AutoPhish: A Grammar-based AutoML Approach to Learn Classifiers for Phishing Detection

João Guilherme Miranda[1], Mateus L.S.D. Barros,[1], Tapas Si[2], Carlo Marcelo R. Silva[3] and Péricles B.C. Miranda[1]

[1]Departamento de Computação – UFRPE, Brazil

[2]Bankura Unnayani Institute of Engineering, India

[3]Universidade de Pernambuco, Brazil

{pericles.miranda}@ufrpe.br

Abstract. *The increasing sophistication of cyber threats—particularly phishing—demands advanced and adaptive detection mechanisms to protect users and organizations. Traditional defenses struggle to keep pace as phishing techniques such as Clone Phishing, Spear Phishing, DNS-Based Phishing, and Man-In-The-Middle attacks evolve. Recent research has extensively applied machine learning (ML) models to phishing detection, emphasizing the importance of attribute selection and classifier optimization. While approaches using rule-based systems, ensemble models, and artificial neural networks (ANNs) have shown promising results, the reliance on static datasets and generalized models limits their effectiveness in dynamic, real-world scenarios. This article proposes AutoPhish, a novel phishing detection approach based on Grammatical Evolution (GE). GE is a grammar-driven genetic programming method capable of generating customized and optimized classifiers. AutoPhish implements a single objective GE aiming to produce classifiers which maximize $F_1-score$. Our method is evaluated in realistic settings with imbalanced datasets and compared to traditional machine learning algorithms using performance metrics such as accuracy, recall, precision, and $F_1-score$. The results show that classifiers evolved using AutoPhish consistently outperform most baseline methods, demonstrating strong potential for practical deployment. This study underscores the value of evolutionary computation in cybersecurity and advances the development of adaptive, high-performance phishing detection systems.*

Resumo. *A crescente sofisticação das ameaças cibernéticas — em especial o phishing — exige mecanismos de detecção avançados e adaptativos para proteger usuários e organizações. As defesas tradicionais têm dificuldade em acompanhar a evolução de técnicas como Clone Phishing, Spear Phishing, DNS-Based Phishing e ataques do tipo Man-In-The-Middle. Pesquisas recentes têm aplicado extensivamente modelos de aprendizado de máquina (ML) à detecção de phishing, destacando a importância da seleção de atributos e da otimização dos classificadores. Embora abordagens baseadas em sistemas de regras, modelos de comitê (ensemble) e redes neurais artificiais (ANNs) tenham apresentado resultados promissores, a dependência de conjuntos de dados estáticos e modelos generalistas limita sua eficácia em cenários dinâmicos e reais. Este*

artigo propõe o AutoPhish, uma nova abordagem para detecção de phishing baseada em Evolução Gramatical (GE). A GE é um método de programação genética orientado por gramática, capaz de gerar classificadores personalizados e otimizados. O AutoPhish implementa uma GE com objetivo único, visando gerar classificadores que maximizem o $F_1-score$. Nossa abordagem é avaliada em cenários realistas com conjuntos de dados desbalanceados e comparada com algoritmos tradicionais de aprendizado de máquina, utilizando métricas de desempenho como acurácia, revocação, precisão e $F_1-score$. Os resultados mostram que os classificadores evoluídos com o AutoPhish superam consistentemente a maioria dos métodos de base, demonstrando alto potencial para aplicação prática. Este estudo destaca o valor da computação evolutiva na cibersegurança e impulsiona o desenvolvimento de sistemas de detecção de phishing adaptativos e de alto desempenho.

1. Introduction

The rapid evolution of cyber threats, particularly phishing, necessitates sophisticated solutions to safeguard users and organizations. Phishing remains one of the most prevalent and devastating techniques employed by cybercriminals, aiming to steal sensitive information and compromise digital security. Methods such as Clone Phishing, Spear Phishing, DNS-Based Phishing, and Man-In-The-Middle attacks have become increasingly sophisticated, challenging traditional defenses and demanding innovative approaches for detection and mitigation.

Phishing detection has been extensively explored through machine learning techniques aimed at distinguishing malicious websites from legitimate ones. Studies have emphasized the importance of selecting relevant attributes—such as the presence of IP addresses and symbols in URLs—for effective classification model training. Various approaches, including rule-based systems, ensemble models, ANNs, and feature selection techniques, have been applied to improve detection accuracy and robustness. Deep neural network models [Hasan et al. 2019] and comparative analyses of classifiers like Naive Bayes and Random Forest [Yaswanth and Nagaraju 2023] have shown high prediction performance. Further research has examined the role of feature selection and classifier performance [Fadheel et al. 2017, de Barros et al. 2019, Barros et al. 2019], underscoring its impact on model optimization. Additionally, the expert systems [de Barros et al. 2020], based on attribute-driven rules, demonstrated superior results compared to traditional machine learning algorithms when evaluated on datasets from Phishtank and OpenPhish, highlighting its potential as an effective tool for phishing detection.

