The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 9, September 2025
PP 374–384

Intelligent Malicious URL Detection Using Deep Reinforcement Learning and Feature Optimization

[1] Asanapuram Mohan Babu. [2] Vunnava Dinesh Babu,

[1]PG Scholar, Department of Computer Science and Engineering, R V Institute of Technology, Autonomous, Chebrolu, Chebrolu Mandal,Guntur,. Andhra Pradesh, India
mohanphy57@gmail.com

[2]Associate Professor, Department of Computer Science and Engineering, R V Institute of Technology, Autonomous, Chebrolu, Chebrolu Mandal,Guntur,. Andhra Pradesh, India.
dineshbabuvunnava@gmail.com

Abstract:

One serious problem with network security is the prevalence of online programs that steal data by pretending to be genuine platforms.
The vast majority of websites are legitimate, so countermeasures based on artificial intelligence (AI) are often used to identify malicious ones.
Here, deep reinforcement learning (DRL) stands up as a promising area for building network intrusion detection models, especially when dealing with very skewed class distributions. But DRL's training time grows exponentially with the complexity of the data. The goal of this research is to expedite training without sacrificing accuracy in classification by integrating a DRL-based classifier with cutting-edge feature selection methods. Five distinct feature-ranking algorithms based on statistics and correlation were utilized in our research on the Mendeley dataset. Compared to the scenario without selection techniques, the findings showed that the selection approach based on the computation of the Gini index greatly improved classification scores, reduced the number of columns in the dataset by 27%, and saved over 10% of training time.

Keywords: Cybersecurity AI, DL, Feature Selection, Malicious URL, and Web Phishing

I. INTRODUCTION

Access to online resources is now more convenient than ever before, thanks to the proliferation of mobile devices. A web application may be accessed from any device by simply clicking on its access reference, which is the universal resource location (URL). To steal critical information, bad actors use this trend to trick users into visiting temporary and dangerous URLs. Web phishing is a prevalent phenomenon that poses a threat to network security [1]. Malware may also be sent using this mechanism [2]. Here, upstream detection of URLs that propagate harmful software is vital [6], even if research on preventing malware-induced computer network infection is constantly progressing [3]-[5]. The short duration of dangerous patterns makes stationary modeling useless when trying to distinguish between genuine and malicious URLs, which makes discriminating between the two exceedingly difficult [7]. Machine learning (ML) enables the fast design of classification algorithms that may proactively handle the challenge of identifying online phishing [8] by minimizing data idea drift. Deep learning (DL) offers very well-suited algorithms that are based on biological principles and fall within this category of domains [9]. Following its successful implementation in network security applications, new research into this field has shown the need of using the resources offered by deep reinforcement learning (DRL) [10]. In this regard, the most well-known DRL algorithm—deep Q-network (DQN)—was used as a classifier in [14] to identify online phishing attempts using a suitable Markov decision process (MDP) formulation. Since legal

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 9, September 2025
PP 374–384

URLs outweigh malicious ones in real-world applications, the generic learner that can handle online phishing detection might have been trained using data that suffers from class imbalance [15]-[17]. In our earlier work [18], we introduced a DRL-based classifier that could adapt its training phase to account for the uneven class distributions in the training data, which was a solution to this issue. It took a long time to train because of the worldview that was established. Decreasing the data's complexity might be one approach to tackling this expense. In most cases, you may pick and choose which training data characteristics to use throughout the development process [19]. In order to increase classification accuracy and minimize data complexity, which impacts time complexity, the most useful variables are picked and extraneous variables are removed [20]. Several research in the present literature have shown that combining feature selection techniques with ML algorithms for network intrusion detection improves their performance [21]-[23]. To the best of our knowledge, no proposal for DRL-based classifiers paired with feature selection processes is available to solve the challenge at hand. Consequently, this paper introduces a generic framework that utilizes the cost-sensitive DRL-based classifier from [18], merging it with lightweight feature selection strategies based on statistics and correlation [24], some of which have demonstrated promising results in detecting malicious URLs [25]. In sum, the following are the key themes discussed in this article:

• It offers proof that feature selection methods increase training time and classification performance.
• It integrates a DRL-based classifier with statistical and correlation-based feature selection approaches.

What follows is an outline of the rest of the paper. A review of the relevant literature on the topic of our investigation is included in Section II. Section III presents the suggested framework and describes its essential components. In Section IV, we outline the experimental strategy and critically assess the outcomes. Section V wraps up the study by summarizing its key findings and suggesting avenues for further research.

