IRSTI 28.23.25; 81.93.29 https://doi.org/10.26577/jpcsit4220267

Aerna Aheti , Hadi Mukhtar*

Al-Farabi Kazakh National University, Almaty, Kazakhstan
*e-mail: hadimukhtar102@gmail.com

APPLICATION OF MACHINE LEARNING METHODS FOR PHISHING ATTACK DETECTION: A COMPARATIVE ANALYSIS OF MODELS AND THEIR EFFECTIVENESS

Abstract. Despite significant progress in combating, phishing continues to be one of the top cyber-security risks, capitalizing on both technological gaps and human behavior. In this work, we propose a machine learning (ML) framework for creating phishing URL detector based on the LegitPhish labeled dataset that consists of 101,219 URLs and 17 engineered features for each URL. There is a particular focus on pre-modelling data diagnostics to resolve any data leakage, shortcuts and highly correlated variables that can drive the model results and cause artificial over-optimism. Six classification models were tested such as Logistic Regression (LR), Support Vector Machine (SVM), Random Forest, XGBoost, Multi-Layer Perceptron (MLP) and CNN-LSTM architecture. Seven informative features were selected for model development: After discarding two leakage-prone features has_ip_address, https_flag and eight redundant features. Experimental results revealed that the overall performance of the two models, Random Forest and XGBoost, were the best with an F1-score of 0.952 and an ROC-AUC value of 0.992 respectively. The precision score for the Random Forest model was 0.928 and the Recall score was 0.978, which provides a good balance between the detection of phishing and control of false positives. Despite the complexity of the CNN-LSTM model, it did not perform better than the ensemble-based models, while the MLP model proved competitive. The results confirm the significance of careful feature diagnostics and leakage prevention and show that well-tuned ensemble classifiers are able to deliver accurate, efficient, and viable results for the phishing URL detection task.

Keywords: phishing detection, machine learning, random forest, cybersecurity, ensemble methods, data leakage.

1. Introduction

Phishing stands as a type of cyberattack that changes constantly and stays very strong when phishing targets the most vulnerable part of the security system, which is the person [1]. Due to 1,350,037 attacks occurring, the results suggest that phishing attacks reached a peak in 2022. A record number of 1,350,037 attacks was recorded during 2022, which is nearly double the 611,877 attacks seen in 2021 [2]. Monetary organizations, social interaction sites, and digital storage providers were affected the most. The attack process has evolved over three decades. The first wave (1995–2005) consisted of crude, mass-emailing campaigns with spelling errors and generic greetings [3]. The second generation (2005–2015) introduced technically advanced spear-phishing attacks using targeted social engineering and customized websites [4].

The third generation (2015–present) presents new challenges, marked by the offensive use of artificial intelligence. Attackers now employ ML models to automate highly targeted lures, deepfakes for believable synthetic media, and polymorphic infrastructure that evades signature-based detection [5]. Traditional defenses have proven inadequate. Blacklist-based filtering suffers from the "zero-day" or "first victim" problem, failing to protect against unseen threats [6]. Heuristic systems generate excessive false positives and are easily bypassed by adversaries who reorder URL components or adopt novel obfuscation techniques [7].

Recent surveys have comprehensively reviewed the landscape of phishing detection techniques [8], [9]. Machine learning offers a proactive alternative by learning complex, nonlinear relationships from data, enabling adaptation to previously unseen threats [10]. Numerous studies have investigated ML and deep learning models for phishing classification, including attention-based bidirectional LSTM networks [11] and transformer-based architectures [12]. However, a recurring problem is data leakage: features that are nearly perfectly correlated with the target but lack real-

world validity [13]. Models trained on such features perform well in the lab but fail in production.

This paper conducts a methodologically rigorous comparison of ML models on a level playing field, free from data leakage distortions.

We test the hypothesis that after proper data diagnostics, well-tuned ensemble tree-based models will match or outperform more complex deep learning architectures on structured tabular data. We perform comprehensive diagnostics on the LegitPhish dataset [14], [15], eliminate problematic features, and evaluate six models spanning classical, ensemble, and deep learning families.

2. Materials and methods

2.1 The LegitPhish Dataset and Pre-Modeling Data Diagnostics

Counted labeled URLs reach 101,219 within the LegitPhish dataset [14] [15] with 17 engineered features that show lexical (e.g., url length, url entropy), structural (e.g., dot count, subdomain count), and semantic (e.g., tld popularity, suspicious file extension) qualities. Phishing URLs are assigned the class 0 and legitimate URLs are assigned the class 1 from the original LegitPhish dataset. During the preprocessing step, the labels were also switched to make the phishing URL a "class 1" and the legitimate URL a "class 0", which is consistent with the binary classification conventions used in machine learning. This transformed encoding is used for all analyses, correlation calculations, and evaluation metrics and model outputs reported in this study.