Despite advances in phishing detection using machine learning and expert systems, developing personalized and adaptive models tailored to specific contexts remains a key challenge [Yaswanth and Nagaraju 2023]. Current approaches often rely on generic datasets and static features, limiting their effectiveness against evolving threats. This underscores the need for optimized models that offer precision and adaptability across diverse scenarios.

This article introduces *AutoPhish*, an innovative phishing detection approach based on Grammatical Evolution (GE) [O'Neill and Ryan 2004]. GE is a grammar-driven

genetic programming method automatically generating tailored and optimized classifiers. AutoPhish employs a single-objective GE strategy to produce classifiers that maximize the F_1-score. The proposed method is evaluated against conventional classification algorithms in realistic scenarios involving imbalanced datasets. Through the analysis of standard performance metrics—accuracy, recall, precision, and F_1-score—the study shows that classifiers generated by AutoPhish are highly competitive, consistently outperforming most baseline methods. These findings underscore the effectiveness of GE in phishing detection and its promising role in enhancing cybersecurity solutions.

2. Background

2.1. The Problem of Phishing Detection

The approach to phishing detection has evolved considerably in response to the continuous advancement of techniques employed by cybercriminals. Phishing, which includes Clone Phishing, Spear Phishing, DNS-Based Phishing, and Man-In-The-Middle attacks, presents persistent challenges to conventional defense mechanisms. Early mitigation strategies relied on browser toolbars and email filtering using rule-based systems and URL analysis; however, these methods were limited by high false negative rates and an inability to adapt to the growing complexity of phishing tactics [Fette et al. 2007]. As phishing attacks have become more sophisticated, deploying intelligent algorithms—particularly those grounded in machine learning—has gained prominence in cybersecurity. Research has demonstrated the effectiveness of these approaches in detecting distinctive patterns and features associated with malicious websites [Mahajan and Siddavatam 2018]. Techniques such as optimal feature selection and artificial neural networks have further improved detection accuracy and robustness [Habib et al. 2022]. Recent advances have expanded phishing detection capabilities by integrating deep learning models and optimization techniques such as particle swarm optimization, enhancing the identification of phishing websites [Lingam et al. 2021]. Email-based phishing detection has similarly benefited from applying natural language processing and machine learning, which facilitate the contextual analysis necessary for accurate threat classification [Bountakas et al. 2021, Peng et al. 2018]. Given the rapid evolution of phishing strategies, there is a growing emphasis on adaptive and continuously learning detection systems that can evolve with emerging threats [Yadollahi et al. 2019]. As a result, effective phishing detection now requires a comprehensive, multifaceted approach that combines traditional methods with cutting-edge developments in artificial intelligence and machine learning to counter increasingly sophisticated cyberattacks.

2.2. Grammatical Evolution

Identifying an appropriate hybrid system for a specific forecasting task remains a complex and open challenge. While Grammatical Evolution (GE) has not yet been widely explored for hybrid system generation, it has demonstrated strong potential for automatically creating models in other domains [da Silva et al. 2021, Si et al. 2021, Miranda and Prudêncio 2020, Diniz et al. 2018, Miranda et al. 2017, da Silva et al. 2023b, da Silva et al. 2023a, Basgalupp et al. 2025]. GE is a type of Genetic Programming (GP) algorithm that employs a context-free grammar to evolve variable-length computer programs [O'Neill and Ryan 2004]. It consists of three main components: a search engine, a grammar, and a mapping process.

The search engine—typically a Genetic Algorithm (GA) — iteratively evolves candidate solutions represented as genotypes (binary arrays). Each genotype consists of codons (8-bit values) arranged in a variable-length binary string.

During evolution, the fitness of each genotype is evaluated through a mapping process that translates the genotype into a corresponding phenotype (a program, expression, or classifier). This mapping is guided by a grammar defined in Backus-Naur Form (BNF), which is domain-specific and outlines the valid syntactic structures for the evolved programs [O'Neill and Ryan 2004]. The fitness function then assesses the quality of the phenotype, determining how well the solution performs for the given task. In GE, a context-free grammar is formally represented as a tuple N, T, P, S, where N denotes the set of non-terminal symbols, T the set of terminal symbols, P the set of production rules, and S the start symbol. Each production rule is $x \models y$, where $x \in N$ and y combine terminals and non-terminals ($y \in T \cup N$). Multiple production alternatives for a non-terminal are separated using the disjunction symbol |. Figure 1 presents an example of a simple context-free grammar capable of generating arithmetic expressions. In this example, `<exp>`, `<var>`, and `<op>` are non-terminals, while symbols such as `x`, `+`, `−`, `/`, and `*` are terminals.

$$
\begin{aligned}
&N = \{\langle \text{exp} \rangle, \langle \text{var} \rangle, \langle \text{op} \rangle\} \\
&T = \{x, , -, , *\} \\
&S = \langle \text{exp} \rangle \\
&P = \{\langle \text{exp} \rangle \models \langle \text{var} \rangle \mid \langle \text{exp} \rangle \langle \text{or} \rangle \langle \text{exp} \rangle \\
&\qquad \langle \text{op} \rangle \models + \mid - \mid / \mid * \\
&\qquad \langle \text{var} \rangle \models \text{x}\}
\end{aligned}
$$

Figura 1. An example of context-free grammar.