II. RELATED WORK

For ML-based phishing detection systems, the authors of [26] provide a two-stage feature selection method that works well. An ensemble function for data perturbation and an innovative cumulative distribution function gradient method are combined in the approach. The results of the studies demonstrated that competitive classifiers trained on the whole collection of features performed worse than a random forest (RF) technique using the set of attributes that were chosen. Using fuzzy rough set theory, the best features are chosen in [27]. When paired with the feature selection technique, RF outperformed all other ML classifiers that use a phishing detection mechanism in this instance. Two distinct feature selection strategies are contrasted in [28]. In the first, a manual technique was used to sort the characteristics into four groups based on their original kind. Then, all four sets of ML classifiers were tested. Second, there was a filter method that used a ranking system for all dataset features in order to exclude the ones with lower rankings. In one study, phishing detection systems were able to double their detection rate when just one set of characteristics was considered, thanks to human feature selection. One study looked at how well malicious URL detectors performed after using chi-square and analysis of variance (ANOVA) as feature selection procedures [29]. The first one uses what is often known as the "test of independence" to identify independent variables by calculating their correlation. In order to find the difference between the feature groups, the second one uses the F1 score as an average. This method has been shown effective in experiments for a voting classifier-based online phishing detection system. Another example of using the ANOVA approach to train an ML classifier is in [30], where the authors utilize a set of malicious URLs to determine which attributes are most important.

A acceptable accuracy score was achieved in this investigation using an extreme-boosting gradient technique. For feature selection in training an ANN-based classifier, a sinecosine method is used in [31]. Similarly, a feature selection technique that utilizes linear and non-linear space transformation methods improved the classification performance of an ANN in [32]. Another area that is investigated in [33] is feature selection for DL-based classifiers. Specifically, an evolutionary algorithm was used to determine the set of essential properties that may

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 9, September 2025
PP 374−384

optimize the malicious URLs detection system's performance. Oversampling harmful URLs was used to address the class imbalance in the entire URL dataset in [34]. Effective detection performance was obtained by integrating a chi-square feature selection method with an RF classifier. The feature selection approach employed in [35] is a multi-objective grasshopper optimization algorithm.

The selected classifier is an ANN of a specific kind that trains using the approach described in [36]. A hybrid ML model created by integrating logistic regression, support vector machine, and decision tree classifiers was used to enhance the performance of a dangerous URLs detector in [37]. The technique included using the canopy feature selection approach. An approach to feature selection based on particle swarm optimization (PSO) is expanded upon in [38] by taking the Laplacian score into account. Laplacian particle swarm optimization (LAPPSO) was one such method; it outperformed rival feature selection techniques based on PSO in terms of the ML classifiers that were examined. Presented in [39] is an expansion on a traditional feature selection method that is based on genetic algorithms (GAs). It all starts with computing information entropy, which either shows where evolution is headed or allows for a consensus method to choose which members of the present population are the most suitable. Combining the basic approach with several ML classifiers allowed it to surpass state-of-the-art alternative techniques. Its name is enhanced genetic algorithm with entropy and consensus (ECGA).

III. METHODOLOGY

The main points of the approach that this study presents are outlined in this section. We begin by outlining the evaluation and ranking procedures for the data characteristics. The next topic is the DRL-based classifier. Part A: Choosing Features

Some of the feature selection techniques that have shown promise in solving the online phishing detection issue are those mentioned in [25]:

• Eq. (1), which gives the formula for calculating such a coefficient [25], is the basis of the Pearson correlation coefficient (PCC)-based method:

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \overline{y})^2}} \quad (1)$$

The i-th sample of each conditional decision attribute is denoted by xi(yi), and the average of all sample points of each conditional decision attribute is x(y). The degree of relationship between two variables x and y is determined by the function rxy ∈ [−1, 1].

Typically, a negative positive correlation is shown by rxy → (−)1. We presume in this research that a larger correlation indicates a more significant effect of the feature on the classifier's choice, however in feature selection applications both positive and negative correlation typologies are taken into account.

Using Eq. (2) [24], the chi-square test assesses the independence between a class label and a generic feature (fi) using a statistical method:

$$\chi_c^2(f_i) = \sum_{j=1}^{t}\sum_{s=1}^{c}\frac{(n_{js} - \mu_{js})^2}{\mu_{js}} \quad (2)$$

The number of examples having the j−th feature, denoted as njs, is given by a feature fi that may take on t values. In addition, the above equation may be written as μjs = n∏snj∏n, where nj∏ is the number of data instances that have the j−th feature value from the set fi and n∏s is the number of samples from class s. Features with a high chi-square score—those that are strongly dependent—are kept when a selection technique is used.

Additionally, we take into account the following [24] while assessing feature selection strategies:

To find out whether a trait is statistically significant in differentiating between two groups, statisticians use the T-score. Here is how this measure is calculated:

$$t - score(f_i) = \frac{|\mu_1 - \mu_2|}{\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}} \quad (3)$$

In equation (3), the variables μ1(2), σ1(2), and n1(2) stand for the first and second class means, standard deviations, and number of occurrences, respectively. A high t-score indicates that the feature is preferred.

