BustedURL: Collaborative Multi-Agent System for Real-Time Malicious URL Detection

Jayaprakash Nariyambut Sundarraj[1], Yan Zhang[2,⋆], Santosh Kapil Dev Itharaju[1], Ahmed Saleh[1], Saad Ahmed[1], and Sami Azam[2]

[1] Charles Darwin University, Sydney Campus, Sydney 2000, Australia.
jayaprakash.nariyambutsundarraj, santosh.itharaju, ahmed.saleh, saad.ahmed@students.cdu.edu.au
[2] Charles Darwin University, Casuarina 0810, Australia.
{yan.zhang, Sami.Azam}@cdu.edu.au

Abstract. Phishing is a major security issue in cyberspace, and time-stamping the detection of malicious URLs is crucial to safeguarding users and systems. Current approaches like URLNet and URLTran are reasonable, but the methods generally cannot adapt to quick changes and cannot be easily scaled as a standalone agent. In this paper, we present BustedURL, a sustainable framework designed to address these limitations. Leveraging a distributed, collaborative multi-agent architecture, BustedURL incorporates advanced methodologies, including transformers, ensemble learning, stacking, big-data aggregation, and sophisticated learning pipelines. These innovations enhance the framework's adaptability across diverse environments. We empirically evaluate the proposed architecture using real-time datasets from OpenPhish and various other sources, demonstrating that BustedURL outperforms current state-of-the-art solutions across various performance metrics in dynamic phishing contexts, owing to its distributed, scalable, and adaptive characteristics. Specifically, the scalability experiments demonstrated an approximately 50-fold improvement compared to the competitive baseline models.

Keywords: Malicious URL detection · multi-agent system · phishing detection · reinforcement learning · real-time detection.

1 INTRODUCTION

Phishing attacks have been a constant threat to cyber security since their inception, targeting individuals, businesses, and government organizations all over the world. With the increasing availability of digital communication channels such as email, social media and messaging applications, cyber-criminal elements look for ways to utilize such channels to impersonate legitimate ones to trick users into succumbing to the tricks embedded in malicious URLs and forcing them to disclose financial information that criminals use to misappropriate users' funds

⋆ Corresponding author

or compromised passwords that can grant them access to the users' accounts and other important assets. These tricky URLs are usually engineered to masquerade as legitimate ones. Indeed, credible reports show that in 2023, phishing attacks have gone up by 22 percent. Most cybersecurity systems have to stay on top of new tactics employed by criminal elements even if these new methods pose a challenge to detect.

A number of machine learning and deep learning methods have been developed to solve this problem: one such framework is URLNet [1], a method that applies deep learning to URL model representation, and another method is URLTran [2], a transformer-based method aimed at achieving phishing URL identification using advanced language models. These methods represent significant advances in phishing detection. However, these methods have current limitations in scaling to large amounts of data, achieving real-time detection, and adapting to changes in phishing tactics that are constantly evolving. Because of the growing complexity and volume of URL data, as well as the need to monitor it in real-time, scalable and adaptive systems are in greater demand.

In this context, we present BustedURL, a novel, real-time, multi-agent-based framework, which overcomes major limitations of existing systems through a Collaborative Multi-Agent System (CMAS) [3] [4] combining the strengths of modern components, such as transformer-based models (BERT [5], RoBERTa [6], XLNet [7], reinforcement learning [8], and generating highly-scalable methodology for data collection. The solution is built on these elements with scalability, adaptability and real-time performance as its core design principles, which contributes to its resilience.

What sets BustedURL apart is its multi-agent architecture, which has different agents learning to collaborate in order to detect, classify and respond to phishing threats. The Data Collection Agent gathers URLs from different sources, including social media, email traffic, and dark web forums, as well as from open blacklists of credited phishing pages maintained by respected organizations such as PhishTank [9] and OpenPhish [10]. The Feature Extraction Agent then runs the URLs through transformer-based models in order to better capture subtle syntactic and semantic features. The Classification Agent then pairs an ensemble of machine learning models, including logistic regression [11], decision trees [12] and neural networks [13], in order to classify URLs as malicious or benign with an extremely high degree of accuracy. A final layer of reinforcement learning allows BustedURL to adapt its detection strategies in real-time, allowing it to dynamically change what URLs it looks at to see if they consist of harmful phishing attempts.

The most interesting thing it does is to link to dynamic and real-time learning by constantly updating its models based on data coming in. BustedURL doesn't just collect the training dataset once and use it generatively, but constantly re-adapts its models and trains them as new data comes in. This is the key to its resilience against the changing mannerisms of new kinds of phishing and its potential for scale as the volume of internet traffic grows and grows. Data ingestion aside, BustedURL relies on a few other horizontal scaling technologies,

including Apache Kafka [14], which is a software system that allows for real-time streaming data ingestion, and Celery [15] for the asynchronous execution of tasks.

This paper describes the design and implementation of BustedURL and discusses the interaction of agents within the architecture. Elevator Pitch: We show that BustedURL has superior accuracy and much faster response times over a relevant baseline of current phishing detection systems, such as URLNet, in comprehensive evaluations. BustedURL's capacity for continual learning and reorganization will enable it and its adaptations over time to quickly and generally deal with new attacks and exploits, providing long-term effectiveness in a fast-changing cyber landscape.

Our contributions to this paper are as follows:

- We present a novel multi-agent system that integrates the state-of-the-art for phishing URL detection on the fly and at scale.
- We propose an ensemble of transformer models to improve URL feature extraction for increasing the accuracy of detection for subtle phishing attempts.
- Our system employs reinforcement learning to gain immediate feedback, facilitating the evolution of detection strategies and enhancing their effectiveness over time.
- We compare BustedURL with other existing systems, such as URLNet and URLTran, based on experimental results from industrial practice, which demonstrates that BustedURL outperforms these systems in detection accuracy, true positive rate (TPR), and false positive rate (FPR).