The genotype-to-phenotype mapping process uses the codon values in the genotype to select production rules during derivation. For each non-terminal, the value of the current codon is taken modulo the number of applicable production rules, effectively selecting one rule based on:

rule = codon's value
MOD
(number of rules of the current non terminal item).

For example, consider the genotype `(18, 102, 203, 57, 62)` and assume that the initial non-terminal is `<exp>` with two production alternatives. Since 18 mod 2 = 0, the first rule is selected. The mapping continues left to right, resolving each non-terminal until a complete expression (phenotype) is formed. If the end of the genotype is reached before all non-terminals are resolved, the process wraps around to the beginning of the genotype. The final output is the phenotype, such as the expression `x * (x + x)`, which is then evaluated by the fitness function. It is important to note that while crossover and mutation operations are applied to the genotype, fitness evaluation occurs at the phenotype level. This separation between search and evaluation allows GE to explore the solution space flexibly while maintaining syntactic correctness through grammar-based constraints.

3. Related Work

Phishing detection has been a widely studied topic, with a focus on applying machine learning techniques to identify malicious websites. To effectively characterize phishing websites and develop databases for training classification models, it is essential to consider specific attributes such as the presence of IP addresses in domain names and the use of symbols in URLs. Numerous studies have focused on leveraging machine learning techniques to distinguish phishing websites from legitimate ones and enhance the detection of these threats [Zhu et al. 2018, Zamir et al. 2020]. These studies have explored various approaches, including rule-based classification, ensemble machine learning models, feature selection, and neural networks, to improve the accuracy and stability of phishing website detection.

For instance, the study by [Hasan et al. 2019] developed models based on Deep Artificial Neural Networks for predicting phishing attacks, achieving promising results in terms of accuracy.

Additionally, [Yaswanth and Nagaraju 2023] compared the performance of Naive Bayes (NB) and Random Forest (RF) classifiers in detecting phishing websites. Using a public Kaggle dataset and employing 10-fold cross-validation, the study evaluated the accuracy of both classifiers. The results indicated that Naive Bayes achieved an accuracy of 95.58%, while Random Forest achieved 94.675%. Although both classifiers demonstrated high efficiency, Naive Bayes outperformed Random Forest slightly and statistically significantly in phishing prediction.

Other studies, such as [Fadheel et al. 2017, de Barros et al. 2019, Barros et al. 2019], focused on feature selection and classifier comparison. The authors evaluated the performance of classifiers such as Logistic Regression and Support Vector Machines, and highlighted the critical role of appropriate feature selection in optimizing model performance.

Barros et al. [de Barros et al. 2020] proposed an expert system for the detection of malicious web pages, named Xphide. Xphide is a rule-based system, with its rules derived from an in-depth analysis of attributes proposed by [Barros et al. 2019]. A comparative analysis of the selected attributes was conducted, allowing for the assignment of weights based on their relevance. Xphide was evaluated against traditional machine learning algorithms in terms of accuracy, precision, and recall. The approaches were assessed using datasets from Phishtank and OpenPhish. The results demonstrated that Xphide outperformed the classical algorithms, presenting itself as a promising alternative for the automated classification of malicious web pages.

Although significant progress has been made in phishing detection through various ML techniques and expert systems, the challenge of developing personalized and optimized models tailored to specific contexts remains an open avenue for research. Despite the promising results of models like Xphide and the high accuracy rates reported by classifiers like Naive Bayes and Random Forest, these approaches often rely on generic datasets and fixed feature sets. As phishing tactics evolve and become more sophisticated, there is a growing need for adaptive models that can be fine-tuned to reflect the dynamic nature of phishing threats across different user profiles, platforms, and environments. This justifies the importance of the current research into customizable detection algorithms.

4. Proposed Approach

To address the limitations of traditional and static phishing detection models, we propose *AutoPhish*, a novel approach based on GE for the automatic generation of optimized and customized classifiers. Unlike conventional machine learning models that rely on fixed feature representations and predefined classifier structures, AutoPhish evolves classifiers specifically tailored to phishing data's underlying patterns. The core idea of AutoPhish is to leverage the flexibility of GE to discover classification rules that maximize the F_1-score, which is particularly relevant in the context of imbalanced datasets—a common characteristic in phishing detection scenarios. By encoding candidate classifiers as derivations of context-free grammar and applying evolutionary operators such as mutation and crossover, AutoPhish iteratively refines solutions toward optimal performance. The result is a set of lightweight, interpretable, and high-performing classifiers capable of distinguishing phishing attempts from legitimate content.

