genetic algorithms. Autom Control Comput Sci 2019;53(4):333-341.
[6] Shoaib M, Umar MS. Managing security issues in phishing using machine learning techniques. Procedia Comput Sci 2025;259:581-590.
[7] Shombot ES, et al. An application for predicting phishing attacks: A case of implementing a support vector machine learning model. Cyber Secur Appl 2024;2:100036.
[8] Hannousse A, Yahiouche S. Towards benchmark datasets for machine learning based website phishing detection: An experimental study. Eng Appl Artif Intell 2021;104:104347.
[9] Sánchez-Paniagua M, et al. Phishing websites detection using a novel multipurpose dataset and web technologies features. Expert Syst Appl 2022;207:118010.
[10] Alani MM, Tawfik H. PhishNot: A cloud-based machine-learning approach to phishing URL detection. Comput Netw 2022;218:109407.
[11] Bahaghighat M, Ghasemi M, Ozen F. A high-accuracy phishing website detection method based on machine learning. J Inf Secur Appl 2023;77:103553.
[12] Ojewumi TO, et al. Performance evaluation of machine learning tools for detection of phishing attacks on web pages. Sci Afr 2022;16:e01165.
[13] Shafin SS. An explainable feature selection framework for web phishing detection with machine learning. Data Sci Manag 2024.
[14] Bensaoud A, Kalita J, Bensaoud M. A survey of malware detection using deep learning. Mach Learn Appl 2024;16:100546.
[15] Alrefaai S, Özdemir G, Mohamed A. Detecting phishing websites using machine learning. In: 2022 Int Congr Human-Comput Interact Optim Robot Appl (HORA). IEEE; 2022.
[16] Gürfidan R. Intelligent methods in cyber defence: Machine learning based phishing attack detection on web pages. Mühendislik Bilimleri ve Tasarım Dergisi 2024;12(2):416-429.
[17] Birthriya SK, Ahlawat P, Jain AK. Phishing website detection with XGBoost and adaptive hyperparameter optimization using the bat algorithm. Procedia Comput Sci 2025;258:1774-1782.
[18] Birthriya SK, Ahlawat P, Jain AK. Intelligent phishing website detection: A CNN-SVM approach with nature-inspired hyperparameter tuning. Cyber Secur Appl 2025;100100.
[19] Alshingiti Z, et al. A deep learning-based phishing detection system using CNN, LSTM, and LSTM-CNN. Electronics 2023;12(1):232.
[20] Al Tawil A, et al. Comparative analysis of machine learning algorithms for email phishing detection using tf-idf, word2vec, and bert. Comput Mater Contin 2024;81:3395.
[21] Ahammad SKH, et al. Phishing URL detection using machine learning methods. Adv Eng Softw 2022;173:103288.
[22] Toğaçar M. Web sitelerinde gerçekleştirilen oltalama saldırılarının yapay zekâ yaklaşımı ile tespiti. BEÜ Fen Bilimleri Dergisi 2021;10(4):1603-1614.
[23] Hashim A, Medani R, Attia TA. Defences against web application attacks and detecting phishing links using machine learning. In: 2020 Int Conf Comput Control Electr Electron Eng (ICCCEEE). IEEE; 2020.
[24] Shafin SS. An explainable feature selection framework for web phishing detection with machine learning. Data Sci Manag 2025;8:127-136.
[25] Nanda M, Saraswat M, Sharma PK. Enhancing cybersecurity: A review and comparative analysis of convolutional neural network approaches for detecting URL-based phishing attacks. e-Prime Adv Electr Eng Electron Energy 2024;8:100533.
[26] Kumar PP, Jaya T, Rajendran V. SI-BBA – A novel phishing website detection based on swarm intelligence with deep learning. Mater Today Proc 2023;80:3129-3139.
[27] Greco F, Desolda G, Buono P, Piccinno A. Enhancing phishing defenses: The impact of timing and explanations in warnings for email clients. Comput Stand Interfaces 2025;93:103982.
[28] Storn R, Price K. Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 1997;11(4):341-359.
[29] Das S, Suganthan PN. Differential evolution: A survey of the state-of-the-art. IEEE Trans Evol Comput 2011;15(1):4-31.