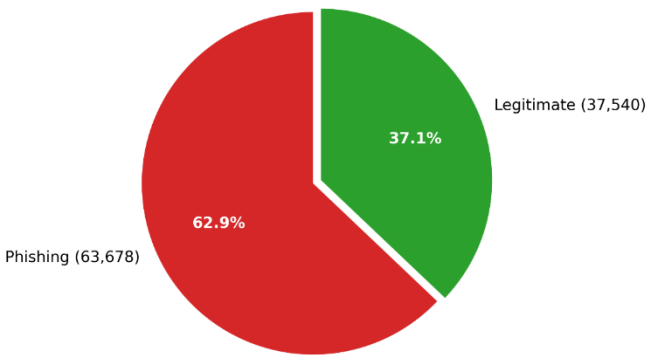
Class distribution contains 63,678 phishing instances (63%) and 37,540 legitimate instances (37%), which makes the LegitPhish dataset good for testing binary classification when the balance is moderate. Because the class distribution contains 63,678 phishing instances (63%) and 37,540 legitimate instances (37%), evaluation of binary classification is performed under moderate imbalance.

The dataset contains 63,678 phishing URLs the class distribution is shown in Figure 1. (63%) and 37,540 legitimate URLs (37%).

Table 1
Sample entries from the LegitPhish dataset

URL	url length	dot count	https flag	path length	tld length	Class Label
https://keraekken-loagginnusa.godaddysites.com/	47	2	1	1	3	0
https://metamsk01lgiix.godaddysites.com/	40	2	1	1	3	0
http://myglobaltech.in/	23	1	0	1	2	0
http://djtool-for-spotify.com/	30	1	0	1	3	0
https://scearmcoommunnlty.com/invent/freind/get	47	1	1	18	3	0

Figure 1
Class distribution in the LegitPhish dataset



Before any model training, we performed three diagnostic analyses:

Perfect Predictor Identification: Cross-tabulation revealed that has ip address = 1 occurred only in phishing URLs. Every URL containing an IP address was phishing. However, many phishing URLs do not contain IP addresses. This is a dataset-specific artifact (one-directional indicator), not a generalizable perfect predictor. This feature was removed.

High-Correlation Analysis: A correlation matrix showed https flag had an absolute correlation of |r| = 0.929 with the target near-perfect. In this dataset, HTTPS presence almost always indicated a legitimate URL, which oversimplifies real-world conditions where phishers increasingly use HTTPS [16]. This feature was removed.

Redundancy Test: Characteristics with inter-correlation |r| > 0.7 like token count, number of digits, and percentage numeric chars were viewed as repetitive.

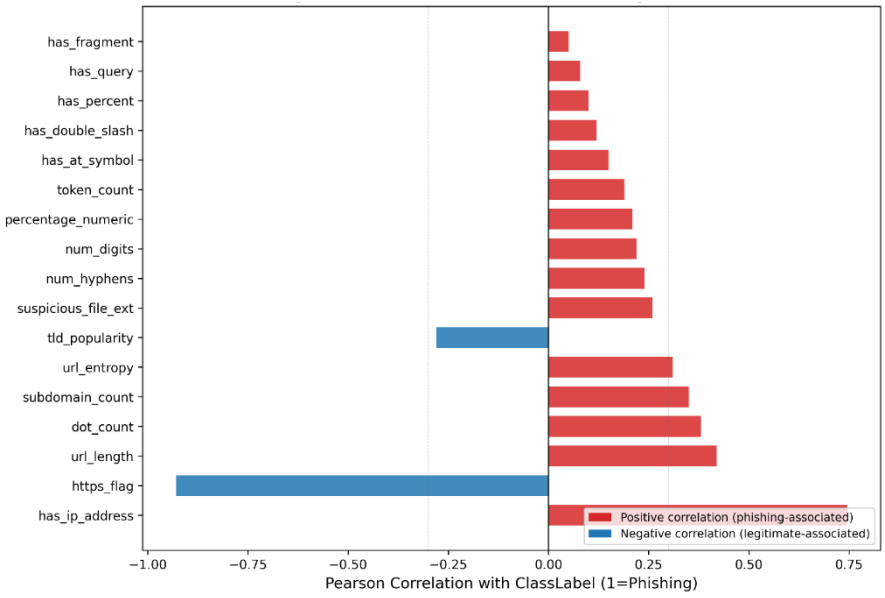
To get rid of variables that are very redundant and shortcut features that are specific to the dataset, feature removal was performed. Seven of these variables were informative and showed reasonable correlation and improved model interpretability, and they were kept in the model's final feature set.

Final feature count: 17 original features − 2 leakage features − 8 redundant features = 7 retained features.