AutoPhish's search and mapping procedures follow the standard GE methodology described in Section 2.2 and will not be repeated here. The following section introduces the novel grammar developed for this work.

The grammar proposed in this study, illustrated in Figure 2, was designed to guide the generation of mathematical expressions that serve as the structural basis for the classification models evolved by AutoPhish. Expressed in BNF, the grammar defines a flexible and expressive syntax capable of representing a wide variety of functional forms. These forms combine arithmetic operations and mathematical functions that operate on the features of the phishing dataset, enabling the synthesis of sophisticated and highly customized classifiers.

The non-terminal symbol `<expr>` serves as the root of the derivation and defines the structure of the mathematical expression to be evolved. It supports the recursive composition of subexpressions using a range of operators and functions, allowing the resulting classifiers to capture complex relationships and non-linear patterns in the data. Including mathematical functions such as `np.abs`, `np.sin`, `np.tanh`, and `np.exp` enhances the grammar's ability to model nuanced behaviors commonly present in phishing datasets, which may not be linearly separable.

The `<var>` non-terminal represents the input variables, which correspond to the features (columns) of the dataset. By referencing each feature as `x[:, i]`, the grammar ensures direct access to all input dimensions, enabling the generation of expressions that leverage specific attribute combinations relevant to phishing detection. Furthermore, the use of protected operators—such as `pdiv` and `plog`—ensures numerical stability during model execution by handling undefined operations (e.g., division by zero or logarithm of non-positive values), which are common pitfalls in symbolic representations.

Overall, this grammar provides a robust and adaptable foundation for the automatic synthesis of classification models. Its design balances expressiveness with control, allows replicability, automatically selects relevant features and promotes the evolution of diverse, problem-specific solutions capable of distinguishing phishing from legitimate instances in varied and dynamic scenarios.

```
<expr> ::= <expr> + <expr>
         | <expr> - <expr>
         | <expr> * <expr>
         | pdiv(<expr>, <expr>)
         | psqrt(<expr>)
         | np.abs(<expr>)
         | np.sin(<expr>)
         | np.tanh(<expr>)
         | np.exp(<expr>)
         | plog(<expr>)
         | <var>

<var> ::= x[:, 0]
        | x[:, 1]
        | x[:, 2]
        | x[:, 3]
        | ...
        | x[:, 15]
        | x[:, 16]
        | x[:, 17]
```

Figura 2. CFG in BNF applied in the study.

5. Experimental Methodology

5.1. Dataset

In this study, we adopted the data collection and preprocessing methodology outlined in [de Barros et al. 2020], which emphasizes the construction of a reliable dataset for phishing detection tasks. We selected three distinct datasets from well-established phishing detection repositories to ensure a diverse and representative sample of real-world cases.

The first dataset was obtained from the Phishtank repository[1], a collaborative platform that collects and verifies phishing URLs reported by users. After filtering and cleaning, this dataset provided 4,040 verified phishing instances. The second dataset was sourced from OpenPhish[2], a widely used automated phishing intelligence platform, contributing an additional 752 confirmed phishing URLs. These two datasets served as the foundation for capturing many phishing attack patterns. To enrich the dataset with legitimate examples and enable the training of effective classification models, we also incorporated a third dataset comprising 661 benign URLs. These were originally flagged as suspicious by Phishtank but, upon further analysis, were confirmed to be non-malicious. Including this subset of legitimate URLs introduces an essential element of class diversity and addresses the class imbalance problem commonly observed in phishing datasets. Prior to merging the datasets, we performed a comprehensive data-cleaning process, which involved the removal of duplicate domains, inconsistent entries, and potentially

[1]https://www.phishtank.com/

[2]https://openphish.com/

mislabeled data. This step was crucial to reduce noise, avoid bias in model training, and minimize false positives during evaluation.

The three curated datasets were then consolidated into a single unified dataset, resulting in a final dataset that includes both phishing (class 0) and benign (class 1) instances. Figure 3 presents the class distribution of the resulting dataset. This integrated dataset was used throughout our experiments to evaluate the performance and generalization capabilities of the proposed approach. The careful selection and preparation of this dataset were essential to ensure the reliability and validity of the experimental results.