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 9, September 2025
PP 374–384

• The F-score, like in the previous section, is used to rank features; hence, a higher F-score indicates that the feature is more relevant [40]:

$$f-score(f_i)=\frac{\sum_{j=1}^{c}\frac{n_j}{c-1}(\mu_j-\mu)^2}{\frac{1}{n-c}\sum_{j=1}^{c}(n_j-1)\sigma_j^2} \quad (4$$

The F-score is determined for each of the c classes by adding together the sample size (n) and the number of occurrences in the j-th class (nj). In addition, the j-th class (μj) and the mean values of all classes (μ) are taken into account. Lastly, the j-th class's standard deviation is denoted by σj.

One way to measure a feature's ability to partition data into two sets, D and ∦D, is with the use of the Gini index. A lower Gini score indicates that the feature is relevant in the feature selection process, as stated in the following formula [24]:

$$gini\_index\_score(f_i)=\min_{\mathcal{D}}(p(\mathcal{D})(1-\sum_{s=1}^{c}p(C_s|\mathcal{D})^2)+p(\bar{\mathcal{D}})(1-\sum_{s=1}^{c}p(C_s|\bar{\mathcal{D}})^2)) \quad (5)$$

where $p(C_s|\bar{\mathcal{D}})^2)$ is the conditional probability of class C_s given $\mathcal{D}$.

B. Deep Reinforcement Learning-based detector

The DRL-based classifier suggested in [18] is used in the methods offered in this research. One of reinforcement learning's (RL) strongest points is how well it can adapt to decision-making difficulties, which opens up a lot of potential uses [41]-[45]. This is achieved because the task at hand is typically conceptualized as an MDP, described by the tuple < S,A, T , r, ζ >, where: (i) S is the observation space; (ii) A is the action space; (iii) T is the state-transition function, such that T : S×A×S → [0, 1] that describes the probability of observing state st, taking action at and producing a new state st+1; (iv) r is the reward, defined at each timestep t as rt = fR(st, at); (v) ζ ∈ [0, 1] is the discount factor that balances current and future rewards. The policy π : S → P(A), where P(A) is the probability distribution on A, has to learn the behaviors that maximize the reward via training, which is a process of trial and error.

The agent learns π during this phase by exploring its environment and choosing an action that maximizes the Q-function Q(st, at) = E[P∞ j=0 ÕjRt+j+1] for each observation (or state) in S and action in A. A cost-sensitive double deep Q-network (DDQN) agent utilizes the unbalanced classification Markov decision process (ICMDP) as outlined in [47] to form our classifier. Because of the ICMDP formulation, this model may adjust learning and reduce bias caused by the dominance of the majority class: (i) The training data provides S. (ii) A is the set of decisions that the agent can infer (|A| = 2 in this case of binary classification). (iii) T is deterministic because the agent moves from st to st+1 according to the order of samples in S. (iv) The reward function needs to inform the agent about the quality of the classification action by comparing at with the true label lt ∈ {0, 1} of the observation st and giving a scalar as feedback.

$$r_t=f_R(s_t,a_t,l_t)=\begin{cases}1, & a_t=l_t \text{ and } s_t\in\mathcal{S}_P\\ \rho, & a_t=l_t \text{ and } s_t\in\mathcal{S}_N\\ -1, & a_t\neq l_t \text{ and } s_t\in\mathcal{S}_P\\ -\rho, & a_t\neq l_t \text{ and } s_t\in\mathcal{S}_N\end{cases} \quad (6)$$

In equation (6), SP(N) stands for the positive or negative class's training data set, and the imbalance ratio, denoted as ρ = |SP | |SN|, is what the agent gets back when they make a mistake with their classification. Since it contains vectorized malicious URLs, we presume that the positive class is in the minority. As a result, in reaction to a classification action, the student may differentiate the sample distribution between classes based on the absolute value of rt. In particular, there will be a larger absolute value for minority class recognition and a lower value for majority class recognition depending on whether the categorization is valid or incorrect. The agent's ultimate objective is to master a categorization strategy that can maximize rewards in the long run. Because it decouples action selection and assessment during the target value calculation, the DDQN agent was able to decrease the phenomena

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 9, September 2025
PP 374−384

of overestimation that the conventional DQN experienced, as mentioned in [18]:

$$y_t^{DDQN} = r_t + \zeta \hat{Q}(s_{t+1}, \arg\max_{a \in A} Q(s_{t+1}, a, \theta_t), \theta_t^-) \quad (7)$$

Specifically, in Eq. (7), the action is chosen by a primary neural network Q, and the value associated with that action is determined by a secondary network (ˆ Q), which is referred to as the target. While ˆQ's architecture is identical to Q's, the primary network parameters (θ) are updated in its parameter vector (θ−) at each τ step. On the other hand, a mini-batch of tuples from a first-in, first-out (FIFO) data structure called the replay buffer are used to update the main network parameters, following a predefined probability distribution function. During the optimization process, these tuples are considered in the loss function (LDDQN(θ) = E[(yDDQN t − Q(st, at, θ))2]). The balancing ratio, as shown in Eq. (6), prevents an additive component associated with the majority class from being dominant when the partial derivatives for the θ update are computed.