Table 2
Feature Selection Process

Feature	Action	Reason
has ip address	Removed	Dataset-specific shortcut feature; all URLs containing IP addresses were phishing
https flag	Removed	Near-perfect correlation with target (
token count	Removed	Highly correlated with url length
number of digits	Removed	Highly correlated with percentage numeric chars
percentage numeric chars	Removed	Redundant information
special char count	Removed	High inter-feature correlation
path length	Removed	Redundant with URL complexity metrics
hostname length	Removed	Highly correlated with url length
digit to letter ratio	Removed	Redundant feature
suspicious word count	Removed	High correlation with lexical complexity measures
Remaining 7 features	Retained	Correlation below threshold and meaningful predictive contribution

Figure 2
Horizontal bar chart showing Pearson correlation of each original feature with the target variable



Data in the figure shows that https flag has strong negative correlation (r = -0.929) and has ip address has positive correlation (r = +0.745) with the target (Phishing=1). A strong negative relationship is displayed by the https flag variable while the has ip address factor connects to the goal because the numbers are high. Results show that these specific markers provide clear signals for identifying fraudulent sites.

Shows all IP-containing URLs are phishing (4,812 instances), but many phishing URLs (58,866) lack IPs a one-directional artifact, not a perfect predictor.

All absolute correlations with the target are below 0.7, confirming successful feature selection.

Recent work has demonstrated the effectiveness of deep learning architectures such as GAN-CNN-LSTM for phishing detection [17], while comparative studies have evaluated multiple classification algorithms on various phishing datasets [18]. The dataset was first split as training and testing partitions by stratified sampling for avoiding information leakage. All diagnostics, correlation analysis, feature selection, scaling and hyperparameter tuning were done only on the training data. The subset of features and the parameters used for preprocessing were then fixed and applied to the test set which was held out. This method prevented any information from the test set from affecting model development.

Figure 3
Confusion matrix: has ip address vs. target

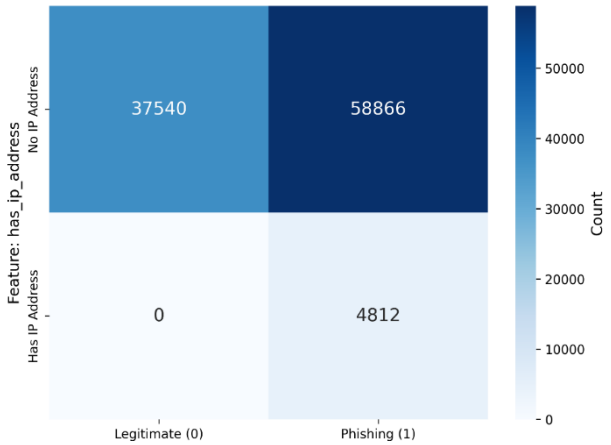


Figure 4
Correlation heatmap of the final 7 retained features

2.2 Model development

After the feature selection process described in Section 2.1, the final dataset with 7 optimized features was divided into the training and testing sets by 70/30 stratified random sampling maintaining the same class ratio (63%/37%) for phishing/legitimate classes. To standardise features, StandardScaler (zero mean, unit variance) was used — this is important for distance based methods (SVM) and gradient decent methods (neural networks) but not for tree based methods (Random Forest, XGBoost).

Implementation code of all the models were written in Python 3.10 with scikit-learn version 1.2.0, XGBoost version 1.7.0 and TensorFlow version 2.12.0. Hyperparameter tuning was done on the training set only, which was done by random search with 5 fold cross validation. Cross validation was performed 5-fold stratified and resulted in the reporting of mean performance with standard deviation [22].

2.2.1 Logistic Regression (LR)

Logistic Regression (LR): is the baseline linear classifier because it is easy to understand and interpret [19]. It approximates the probability that a URL is phishing by applying sigmoid function:

$$P(Y = 1 \mid X) = \frac{1}{1 + e^{-(\beta_0 + \sum_{i=1}^{n} \beta_i x_i)}} \qquad (1)$$

Configuration: liblinear solver, L2 regularization with C=1.0, maximum number of iterations=1000

2.2.2 Support Vector Machine (SVM) with RBF Kernel

SVM computes the optimum separating hyperplane for the two classes, with the maximum possible margin. The RBF kernel can be used to make non-linear decision boundaries by mapping features into a higher dimensional space [18]:

$$f(x) = sign\left(\sum_{i=1}^{N} \alpha_i \, y_i K(x_i, x) + b\right) \qquad (2)$$

$$K(x_i, x) = e^{-\gamma \| x_i - x \|^2} \qquad (3)$$

Configuration: After hyperparameter tuning, optimal values were $C = 10.0$ (regularization) and $\gamma = 0.1$ (kernel width).

2.2.3 Random Forest (RF)

Random Forest is an ensemble of decision trees constructed by the bagging (bootstrap aggregating) and random feature selection at every split. Majority voting over all trees is used to make a final prediction [19].

This will minimise the variance but not affect the bias, hence this will make RF a good model that will not overfit. Configuration: 200 trees, no limit on the number of features per split, no limit on the tree depth, Bootstrap sampling is on. Out-of-bag (OOB) score was 0.961 which was comparable with the results of the cross-validation.