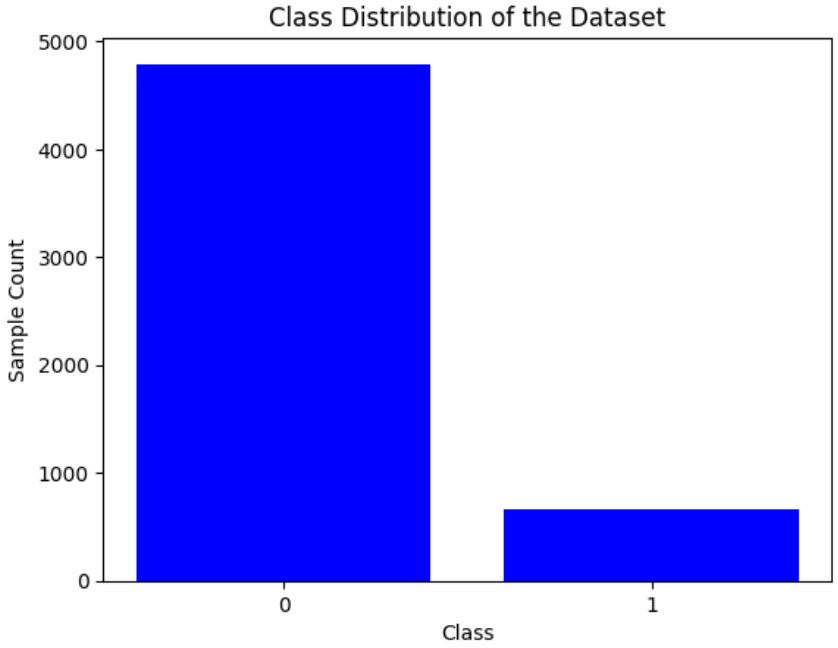

Figura 3. Class distribution of the integrated dataset.

5.2. Competing Methods

To evaluate the effectiveness of the proposed AutoPhish approach, we conducted a comprehensive comparative analysis against ten well-established classification algorithms. These included linear and non-linear models and ensemble-based methods, ensuring a robust and diverse benchmark. The list of comparison algorithms is as follows: Light Gradient Boosting Machine (LightGBM), Extreme Gradient Boosting (XGBoost), Decision Tree Classifier (DT), Extra Trees Classifier (ET), Gradient Boosting Classifier (GBC), Random Forest Classifier (RF), Support Vector Machine (SVM) with a Linear Kernel, Logistic Regression (LR), K-Nearest Neighbors Classifier (KNN), and Linear Discriminant Analysis (LDA).

All models, including AutoPhish, were evaluated using a 10-fold cross-validation procedure to ensure statistical reliability and mitigate overfitting. Moreover, the same random seed was used across all experiments to guarantee reproducibility and fairness in performance comparisons. It is important to mention that the dataset was kept imbalanced to simulate more realistic conditions. The objective of AutoPhish was to evolve a classifier that maximizes the F_1-score, a balanced metric that considers both precision and recall

- critical in phishing detection due to class imbalance. Among the evolved classifiers, the best-performing model produced by AutoPhish achieved the highest F_1-score on the dataset and is represented by the following mathematical expression:

```
plog(np.sin(psqrt(pdiv(x[:, 16] * np.abs(x[:, 3]), x[:, 15]) - x[:, 14]))) -
np.sin(np.abs(plog(np.sin(psqrt(pdiv(x[:, 17] * np.abs(x[:, 2]), x[:, 16] *
np.abs(x[:, 3])) + x[:, 2])))) + x[:, 17] + x[:, 0])) - psqrt(np.abs(x[:, 0]
+ x[:, 7] * plog(plog(x[:, 8] * x[:, 6]) + x[:, 2]) - x[:, 14]) - x[:, 8]) -
psqrt(x[:, 17])
```

This evolved expression illustrates the strength of *AutoPhish* in automatically uncovering complex, non-linear relationships among input features - patterns that would be challenging to engineer manually or detect using traditional classification models. The generated classifier incorporates a diverse set of input variables (e.g., `x[:, 0]`, `x[:, 2]`, `x[:, 14]`, etc.) and applies a wide range of mathematical transformations, including protected division (`pdiv`), protected logarithm (`plog`), square root, trigonometric, and exponential functions. Notably, the AutoPhish process also performs implicit feature selection: only variables contributing to improving the model's performance are retained, while irrelevant or redundant features are excluded from the final expression. This results in models that are accurate and expressive, more interpretable and computationally efficient, highlighting *AutoPhish* 's capability to evolve tailored and compact classifiers for phishing detection.

5.3. Evaluation Metrics

To comprehensively evaluate the performance of the classification models, we employed a set of well-established evaluation metrics commonly used in the machine learning literature [Tharwat 2021]. These metrics were selected due to their ability to provide a multifaceted assessment of the models' effectiveness, particularly in binary classification tasks such as phishing detection, where class imbalance is a common issue. The chosen metrics are accuracy, recall, precision, and F_1-score, each offering unique insights into different aspects of predictive performance.