IV. EXPERIMENTAL EVALUATION

A. Materials and methods

1) Database: The Mendeley dataset [48] is used, same as in [18]. Table I provides a concise overview of the dataset's primary structural features. Vectorized URLs make up this collection.

TABLE I: Mendeley structure summary.

No. malicious URLs	*No. legitimate URLs*	*No. features*
30647	58000	110[a]

[a] To which is added the label phishing.

comprised of a collection of numerical characteristics derived from the textual elements that make up the generic URL. Also, the generic sample's validity of its transport layer security (TLS) certificate and its indexing in some search engines are both determined by a set of Boolean criteria. The general rule of thumb is that 25% of the data set is for testing purposes, and 75% is for training purposes (S). We obtain ρ = 0.3 by randomly removing samples from the minority class in the training set (SP) in order to handle a moderate imbalanced scenario. Lastly, a min-max technique was used to normalize the data, which means that values between 0 and 1 were scaled, as follows: xnew = xold−xmin xmax−xmin.

(2) KPIs: We tested two metrics—training time and classification accuracy. The first step is to examine the relationship between the growth of S's feature count and the training time trend (in seconds). As a result, the agent undergoes 110 iterations of training and testing for every method, with the exception of the non-feature selection (WFS) scenario.

See Eq. (1)–(5) for details on how the feature selection strategies examined rank features, which in turn determine the incremental number of features.

True positives (TPs) or false positives (FPs) indicate a (n) incorrect categorization of samples linked to the positive class, in this case, harmful URLs, depending on the situation at hand. Similarly, "true negative" (TN) and "false negative" (FN) indicate accurate and wrong negative class classifications, respectively.

If we want to know how well each algorithm does at classification, we may look at the same measures that were used in [18]: accuracy, recall, geomertic mean, F1 score, area under the receiver operating characteristic, and index of balanced accuracy. Additionally, the measures themselves are generated from the confusion matrices CM = _ TN FP FN TP _ for each top performance, which are shown as follows: (i) The precision is equal to TP plus FP, (ii) the total probability of a success is equal to TP plus FN, (TNR is equal to TN plus FP), the G-Mean is equal to the square of TPR times TNR, (iv) the internal bias analysis is equal to (1 plus γ times (TPR minus TNR)) times G-Mean2, and (v) the first statistic, F1, is equal to 2 times TPR times PREC. P is set to 0.1 for the IBA calculation. Notably, when the following improvement requirements are fulfilled concurrently with regard to the WFS example, we evaluated the algorithm trained on the data constituted of the smallest amount of features picked by each approach as the best performer: (i) minimizing training time (in seconds) by 10%; (ii) increasing G-Mean by 2%. The top performers' training times and feature count as a function of each feature selection strategy are also highlighted.

3) Configuring algorithms: The DRL-based classifier in this study is based on the identical agent design

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 9, September 2025
PP 374–384

configuration as in [18]. To be more precise, Table II provides a summary. First, feature selection; second, DDQN-based classifier

TABLE II: DDQN-based classifier hyper-parameter setting.

Hyper-parameter	*Value*
No. hidden layers	2
Hidden layer size	256
Learning rate	2.5×10^{-4}
Activation function	Rectified Linear Unit
Target network update period	800
Optimizer	Adam
Replay buffer size	1.7×10^{5}
Mini-batch size	128
Sampling distribution	Uniform
Discount factor	10^{-1}
Exploration strategy	Polynomial ε decay
ε_{min}	0.5
No. episodes to decay from 1 to ε_{min}	10^{3}
No. training epochs	10^{6}

strategies2 were put into play by augmenting the Python code that was given by the two open-source sources mentioned in the footnotes. The following hardware configurations were used in the studies on an Ubuntu-OS machine: This system has a 2.40 GHz Intel Xeon(R) E5-2620 v3 processor and 16 GB of RAM.