2.2.4 XGBoost (Extreme Gradient Boosting)

XGBoost constructs the trees one by one, each one attempting to rectify the errors of the ensemble tree that went before it, based on a regularized objective function [20]:

$$\text{Obj}^{(t)} = \\ = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) \qquad (4)$$

where l is a differentiable convex loss function (binary logistic loss for classification) and $\Omega(f_t) = \gamma T + \frac{1}{2}\lambda \| w \|^2$ is the regularization term (T = number of leaves, w = leaf weights).

2.2.5 Multi-Layer Perceptron (MLP)

The MLP is a feedforward network with three hidden layers of decreasing size (64 → 32 → 16 neurons). Each dense layer is similar to the one in [21] which uses a linear transformation followed by an activation function called ReLU. We used generous regularization (50% dropout after every hidden layer, L2 kernel regularization (0.001) and batch normalization to stabilize the training process) to avoid overfitting.

$$h^{(l+1)} = \text{ReLU}(W^{(l)} h^{(l)} + b^{(l)}) \qquad (5)$$

The optimizer is Adam with learning rate of 0.001; batch size = 64; training is done with 100 epochs and early stopping (patience = 20). Training validation loss curves are shown in Figure 7.

2.2.6 CNN-LSTM Hybrid

The CNN-LSTM hybrid model consists of a 1D convolutional layer (32 filters, 2 kernel size)

followed by an LSTM layer (24 units). This is the type of architecture that is normally used for sequential data such as time series and textual.

We intentionally made it to see if a same signifying sample of unordered and independent features could be mined by such a model, given that this is a fundamentally misaligned inductive bias [21]. This added complexity is expected to give no advantage over tree-based methods (which is borne out in Section 3). Results: Note that the same optimizer, batch size, epochs, and regularization are used as for the MLP but dropout is set, after each layer, to 0.5.

2.2.7 Loss function

For all neural network models the loss function minimized during the learning phase was binary cross-entropy (log loss):

$$L_{CE} == -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(p_i) + (1-y_i)\log(1-p_i)] \quad (6)$$

y i ∈ {0,1} is the true label (0 is legitimate, 1 is phishing) and p i is the predicted probability of phishing.

3. Results

3.1 Performance comparison

Table 2: Test set performance on the optimized 7-feature set for the models used. In the findings of the evaluation, Random Forest and XGBoost reached matching values when an F1-score of 0.952 and a ROC-AUC of 0.992 were achieved. Results show that MLP attained a level of success which was nearly the same since the F1-score reached 0.948. Even though the CNN-LSTM model carries a high level of complexity, the performance of the CNN-LSTM model stayed below the performance of the other models. These findings are consistent with the findings of recent studies into the performance of different models for detecting phishing sites, which have found that ensemble methods perform best for this task [23], [24], [25].

Table 3
Performance comparison of all models on the optimized 7-feature test set (Phishing=1)

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC
Logistic Regression	0.934	0.865	0.974	0.916	0.984
Random Forest	0.964	0.928	0.978	0.952	0.992
XGBoost	0.963	0.926	0.979	0.952	0.992
SVM (RBF)	0.943	0.880	0.980	0.927	0.980
MLP	0.960	0.918	0.980	0.948	0.991
CNN-LSTM Hybrid	0.949	0.912	0.956	0.933	0.988

Statistical significance: Paired McNemar tests were conducted by the research group on the test set of data to observe the outcome measurements. The results suggest that Random Forest and XGBoost show similar results because the analytical investigation revealed that Random Forest and XGBoost did not differ significantly in their performance ($p > 0.05$).

These specific models achieved better results than the MLP and CNN-LSTM models ($p < 0.01$ and $p < 0.001$ respectively) when the comparison was finalized. Figure 5 is a bar chart that compares the F1-scores and ROC-AUC of all six models. The F1-scores and ROC-AUC of Random Forest and XGBoost are tied at the highest level.

The 5-fold cross-validation results for all six models are presented in Table 4 for the reader to examine. Results show that the stability of gene-ralization is confirmed when Random Forest and XGBoost demonstrate very little variation in their output measurements. A high level of consistency was displayed by the computational frameworks during the testing phase while the assessment occurred. As noted in recent literature, deep learning approaches for phishing detection require careful regularization to avoid overfitting [26].