Accuracy quantifies the overall correctness of the model by measuring the proportion of correctly classified instances (both true positives and true negatives) over the total number of predictions. While useful for balanced datasets, it may be misleading when the data is imbalanced, as it does not differentiate between types of classification errors:

$$Accuracy = \frac{TP\ TN}{TP\ TN\ FP\ FN}$$

Recall, also known as sensitivity or true positive rate, evaluates the model's ability to identify all actual positive instances correctly. This metric is particularly important in phishing detection, where failing to identify a phishing instance (false negative) can have severe consequences:

$$Recall = \frac{TP}{TP\ FN}$$

Precision assesses the proportion of true positive predictions among all instances predicted as positive. It is a critical metric in scenarios where false positives are costly, helping to ensure that identified phishing instances are indeed malicious:

$$Precision = \frac{TP}{TP\ FP}$$

F_1**-score** is the harmonic mean of precision and recall, providing a balanced measure that considers both false positives and false negatives. This metric is especially useful when dealing with imbalanced datasets, as it captures the trade-off between the completeness and exactness of the model:

$$F_1 - score = 2 \times \frac{Precision \times Recall}{Precision\ Recall}$$

By leveraging this set of complementary metrics, we aim to ensure a thorough and reliable assessment of model performance, allowing for fair comparisons across different algorithms and highlighting their strengths and weaknesses in phishing detection tasks.

5.4. Implementation and Setup

All ML algorithms used for comparison were sourced from the `PyCaret` library[3], a high-level Python framework for automating ML workflows. Each model was fine-tuned using PyCaret's default hyperparameter optimization routine, ensuring a fair and reproducible comparison across all algorithms.

The proposed approach, *AutoPhish*, relies on the GE technique and was implemented using the `PonyGE2` framework [Fenton et al. 2017], a well-established and actively maintained Python library for grammar-based evolutionary computation. `PonyGE2` offers a flexible architecture for symbolic expression generation and has been widely adopted in research for its reliability and extensibility.

The evolutionary configuration employed in this study is summarized in Table 1. A total of 1,000 generations were executed with a population size of 500 individuals. Each individual, or candidate solution, was encoded using genotypes of up to 10,000 codons. The initial population was generated using the `PI_grow` method, a probabilistic initialization strategy that balances exploration and exploitation by varying the depth and structure of the initial trees.

During evolution, genetic diversity was maintained through crossover and mutation. Crossover was applied with a probability of 75%, while mutation occurred with a 10% chance, using the `int_flip_per_codon` strategy, which perturbs codon values within individuals. Selection was conducted via tournament selection, with a tournament size of 2, promoting a steady pressure toward fitter individuals. Additionally, the mutation of duplicate individuals was enabled to avoid premature convergence.

These settings were selected based on preliminary experiments and prior literature to balance convergence speed with population diversity and expressive search space exploration.

[3] `https://pycaret.org/`

Tabela 1. Parameter settings used for Grammatical Evolution (GE) in AutoPhish.

Parameter	Value
Generations	1000
Population size	500
Codon size	10000
Initialization method	PI_grow
Crossover probability	75%
Mutation probability	10%
Mutation operator	int_flip_per_codon
Selection method	Tournament
Tournament size	2
Mutate duplicates	True

6. Results and Discussion

Table 2 presents the performance of *AutoPhish* compared to ten widely used classification algorithms on the phishing detection task. The evaluation was conducted on an imbalanced dataset, where malicious (phishing) instances represent the majority class and benign instances represent the minority. The models were evaluated using four key metrics: Accuracy, Recall, Precision, and F_1-score.

A statistical analysis was conducted to ensure an unbiased evaluation of model performance. The non-parametric Friedman test ($\alpha = 0.05$) was applied to test the null hypothesis that there are no significant differences among model performance metrics. In all cases, the p-values were below the significance threshold, leading to the rejection of the null hypothesis and indicating that at least one model differs statistically from the others. The Nemenyi post hoc test was subsequently used to identify statistically similar groups and compute average rankings. Bold values in the results tables indicate statistically significant differences.

Tabela 2. Performance of classification algorithms on phishing data.

Method	Accuracy	Recall	F_1-score	Precision
AutoPhish	91.74%	**91.62%**	**92.25%**	**91.66%**
LightGBM	**92.64%**	60.28%	66.31%	74.75%
XGBoost	**92.58%**	58.98%	65.52%	74.87%
DT	92.35%	58.32%	64.75%	73.83%
ET	92.35%	56.60%	64.02%	74.53%
GBC	92.22%	58.54%	64.43%	72.46%
RF	92.06%	55.10%	62.59%	73.50%
SVM	91.90%	59.62%	63.97%	69.50%
LR	91.72%	55.95%	61.72%	69.65%
KNN	91.67%	51.00%	59.31%	72.55%
LDA	91.61%	61.99%	63.98%	66.60%

Despite not achieving the highest accuracy, *AutoPhish* outperforms all other methods in terms of Recall, Precision, and F_1-score - the most relevant metrics in the context of imbalanced data. While models such as LightGBM and XGBoost attained

slightly higher accuracy rates (92.64% and 92.58%, respectively), these results are primarily driven by their tendency to favor the majority class (phishing instances), as reflected by their significantly lower recall values (60.28% and 58.98%).