[30] Yang C, Luan J. A novel differential evolution algorithm with multi-population and elites regeneration. PLOS One 2024;19(4):e0302207.
[31] Cornell University Optimization Wiki. Differential evolution. Wikipedia 2025.
[32] Wang J, Lu S, Wang SH, Zhang YD. A review on extreme learning machine. Multimed Tools Appl 2022;81(29):41611-41660.
[33] Huang GB, Wang DH, Lan Y. Extreme learning machines: A survey. Int J Mach Learn Cybern 2011;2:107-122.

[34] Niu WJ, Feng ZK, Feng BF, Min YW, Cheng CT, Zhou JZ. Comparison of multiple linear regression, artificial neural network, extreme learning machine, and support vector machine in deriving operation rule of hydropower reservoir. Water 2019;11(1):88.
[35] Diker A, Sönmez Y, Özyurt F, Avcı E, Avcı D. Examination of the ECG signal classification technique DEA-ELM using deep convolutional neural network features. Multimed Tools Appl 2021;80:24777-24800.
[36] Engelbrecht AP. Computational intelligence: An introduction. Wiley; 2007.
[37] Karcı A. Differential evolution algorithm and its variants. Comput Sci 2017;2(1):10-14.
[38] Qin AK, Huang VL, Suganthan PN. Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Trans Evol Comput 2008;13(2):398-417.
[39] Yang WA, Zhou Q, Tsui KL. Differential evolution-based feature selection and parameter optimisation for extreme learning machine in tool wear estimation. Int J Prod Res 2016;54(15):4703-4721.
[40] Krstinić D, Braović M, Šerić L, Božić-Štulić D. Multi-label classifier performance evaluation with confusion matrix. Comput Sci Inf Technol 2020;1:1-14.
[41] Hannousse A, Yahiouche S. Web page phishing detection. Mendeley Data 2021;3.
[42] Krstinić D, Braović M, Šerić L, Božić-Štulić D. Multi-label classifier performance evaluation with confusion matrix. Comput Sci Inf Technol 2020;1:1-14.
[43] Abdalla G, Özyurt F. Sentiment analysis of fast food companies with deep learning models. Comput J 2021;64(3):383-390.
[44] Subasi A, Mitra A, Ozyurt F, Tuncer T. Automated COVID-19 detection from CT images using deep learning. In: Comput-Aided Des Diagn Methods Biomed Appl. 2021. p. 153-176.
[45] Tuncer T, Aydemir E, Ozyurt F, Dogan S. A deep feature warehouse and iterative MRMR based handwritten signature verification method. Multimed Tools Appl 2022;81:3899-3913.
[46] Kumar S, et al. Mlspd-machine learning based spam and phishing detection. In: Int Conf Comput Social Netw. Springer; 2018.
[47] Babagoli M, Pourmahmood Aghababa M, Solouk V. Heuristic nonlinear regression strategy for detecting phishing websites. Soft Comput 2019;23(12):4315-4327.
[48] Rao RS, Pais AR, Anand P. A heuristic technique to detect phishing websites using TWSVM classifier. Neural Comput Appl 2021;33(11):5733-5752.
[49] Alsariera YA, et al. AI meta-learners and extra-trees algorithm for the detection of phishing websites. IEEE Access 2020;8:142532-142542.
[50] Azeez NA, et al. Adopting automated whitelist approach for detecting phishing attacks. Comput Secur 2021;108:102328.
[51] Rashid J, et al. Phishing detection using machine learning technique. In: 2020 1st Int Conf Smart Syst Emerg Technol (SMARTTECH). IEEE; 2020.
[52] Sahingoz OK, et al. Machine learning based phishing detection from URLs. Expert Syst Appl 2019;117:345-357.
[53] Sindhu S, et al. Phishing detection using random forest, SVM and neural network with backpropagation. In: 2020 Int Conf Smart Technol Comput Electr Electron (ICSTCEE). IEEE; 2020.
[54] Prince MSM, Hasan A, Shah FM. A new ensemble model for phishing detection based on hybrid cumulative feature selection. In: 2021 IEEE 11th Symp Comput Appl Ind Electron (ISCAIE). IEEE; 2021.
[55] Anupam S, Kar AK. Phishing website detection using support vector machines and nature-inspired optimization algorithms. Telecommun Syst 2021;76(1):17-32.