B. Results

Training time trends with increasing feature count, as shown in Fig. 1, are ordered by the relative relevance scores awarded by the various selection strategies. The real patterns are shown via scatter plots. Each of them is roughly represented by a regression line with an increasing gradient. The effect of data complexity, such as the amount of features in the training dataset, on the training time of our DRL agent is therefore well understood. As a result, feature selection algorithms are a great tool for cutting down on computing burden. The findings are shown in Figure 2, which concludes the training time study. The data demonstrates that, with the exception of the GINI scenario, training time is directly proportional to feature count. Whatever the case may be, as stated in Section IV-A2, for a set of characteristics equal to: (i) 99 using CHI-SQUARE; (ii) 93 using T-SCORE; (iii) 92 using F-SCORE; (iv) 90 using PCC; and (v) 81 using GINI, the training time for each top performer is at least 100 seconds less than in the WFS example. The effect of feature selection is affected by the use of normalized data in our tests, which is worth considering even if the greatest decrease in features is rather minor (27% compared to the baseline dataset) [49].

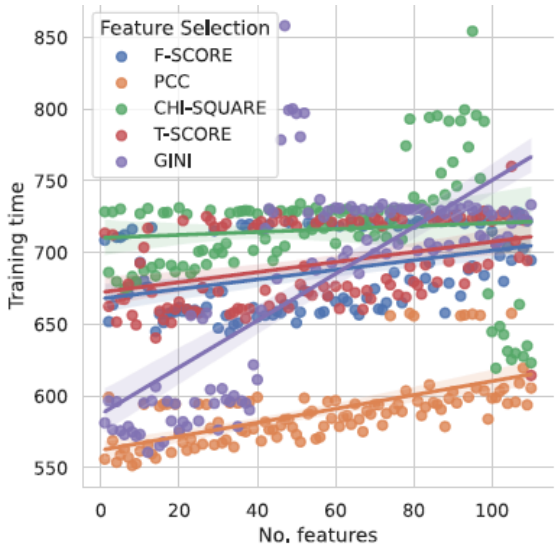

Fig. 1: Training time as a linear approximation function of the number of features ranked according to the feature selection strategy adopted.

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 9, September 2025
PP 374–384

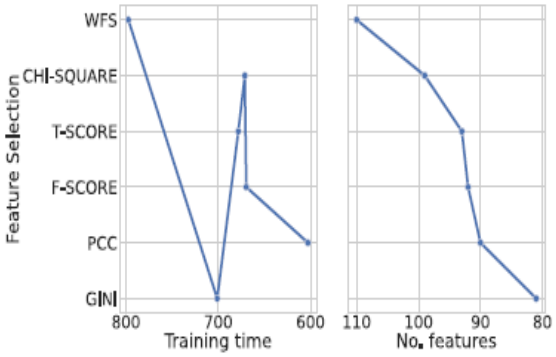

Fig. 2: Training time required and number of features in training data used by the top performers (arranged by amount of features) versus WFS case.

Assuming the evaluated techniques can reduce training time, our next step will be to analyze their effects on classification performance. Figure 3 displays confusion matrices that reflect the classification performance of the top performers on the full (WFS) dataset and the DRL-based classifier on the reduced dataset, respectively. True or predicted values are shown in the generic matrix's columns (rows). With the set of values (TN, FP, FN, TP) presented in Fig. 3, the scores for the classification metrics in Fig. 4 may be obtained for each case:

Using all features in training (WFS) leads to a high number of false positives (FPs) and an accuracy score of 84%. Conversely, getting a low (high) value of TPs (FNs) results in a recall score of around 96%. With an F1 score of 89.6%, we can see that the preceding metrics are harmonically related. Even while the classifier doesn't seem to be particularly good at recognizing the minority class, its high recall value has a beneficial impact on both the geometric mean (about 93%) and the IBA (0.87), indicating that it can handle the unbalanced situation.
At last, we have an AUC covered of 0.931.
• When compared to the WFS example, all metrics (except recall) improve with the adoption of feature selection methods (near to 96% independent of the feature selection approach). As seen in Figure 3, the most striking improvement is the decrease in FPs and, as a result, the increase in TNs. The improvement in accuracy, which falls within the range of 89% to 92%, is evidence of this. Consequently, the F1 score always falls within the range of 93 to 93.6%. The geometric mean also outperforms the WFS instance by 2-2.3% using the criteria that were used to choose the best performers. To no one's surprise, the AUC measure shows the same upward tendency as the geometric mean.
Finally, great progress is noted for the IBA score, which reaches a maximum value of 0.91 in the T-SCORE and GINI cases.