In phishing detection, especially when benign instances are the minority, the cost of misclassifying a benign instance as malicious (false positive) is less critical than failing to detect a benign instance (false negative), which can lead to trust and service issues. Thus, models must prioritize recall and maintain a balanced F_1-score to ensure effective generalization.

AutoPhish demonstrates a superior ability to identify benign instances correctly (high recall) while maintaining high precision, indicating that benign predictions are rarely false. This balance results in the highest F_1-score (92.25%), confirming the effectiveness of the GE strategy in generating customized classifiers that adapt to the data distribution and capture complex patterns.

Additionally, Table 3 presents the algorithm rankings based on the Nemenyi post hoc test across Recall, F_1 Score, and Precision, which are especially critical for imbalanced datasets. AutoPhish achieved a rank 1.0 in all metrics, leading to the lowest average rank overall. The results highlight AutoPhish's superiority in handling such scenarios, outperforming ensemble methods (e.g., LightGBM, XGBoost) and traditional classifiers statistically.

Tabela 3. Ranking of algorithms for each metric (Rank 1 = Best).

Method	Recall Rank	F_1-Score Rank	Precision Rank	Average Rank
AutoPhish	**1.0**	**1.0**	**1.0**	**1.0**
LightGBM	3.0	2.0	3.0	2.7
XGBoost	5.0	3.0	2.0	3.3
DT	7.0	4.0	5.0	5.3
ET	8.0	6.0	4.0	6.0
GBC	6.0	5.0	8.0	6.3
RF	10.0	9.0	6.0	8.3
SVM	4.0	8.0	10.0	7.3
LR	9.0	10.0	9.0	9.3
KNN	11.0	11.0	7.0	9.7
LDA	2.0	7.0	11.0	6.7

Overall, these results validate the competitiveness of *AutoPhish*, not only outperforming classical models but also challenging state-of-the-art ensemble methods and reinforcing its potential for deployment in real-world phishing detection systems where class imbalance and detection robustness are critical factors.

7. Acknowledgements

This research has been supported by the National Council for Scientific and Technological Development (CNPq).

8. Conclusion

This study proposed *AutoPhish*, a novel approach for phishing detection based on GE. Unlike traditional machine learning models that rely on fixed structures and hyperparameter tuning, *AutoPhish* can automatically generate customized and optimized classifiers through evolutionary search guided by a domain-specific grammar. We evaluated *AutoPhish* using a realistic, imbalanced dataset that reflects the nature of real-world phishing detection scenarios, where malicious instances vastly outnumber benign ones. The approach was compared against ten widely adopted machine learning algorithms, including state-of-the-art ensemble methods like LightGBM and XGBoost. Results showed that *AutoPhish* achieved the highest performance in terms of Recall, Precision, and F_1-score, the most relevant metrics in the context of class imbalance. Although some algorithms slightly outperformed *AutoPhish* in raw accuracy, they exhibited poor recall and F_1-score due to their bias toward the majority class. In contrast, *AutoPhish* demonstrated a balanced and robust detection capability, correctly identifying phishing and benign instances with high confidence. The strong performance of *AutoPhish* underscores the value of evolutionary computation in cybersecurity applications, particularly for tasks requiring interpretability, adaptability, and resistance to data imbalance. Future work includes extending the grammar to incorporate temporal and contextual features and exploring multi-objective optimization strategies to evolve models that simultaneously trade-off multiple performance goals. Integration into real-time detection environments and evaluation on larger-scale datasets would further validate its practical utility.

Referências

Barros, M., Silva, C., and de Miranda, P. (2019). Adoção da seleção de características como mecanismo antiphishing: aplicabilidade e impactos. In *Encontro Nacional de Inteligência Artificial e Computacional (ENIAC)*, pages 214–225. SBC.

Basgalupp, M. P., Barros, R. C., Cerri, R., Neri, F., Miranda, P. B., and Ludermir, T. (2025). Grammar-based evolutionary approaches for software effort estimation. In *2025 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–4. IEEE.