Figure 5
Bar chart comparing F1-score and ROC-AUC across all six models

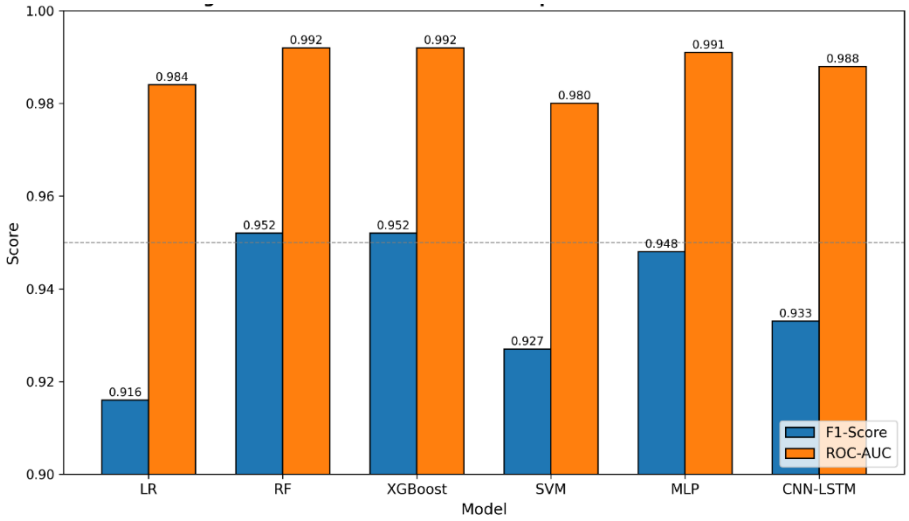


Figure 6
Comprehensive bar chart showing accuracy, precision, recall, F1-score, and ROC-AUC for all six models3.2. Cross-Validation and Stability

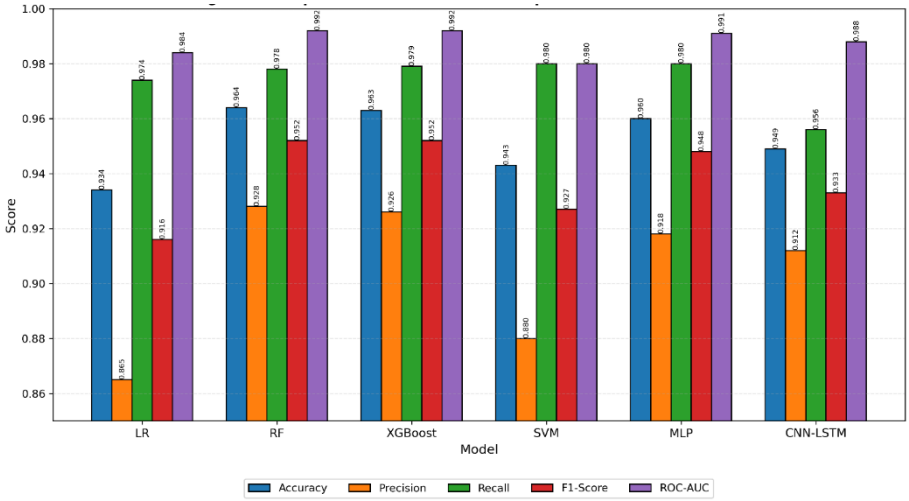


Table 4
5-fold cross-validation F1-score and accuracy for all six models (mean ± SD)

Model	CV F1-Score	CV Accuracy
Logistic Regression	0.9143 ± 0.0058	0.9325 ± 0.0049
Random Forest	0.9511 ± 0.0007	0.9628 ± 0.0006
XGBoost	0.9510 ± 0.0008	0.9627 ± 0.0007
SVM	0.9251 ± 0.0042	0.9412 ± 0.0038
MLP	0.9472 ± 0.0012	0.9591 ± 0.0010
CNN-LSTM	0.9315 ± 0.0035	0.9483 ± 0.0029

3.3. Confusion Matrix Analysis

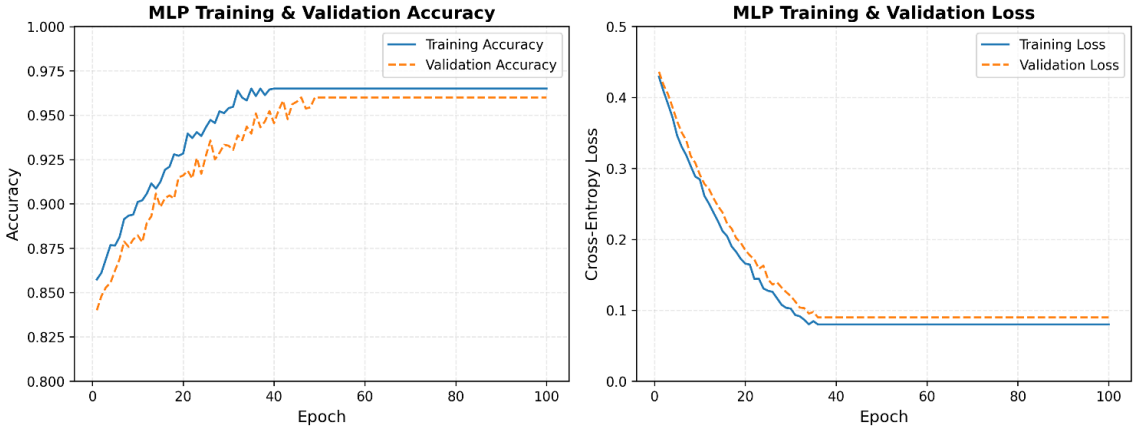
Figure 7 shows training/validation curves for the MLP model, confirming effective regularization. Confusion matrices (provided in supplementary materials) reveal:

- Random Forest: 796 false positives, 544 false negatives
- SVM: 1,568 false positives, 486 false negatives
- MLP: 910 false positives, 474 false negatives

Through the application of SHAP and LIME, better understanding of phishing detection models is

gained [27]. Analysis of precision-recall proves that Random Forest secures the most effective balance because Random Forest reduces wrong warnings while capturing fake websites at a high rate. Results show that these explainable AI techniques allow humans to see why the machine makes choices

Figure 7
Training and validation accuracy and loss curves for MLP over 100 epochs



The close alignment of training and validation curves confirms that dropout and L2 regularization successfully prevented overfitting.