Among the contrasted strategies, we choose the one that employed less characteristics in training, while not precisely matching the shorter training duration, as indicated in Fig. 2; still, this decision is chosen according to the analysis originating from Fig. 1. In addition, observe that the line-approximating training time trend in the Gini example has the greatest angular coefficient compared with the other instances. By reducing the dataset from 110 to 81 columns, the feature selection technique significantly improves computational and classification performance. This is achieved by using the Gini index calculation. The agent's performance improves as the number of features decreases, in line with the work at hand (this indicates that it is learning to complete the task), which is in line with the curse of dimensionality problem that is common in classification tasks. This is a significant discovery since it goes against the grain of other DRL applications, which use complex models to increase performance on a given job by broadening the observation area [50].

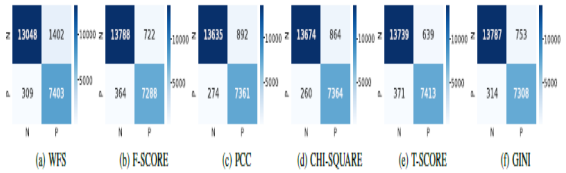

Fig. 3: C_M showing the overview of classification performance achieved by the DRL agent trained on the: (a) full dataset; (b)-(f) dataset reduced as a consequence of the feature selection strategy.

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 9, September 2025
PP 374–384

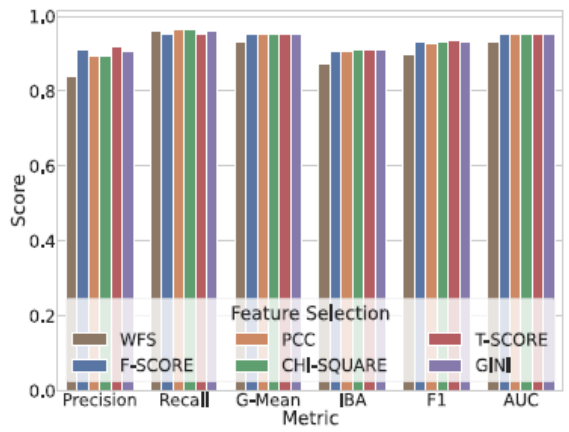

Fig. 4: Classification metric scores derived from Fig. 3.

V. CONCLUSION

The need for intrusion detection models that can fulfill the requirement for resilience with regard to the conceptual drift characterized by the data has risen due to the expansion of online phishing, which includes malicious web apps. Furthermore, the uneven distribution of distinct class samples prevented the same data from undergoing manipulations to eliminate inherent bias, thereby maintaining the portrayal of reality. On top of that, there is the potential drawback of training time overhead for complex ML models like DRL-based classifiers, which meet the aforementioned criteria. This work set out to solve this problem by studying how different feature selection techniques affect training duration and classification accuracy. We focused on lightweight statistical and correlation-based methods. Improvements in training time and classification performance were seen in the experimental assessment, highlighting the efficiency of lowering the observation space size, or the number of columns in the training set. This is where the feature selection method that derived its findings from the Gini index outperformed its rivals. Since there is often an imbalance in the availability of samples from a given family, future work may explore applying such a solution to other imbalanced classification challenges in the cybersecurity area, including multiclass malware classification.

REFERENCES

[1] M. Vijayalakshmi, S. Mercy Shalinie, M. H. Yang, and R. M. U, “Web phishing detection techniques: a survey on the state-of-the-art, taxonomy and future directions,” Iet Networks, vol. 9, no. 5, pp. 235–246, 2020.

[2] D. Rendell, “Understanding the evolution of malware,” Computer Fraud & Security, vol. 2019, no. 1, pp. 17–19, 2019.

[3] A. Coscia, V. Dentamaro, S. Galantucci, A. Maci, and G. Pirlo, “Yamme: a yara-byte-signatures metamorphic mutation engine,” IEEE Transactions on Information Forensics and Security, vol. 18, pp. 4530–4545, 2023.

[4] P. Kishore, S. K. Barisal, D. P. Mohapatra, and R. Mall, “An efficient two-stage pipeline model with filtering algorithm for mislabeled malware detection,” Computers & Security, vol. 135, p. 103499, 2023.

[5] H. Alamro, W. Mtouaa, S. Aljameel, A. S. Salama, M. A. Hamza, and A. Y. Othman, “Automated android malware detection using optimal ensemble learning approach for cybersecurity,” IEEE Access, vol. 11, pp. 72 509–72 517, 2023.

[6] B. N. Chaithanya and S. H. Brahmananda, “Detecting ransomware attacks distribution through phishing urls using machine learning,” in Computer Networks and Inventive Communication Technologies, 2022, pp. 821–832.

[7] 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. 18 499–18 519, 2023.

[8] L. Tang and Q. H. Mahmoud, “A survey of machine learning-based solutions for phishing website detection,” Machine Learning and Knowledge Extraction, vol. 3, no. 3, pp. 672–694, 2021.