Bountakas, P., Koutroumpouchos, K., and Xenakis, C. (2021). A comparison of natural language processing and machine learning methods for phishing email detection. *Proceedings of the 16th International Conference on Availability, Reliability and Security*.

da Silva, C. A., Miranda, P. B., and Cordeiro, F. R. (2021). A new grammar for creating convolutional neural networks applied to medical image classification. In *2021 34th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, pages 97–104. IEEE.

da Silva, C. A., Rosa, D. C., Miranda, P. B., Cordeiro, F. R., Si, T., Nascimento, A. C., Mello, R. F., and de Mattos Neto, P. S. (2023a). A novel multi-objective grammar-based framework for the generation of convolutional neural networks. *Expert Systems With Applications*, 212:118670.

da Silva, C. A., Rosa, D. C., Miranda, P. B., Si, T., Cerri, R., and Basgalupp, M. (2023b). Automated cnn optimization using multi-objective grammatical evolution. *Applied Soft Computing*, page 111124.

de Barros, M., da Silva, C., and de Miranda, P. (2019). Aplicabilidade e impactos quanto a adoção de modelos de classificação como mecanismos anti-phishing. In *Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais (SBSeg)*, pages 39–42. SBC.

de Barros, M. L., da Silva, C. M., and de Miranda, P. B. (2020). Xphide: Um sistema especialista para a detecçao de phishing. In *Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais (SBSeg)*, pages 161–174. SBC.

Diniz, J. B., Cordeiro, F. R., Miranda, P. B., and da Silva, L. A. T. (2018). A grammar-based genetic programming approach to optimize convolutional neural network architectures. In *Encontro Nacional de Inteligência Artificial e Computacional (ENIAC)*, pages 82–93. SBC.

Fadheel, W., Abusharkh, M., and Abdel-Qader, I. (2017). On feature selection for the prediction of phishing websites. *2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech)*, pages 871–876.

Fenton, M., McDermott, J., Fagan, D., Forstenlechner, S., Hemberg, E., and O'Neill, M. (2017). Ponyge2: Grammatical evolution in python. In *Proceedings of the genetic and evolutionary computation conference companion*, pages 1194–1201.

Fette, I., Sadeh, N., and Tomasic, A. (2007). Learning to detect phishing emails. In *Proceedings of the 16th international conference on World Wide Web*, pages 649–656.

Habib, P., Sharma, U., and Sethi, K. (2022). Phishing detection with machine learning. *International Journal for Research in Applied Science and Engineering Technology*.

Hasan, K. M. Z., Hasan, M. Z., and Zahan, N. (2019). Automated prediction of phishing websites using deep convolutional neural network. *2019 International Conference on Computer, Communication, Chemical, Materials and Electronic Engineering (IC4ME2)*, pages 1–4.

Lingam, G., Rout, R. R., Somayajulu, D., and Ghosh, S. (2021). Particle swarm optimization on deep reinforcement learning for detecting social spam bots and spam-influential users in twitter network. *IEEE Systems Journal*, 15:2281–2292.

Mahajan, R. and Siddavatam, I. A. (2018). Phishing website detection using machine learning algorithms. *International Journal of Computer Applications*.

Miranda, P. B. and Prudêncio, R. B. (2020). A novel context-free grammar for the generation of pso algorithms. *Natural Computing*, 19(3):495–513.

Miranda, P. B., Prudêncio, R. B., and Pappa, G. L. (2017). H3ad: A hybrid hyper-heuristic for algorithm design. *Information Sciences*, 414:340–354.

O'Neill, M. and Ryan, C. (2004). Grammatical evolution by grammatical evolution: The evolution of grammar and genetic code. In *European Conference on Genetic Programming*, pages 138–149. Springer.

Peng, T., Harris, I., and Sawa, Y. (2018). Detecting phishing attacks using natural language processing and machine learning. *2018 IEEE 12th International Conference on Semantic Computing (ICSC)*, pages 300–301.

Si, T., Miranda, P., Galdino, J. V., and Nascimento, A. (2021). Grammar-based automatic programming for medical data classification: an experimental study. *Artificial Intelligence Review*, 54:4097–4135.

Tharwat, A. (2021). Classification assessment methods. *Applied Computing and Informatics*, 17(1):168–192.

Yadollahi, M. M., Shoeleh, F., Serkani, E., Madani, A., and Gharaee, H. (2019). An adaptive machine learning based approach for phishing detection using hybrid features. *2019 5th International Conference on Web Research (ICWR)*, pages 281–286.

Yaswanth, P. and Nagaraju, V. (2023). Prediction of phishing sites in network using naive bayes compared over random forest with improved accuracy. *2023 Eighth International Conference on Science Technology Engineering and Mathematics (ICONSTEM)*, pages 1–5.

Zamir, A., Khan, H., Iqbal, T., Yousaf, N., Aslam, F., Anjum, M. A., and Hamdani, M. (2020). Phishing web site detection using diverse machine learning algorithms. *Electron. Libr.*, 38:65–80.

Zhu, E., Ye, C., Liu, D., Liu, F., Wang, F., and Li, X. (2018). An effective neural network phishing detection model based on optimal feature selection. *2018 IEEE Intl Conf on Parallel Distributed Processing with Applications, Ubiquitous Computing Communications, Big Data Cloud Computing, Social Computing Networking, Sustainable Computing Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom)*, pages 781–787.