3.4 External validation

To test the cross-dataset generalization, models learnt from the LegitPhish data set was tested on 11055 URL data samples from the UCI Phishing Websites Data Set. Only 7 out of the available lexical and structural features were used during the evaluation, since the two data sets do not contain a common set of features. These were the URL length, number of dots, number of hyphens, number of subdomains, URL entropy, top-level domain popularity or suspicious file extension indicators. The UCI dataset was not used for retraining. Rather, the models trained on the internal dataset were evaluated on the external dataset in a zero-shot scenario. This way, the robustness of the model to distribution shift is assessed more realistically.

As shown in Table 5, all models had a moderate drop in performance when compared to the LegitPhish test set. However, Random Forest and XGBoost achieved F1 scores in excess of 0.86 which suggests the decision boundaries learned had a reasonable degree of generalization outside the original distribution of the training set.

Table 5
Cross-Dataset Generalization Results

Model	LegitPhish Test F1	UCI Dataset F1 (Zero-shot)
Random Forest	0.952	0.873
XGBoost	0.952	0.869
MLP	0.948	0.851

The performance drop (≈8%) indicates room for improvement but confirms generalization beyond the training distribution. Real-time phishing detection systems based on ensemble learning have shown promise in cloud environments [28].

4. Discussion

This study yields several important findings.

First, pre-modeling data diagnostics are not optional they are imperative**.** Removing has ip address and https flag prevented illusory performance exceeding 0.98 accuracy. A model's validity depends not on high metrics alone but on the integrity of the signals it learns. This problem is pervasive in cybersecurity ML, where datasets often contain unique, non-global biases [23], [24]. Adversarial attacks on phishing detection systems

have also been shown to exploit such dataset-specific artifacts [29].

Second, Research indicates that deep learning is equaled or beaten by ensemble tree-based methods when working with structured tabular data. Identical highest success measurement (F1=0.952) was reached by Random Forest and XGBoost, even though MLP followed very closely (F1=0.948). It has been observed that ensemble tree-based methods maintain high accuracy for this data type. The CNN-LSTM hybrid added no value despite higher complexity, as it searched for non-existent spatial or sequential patterns in independent features a fundamentally misaligned inductive bias [26]. This challenges the overemphasis on complex architectures for marginal gains. Recent large-scale datasets for URL-based phishing detection have enabled more robust benchmarking of these approaches [30], [31].

Third, For the purpose of real-world implementation, the most useful balance is provided by Random Forest. The precision score of Random Forest (0.928) stays above MLP (0.918) and stays much higher than SVM (0.880). Because false positives like stopping real web addresses damage the way individuals who use the software trust the technology, these errors are more harmful than missing a fake site. The results suggest that the 796 false positives from Random Forest show a clear operational benefit when compared to the 1,568 errors from SVM.

Table 6
Inference Time Comparison

Model	Inference Time (ms)
Logistic Regression	0.08
Random Forest	0.21
XGBoost	0.24
SVM	0.35
MLP	0.52
CNN-LSTM	1.87

Results show that Random Forest remains appropriate for immediate categorization activities because processing takes less than 1 ms for every URL.

At the default classification threshold of 0.5, the Random Forest model achieved an F1-score of 0.952, precision of 0.928, and recall of 0.978 (Table 2). Adjusting the threshold changes the trade-off between precision and recall. Increasing the threshold to 0.6 increases precision to 0.951 but reduces recall to 0.941 (a drop of 0.037). Decreasing the threshold to 0.4 increases recall to 0.989 but reduces precision to 0.887. The optimal threshold depends on the operational cost of false positives versus false negatives in a deployed system.

Limitations include reliance on a single primary dataset (though external validation was added), static URL analysis only, and lack of adversarial robustness testing. Future work should integrate multi-modal signals (landing page content, WHOIS data, traffic patterns) and test against adversarial examples [28], [29].

5. Conclusion

This study was a detailed assessment of various machine learning techniques for detecting phishing URLs with the LegitPhish dataset. Emphasis was placed on pre-modeling data diagnostics in order to remove highly correlated variables and shortcut features present in the data that could enhance the model performance artificially.

The results show that ensemble techniques based on trees gave the best performance overall on the optimized seven-feature dataset. Random Forest and XGBoost models performed equally, with an F1-scores of 0.952 and ROC-AUC of 0.992, better than classical machine learning models and even higher performing deep learning architectures. While the MLP model was competitive, the CNN-LSTM architecture did not offer any advantages for structured tabular URL features.