[9] E. Benavides, W. Fuertes, S. Sanchez, and M. Sanchez, “Classification of phishing attack solutions by employing deep learning techniques: A systematic

literature review," in Developments and Advances in Defense and Security, 2020, pp. 51–64.

[10] N. Q. Do, A. Selamat, O. Krejcar, E. Herrera-Viedma, and H. Fujita, "Deep learning for phishing detection: Taxonomy, current challenges and future directions," IEEE Access, vol. 10, pp. 36 429–36 463, 2022.

[11] T. T. Nguyen and V. J. Reddi, "Deep reinforcement learning for cyber security," IEEE Transactions on Neural Networks and Learning Systems, vol. 34, no. 8, pp. 3779–3795, 2023.

[12] Z. Utic and K. Ramachandran, "A survey of reinforcement learning in intrusion detection," in 2022 1st International Conference on AI in Cybersecurity (ICAIC), 2022, pp. 1–8.

[13] M. Lopez-Martin, B. Carro, and A. Sanchez-Esguevillas, "Application of deep reinforcement learning to intrusion detection for supervised problems," Expert Systems with Applications, vol. 141, p. 112963, 2020.

[14] M. Chatterjee and A.-S. Namin, "Detecting phishing websites through deep reinforcement learning," in 2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC), vol. 2, 2019, pp. 227–232.

[15] A. Azari, J. M. Namayanja, N. Kaur, V. Misal, and S. Shukla, "Imbalanced learning in massive phishing datasets," in 2020 IEEE 6th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing, (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS), 2020, pp. 127–132.

[16] A. El Aassal, S. Baki, A. Das, and R. M. Verma, "An in-depth benchmarking and evaluation of phishing detection research for security needs," IEEE Access, vol. 8, pp. 22 170–22 192, 2020.

[17] I. Ul Hassan, R. H. Ali, Z. Ul Abideen, T. A. Khan, and R. Kouatly, "Significance of machine learning for detection of malicious websites on an unbalanced dataset," Digital, vol. 2, no. 4, pp. 501–519, 2022.

[18] A. Maci, A. Santorsola, A. Coscia, and A. Iannacone, "Unbalanced web phishing classification through deep reinforcement learning," Computers, vol. 12, no. 6, 2023.

[19] J. Cai, J. Luo, S. Wang, and S. Yang, "Feature selection in machine learning: A new perspective," Neurocomputing, vol. 300, pp. 70–79, 2018.

[20] J. M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," Journal of Big Data, vol. 6, no. 1, pp. 1–54, 2019.

[21] S. Mohammadi, H. Mirvaziri, M. Ghazizadeh-Ahsaee, and H. Karimipour, "Cyber intrusion detection by combined feature selection algorithm," Journal of Information Security and Applications, vol. 44, pp. 80–88, 2019.

[22] K. A. Taher, B. Mohammed Yasin Jisan, and M. M. Rahman, "Network intrusion detection using supervised machine learning technique with feature selection," in 2019 International Conference on Robotics,Electrical and Signal Processing Techniques (ICREST), 2019, pp. 643–646.

[23] S. B and M. K, "Firefly algorithm based feature selection for network intrusion detection," Computers & Security, vol. 81, pp. 148–155, 2019.

[24] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu, "Feature selection: A data perspective," ACM Computing Surveys, vol. 50, no. 6, 2017.

[25] S. R. Sharma, R. Parthasarathy, and P. B. Honnavalli, "A feature selection comparative study for web phishing datasets," in 2020 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT), 2020, pp. 1–6.

[26] K. L. Chiew, C. L. Tan, K. Wong, K. S. Yong, and W. K. Tiong, "A new hybrid ensemble feature selection framework for machine learning based phishing detection system," Information Sciences, vol. 484, pp. 153–166, 2019.

[27] M. Zabihimayvan and D. Doran, "Fuzzy rough set feature selection to enhance phishing attack

detection," in 2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), 2019, pp. 1–6.

[28] S. Zaman, S. M. Uddin Deep, Z. Kawsar, M. Ashaduzzaman, and A. I. Pritom, "Phishing website detection using effective classifiers and feature selection techniques," in 2019 2nd International Conference on Innovation in Engineering and Technology (ICIET), 2019, pp. 1–6.

[29] H. M. Junaid Khan, Q. Niyaz, V. K. Devabhaktuni, S. Guo, and U. Shaikh, "Identifying generic features for malicious url detection system," in 2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), 2019, pp. 0347–0352.

[30] Y.-C. Chen, Y.-W. Ma, and J.-L. Chen, "Intelligent malicious url detection with feature analysis," in 2020 IEEE Symposium on Computers and Communications (ISCC), 2020, pp. 1–5.

[31] R. Talaei Pashiri, Y. Rostami, and M. Mahrami, "Spam detection through feature selection using artificial neural network and sine–cosine algorithm," Mathematical Sciences, vol. 14, pp. 193–199, 2020.