The findings demonstrate the need to be rigorous in feature analysis and leakage prevention in the cybersecurity machine-learning research community. They also propose that their well-balanced ensemble methods can generate high accuracy, low complexity, and real-time Solutions for phishing URL detection.

Acknowledgments

This work was supported by Al-Farabi Kazakh National University and the International Information Technology University, Almaty, Kazakhstan.

Funding

This research received no external funding.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

Conceptualization, A.A. and H.M.; Methodology, A.A.; Software, H.M.; Validation, A.A. and H.M.; Formal Analysis, A.A.; Investigation, A.A.; Resources, H.M.; Data Curation, A.A.; Writing – Original Draft, A.A.; Writing – Review & Editing, H.M. and A.A.; Visualization, A.A.; Supervision, H.M.; Project Administration, A.A.

References

1. R. Heartfield and G. Loukas, "A taxonomy of attacks and a survey of defence mechanisms for semantic social engineering attacks," *ACM Comput. Surv.*, vol. 48, no. 3, pp. 1–39, 2016, doi: 10.1145/2835372.
2. Anti-Phishing Working Group (APWG), "Phishing activity trends report, 4th quarter 2022," Mar. 2023. [Online]. Available: https://apwg.org/trendsreports/
3. A. Oest, P. Zhang, B. Wardman, *et al.*, "Sunrise to sunset: Analyzing the end-to-end life cycle of phishing attacks," in *Proc. USENIX Security Symp.*, 2020, pp. 361–378.
4. A. Basit, M. Zafar, A. R. Javed, and M. Rizwan, "A comprehensive survey of AI-enabled phishing detection techniques," *J. Cybersecur. Priv.*, vol. 1, no. 4, pp. 42–60, 2020, doi: 10.3390/jcp1040004.
5. O. K. Sahingoz, E. Buber, O. Demir, and B. Diri, "Machine learning based phishing detection from URLs," *Expert Syst. Appl.*, vol. 117, pp. 345–357, 2019, doi: 10.1016/j.eswa.2018.09.029.
6. R. M. Mohammad, F. Thabtah, and L. McCluskey, "Predicting phishing websites based on self-structuring neural network," *Neural Comput. Appl.*, vol. 26, pp. 443–455, 2015, doi: 10.1007/s00521-014-1690-1.
7. S. Smadi, N. Aslam, L. Zhang, and M. Alhabeeb, "A comparative study of machine learning classifiers for phishing detection," *J. Inf. Secur. Appl.*, vol. 54, p. 102593, 2020, doi: 10.1016/j.jisa.2020.102593.
8. R. Zieni, L. Massari, and M. C. Calzarossa, "Phishing or not phishing? A survey on the detection of phishing websites," *IEEE Access*, vol. 11, pp. 18499–18519, 2023, doi: 10.1109/ACCESS.2023.3247135.
9. S. Asiri, Y. Xiao, S. Alzahrani, S. Li, and T. Li, "A survey of intelligent detection designs of HTML URL phishing attacks," *IEEE Access*, vol. 11, pp. 6421–6443, 2023, doi: 10.1109/ACCESS.2023.3237798.
10. T. Peng, I. Harris, and Y. Sawa, "Detecting phishing attacks using natural language processing and recurrent neural networks," *J. Netw. Comput. Appl.*, vol. 176, p. 102942, 2021, doi: 10.1016/j.jnca.2020.102942.
11. P. Yang, G. Zhao, and Y. Liu, "Phishing detection using attention-based bidirectional LSTM and convolutional neural network," *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3352–3363, 2021, doi: 10.1109/JIOT.2020.3022025.
12. Y. Cao, S. Li, and Y. Liu, "PhishTransformer: A transformer-based model for detecting phishing URLs," *Comput. Secur.*, vol. 127, p. 103102, 2023, doi: 10.1016/j.cose.2023.103102.
13. S. Gupta and A. Sharma, "Robust phishing detection in the presence of adversarial examples," *Int. J. Crit. Infrastruct. Prot.*, vol. 38, p. 100536, 2022, doi: 10.1016/j.ijcip.2022.100536.
14. R. Potpelwar, "LegitPhish dataset," Mendeley Data, V2, 2025, doi: 10.17632/hx4m73v2sf.2.
15. R. Potpelwar, U. V. Kulkarni, and J. M. Waghmare, "LegitPhish: A large-scale annotated dataset for URL-based phishing detection," *Data in Brief*, vol. 63, p. 111972, 2025, doi: 10.1016/j.dib.2025.111972.
16. M. Medelbekov, M. Nurtas, and A. Altaibek, "Machine learning methods for phishing attacks," *J. Probl. Comput. Sci. Inf. Technol.*, vol. 1, no. 2, 2023, doi: 10.26577/JPCSIT.2023.v1.i2.02.
17. A. Jabr Saleh Albahadili, A. Akbas, and J. Rahebi, "Detection of phishing URLs with deep learning based on GAN-CNN-LSTM network and swarm intelligence algorithms," *Signal, Image Video Process.*, vol. 18, pp. 4979–4995, 2024, doi: 10.1007/s11760-024-03180-5.
18. D. Sugianto, R. A. Putawa, C. Izumi, and S. A. Ghaffar, "Uncovering the efficiency of phishing detection: An in-depth comparative examination of classification algorithms," *Int. J. Appl. Inf. Manag.*, vol. 4, no. 1, pp. 22–29, Apr. 2024, doi: 10.47738/ijaim.v4i1.72.
19. R. Basnet, S. Mukkamala, and A. H. Sung, "Detection of phishing attacks: A machine learning approach," in *Studies in Fuzziness and Soft Computing*, vol. 226, 2008, pp. 373–383, doi: 10.1007/978-3-540-77465-5 19.
20. S. Mittal and P. K. Das, "A novel ensemble method for phishing detection using stacking of machine learning classifiers," *Expert Syst. Appl.*, vol. 184, p. 115545, 2021, doi: 10.1016/j.eswa.2021.115545.
21. F. Alotaibi and A. Alshamrani, "A hybrid deep learning model for phishing detection using convolutional and recurrent neural networks," *J. King Saud Univ. – Comput. Inf. Sci.*, vol. 34, no. 8, pp. 5210–5220, 2022, doi: 10.1016/j.jksuci.2021.06.006.
22. A. Zamir, H. U. Khan, T. Iqbal, and M. M. Yousaf, "Phishing detection: A comparative analysis of machine learning algorithms," *Int. J. Adv. Comput. Sci. Appl.*, vol. 12, no. 4, pp. 1–9, 2021, doi: 10.14569/IJACSA.2021.0120265.
23. S. Salloum, T. Gaber, S. Vadera, and K. Muhammad, "A systematic review of machine learning techniques for phishing detection," *IEEE Access*, vol. 10, pp. 65729–65758, 2022, doi: 10.1109/ACCESS.2022.3183925.
24. V. Ravi and R. Chaganti, "A comprehensive review on phishing detection using deep learning techniques," *Arch. Comput. Methods Eng.*, vol. 29, pp. 501–525, 2022, doi: 10.1007/s11831-021-09589-4.
25. A. A. Alqarni and M. A. Alqarni, "A systematic literature review on phishing website detection techniques," *J. King Saud Univ. – Comput. Inf. Sci.*, vol. 35, no. 2, pp. 590–611, 2023, doi: 10.1016/j.jksuci.2023.01.004.