[32] T. Li, G. Kou, and Y. Peng, "Improving malicious urls detection via feature engineering: Linear and nonlinear space transformation methods," Information Systems, vol. 91, p. 101494, 2020.

[33] S.-J. Bu and H.-J. Kim, "Optimized url feature selection based on genetic-algorithm-embedded deep learning for phishing website detection," Electronics, vol. 11, no. 7, 2022.

[34] S. Ghareeb, M. Mahyoub, and J. Mustafina, "Analysis of feature selection and phishing website classification using machine learning," in 2023 15th International Conference on Developments in eSystems Engineering (DeSE), 2023, pp. 178–183.

[35] S. A. A. Ghaleb, M. Mohamad, W. A. H. M. Ghanem, A. B. Nasser, M. Ghetas, A. M. Abdullahi, S. A. M. Saleh, H. Arshad, A. E. Omolara, and O. I. Abiodun, "Feature selection by multiobjective optimization: Application to spam detection system by neural networks and grasshopper optimization algorithm," IEEE Access, vol. 10, pp. 98 475–98 489, 2022.

[36] S. A. A. Ghaleb, M. Mohamad, S. A. Fadzli, and W. A. H. M. Ghanem, "Training neural networks by enhance grasshopper optimization algorithm for spam detection system," IEEE Access, vol. 9, pp. 116 768–116 813, 2021.

[37] A. Karim, M. Shahroz, K. Mustofa, S. B. Belhaouari, and S. R. K. Joga, "Phishing detection system through hybrid machine learning based on url," IEEE Access, vol. 11, pp. 36 805–36 822, 2023.

[38] M. Akhavan and S. M. Hossein Hasheminejad, "An unsupervised feature selection for web phishing data using an evolutionary approach," in 2021 7th International Conference on Web Research (ICWR), 2021, pp. 41– 47.

[39] J. Wang, "An improved genetic algorithm for web phishing detection feature selection," in 2022 Asia Conference on Algorithms, Computing and Machine Learning (CACML), 2022, pp. 130–134.

[40] N. Sevani, I. Hermawan, and W. Jatmiko, "Feature selection based on fscore for enhancing ctg data classification," in 2019 IEEE International Conference on Cybernetics and Computational Intelligence (CyberneticsCom), 2019, pp. 18–22.

[41] G. Aikins, S. Jagtap, and W. Gao, "Resilience analysis of deep qlearning algorithms in driving simulations against cyberattacks," in 2022 1st International Conference on AI in Cybersecurity (ICAIC), 2022, pp. 1–6.

[42] M. Sewak, S. K. Sahay, and H. Rathore, "Deep reinforcement learning for cybersecurity threat detection and protection: A review," in Secure Knowledge Management In The Artificial Intelligence Era, 2022, pp. 51–72.

[43] A. Santorsola, A. Maci, P. Delvecchio, and A. Coscia, "A reinforcementlearning- based agent to discover safety-critical states in smart grid environments," in 2023 3rd International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME), 2023, pp. 1–7.

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 9, September 2025
PP 374–384

[44] W. Brescia, A. Maci, S. Mascolo, and L. De Cicco, “Safe reinforcement learning for autonomous navigation of a driveable vertical mast lift,” IFAC-PapersOnLine, vol. 56, no. 2, pp. 9068–9073, 2023, 22nd IFAC World Congress.

[45] K. Ren, Y. Zeng, Z. Cao, and Y. Zhang, “Id-rdrl: a deep reinforcement learning-based feature selection intrusion detection model,” Scientific Reports, vol. 12, no. 1, p. 15370, 2022.

[46] X. Wang, S. Wang, X. Liang, D. Zhao, J. Huang, X. Xu, B. Dai, and Q. Miao, “Deep reinforcement learning: A survey,” IEEE Transactions on Neural Networks and Learning Systems, pp. 1–15, 2022.

[47] E. Lin, Q. Chen, and X. Qi, “Deep reinforcement learning for imbalanced classification,” Applied Intelligence, vol. 50, pp. 2488–2502, 2020.

[48] G. Vrbanˇciˇc, I. Fister Jr, and V. Podgorelec, “Datasets for phishing websites detection,” Data in Brief, vol. 33, p. 106438, 2020.

[49] D. Singh and B. Singh, “Investigating the impact of data normalization on classification performance,” Applied Soft Computing, vol. 97, p. 105524, 2020.

[50] K. Ota, T. Oiki, D. Jha, T. Mariyama, and D. Nikovski, “Can increasing input dimensionality improve deep reinforcement learning?” in Proceedings of the 37th International Conference on Machine Learning, vol. 119, 2020, pp. 7424–7433.