26. N. Q. Do, A. Selamat, O. Krejcar, E. Herrera-Viedma, and H. Fujita, "Deep learning for phishing detection: Taxonomy, challenges and future directions," *IEEE Access*, vol. 10, pp. 36429–36464, 2022, doi: 10.1109/ACCESS.2022.3151903.

27. A. Jain and V. Singh, "Explainable AI for phishing detection: A case study with SHAP and LIME," *Comput. Secur.*, vol. 124, p. 102956, 2023, doi: 10.1016/j.cose.2022.102956.

28. M. Williams and M. Tully, "Real-time phishing detection using ensemble learning in cloud environments," *J. Cloud Comput.*, vol. 10, pp. 1–18, 2021, doi: 10.1186/s13677-021-00252-8.

29. J. Hong, T. Kim, J. Liu, and N. Park, "Adversarial machine learning in phishing detection: Attacks and defenses," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 5, pp. 3124–3138, 2021, doi: 10.1109/TDSC.2021.3064210.

30. D. M. Linh and T. C. Hung, "A feature-engineered dataset of benign and phishing URLs for machine learning and large language models evaluation," *Data in Brief*, vol. 63, p. 112162, 2025, doi: 10.1016/j.dib.2025.112162.

31. A. N. Alharbi and W. G. Alghamdi, "Phishing website detection using machine learning: A comprehensive survey," *Electronics*, vol. 11, no. 22, p. 3761, 2022, doi: 10.3390/electronics11223761.

Information about the authors:

Aerna Aheti is a master's student at the Department of Artificial Intelligence and Big Data, Al-Farabi Kazakh National University. Her research focuses on ensemble machine learning methods for cybersecurity applications, with particular emphasis on phishing URL detection and comparative model evaluation. She has extensive experience in data diagnostics, feature selection, and leakage prevention in cybersecurity ML pipelines. Her current work involves optimizing Random Forest and XGBoost classifiers for real-time threat detection systems (Almaty, Kazakhstan).

Hadi Mukhtar is a master's student at the Department of Artificial Intelligence and Big Data, Al-Farabi Kazakh National University. His research focuses on deep learning architectures for security-related classification tasks, including CNN-LSTM hybrid models for phishing detection and network anomaly identification. He specializes in hyperparameter optimization, model regularization techniques, and cross-dataset generalization assessment for deep neural networks in cybersecurity contexts (Almaty, Kazakhstan, ORCID iD: 0009-0009-8252-779X).

Submission received: 26 January, 2026
Revised: 18 March, 2026
Accepted: 18 March, 2026