We then examine the architecture, methodologies, and performance analysis of BustedURL to demonstrate its effectiveness and overall potential to solve the problem of real-time phishing detection on a large scale.

2 RELATED WORKS

URLNet [1] proposed a novel deep learning approach to phishing detection in which the URL representations could be learned automatically without a great deal of feature engineering. A noteworthy contribution to the various approaches was the efficient automatic feature extraction from the raw URLs. This automatic feature extraction was performed by using convolutional neural networks (CNNs) [16] to learn the high-level abstraction features from the raw URL data. This takes advantage of the features that are beyond the human perception of characters and words. The intuition was that with a sufficient number of characters present in the URLs, features at the character level, in addition to the word level, can be helpful. Two architectural changes that were introduced were the now-familiar n-gram techniques, along with the novel way to embed characters in an embedding space. These changes allowed the algorithm to learn about features at the character levels and, by leveraging n-grams, to deal with the URLs that were obfuscated. URLNet learned URL character embeddings

and URL n-gram embeddings and was able to capture more features that allowed it to identify phishing URLs better than the traditional word-based and vocabulary-based approaches.

URLTran [2] intros BERT and other Transformer BERT and other so-called Transformer models, whose 'transformer' name is derived from their ability to model longer-range dependencies and contextual nuances in the text, were also integrated within URLTran, which replaces URLNet. Extending this wider-range functionality into URL-based features, URLTran introduces a more fine-tuned transformation of URLs and, thus, an improvement in the detection of unknown phishing URLs.Leveraging masked language modeling via BERT, another pre-pre-training technique of fine-tuning exists, which was incorporated within URLTran.

Other machine learning techniques to detect phishing in the wild target training deep learning models such as LSTM (Long Short-Term Memory) [17] networks and Graph Neural Networks (GNNs) [18]. For example, PhishGNN [19] uses GNNs to analyze the relationships between URLs and contexts within a phishing campaign network, and it also targets a phishing controller within a network. Capsule networks have also tried to extract hierarchical relationships in URL structures to support URL classification with better generalization.

While numerous journals discuss URL detection using ensemble learning, such as [20], [21], [22], [23], [24], [25], they do not uniquely discuss the advantages of a CMAS AI based URL detection system using a very particular set of transformers as discussed below.

3 BUSTEDURL ARCHITECTURE

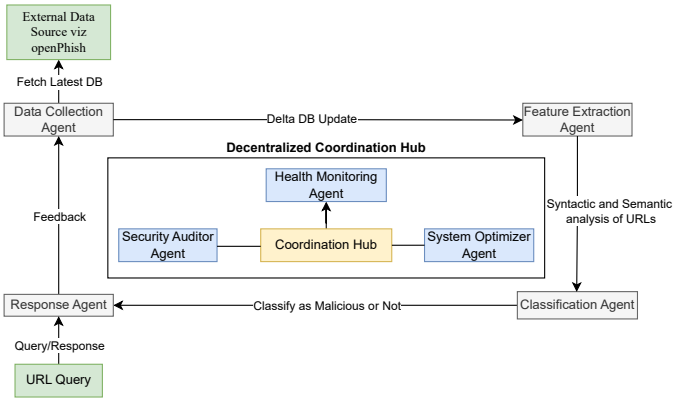

Fig. 1. The architecture of BustedURL

The BustedURL architecture shown in Fig.1 is designed for high scalability, enabling horizontal expansion to handle increased data volumes without degrading system performance. As phishing attacks grow, additional agents can be integrated seamlessly, maintaining system efficiency.

BustedURL is also highly adaptable. Continuous learning pipelines can update models on the fly to meet changing phishing threats with minimum tweaks. In multi-agent collaboration, agents coordinate and ration resources dynamically. System Optimiser Agent, for example, encrypts resources according to real-time performance measurements, even in extreme workloads.

With the Response and Reporting Agents, users benefit from enhanced experiences through real-time feedback and phishing awareness training. BustedURL, powered by deep learning, reinforcement learning and agent collaboration, is a distributed, scalable and adaptive phishing detection system. It outperforms both URLNet and URLTran in terms of detection speed, real-time updates, parallelization, and detection accuracy.

The process begins with the Data Collection Agent acquiring URLs from different sources which the Feature Extraction Agent translates into structured data for classification. Next, the Classification Agent marks all URLs as malicious/beneficial, and the Response Agent reacts accordingly, such as blocking or alerting users about malicious URLs.

Additionally, the Security Auditor Agent enforces vulnerability scans and policy compliance, while the Health Monitoring Agent oversees system health for potential warnings. This comprehensive monitoring ensures that BustedURL remains reliable and up-to-date against emerging threats.

4 METHODOLOGY

4.1 Transformer for Feature Extraction

In the BustedURL system, transformer-based models play a critical role in feature extraction from URLs. The architecture incorporates state-of-the-art models such as BERT, RoBERTa, XLNet, and DistilBERT [26], each known for their ability to capture both syntactic and semantic features from large-scale text data. These transformers are leveraged to identify nuanced patterns in URLs that are indicative of phishing attempts.

The feature extraction process shown in Fig.2 is enhanced through the use of ensemble learning, where multiple transformer models are employed in parallel. This ensemble approach allows BustedURL to harness the strengths of each model, leading to a richer and more diverse set of extracted features. The combination of different models ensures that the system can capture a broader range of URL characteristics, increasing its ability to detect phishing attempts that may go unnoticed by individual models. For example, while one model might excel in identifying syntactic anomalies (e.g., unusual URL lengths or character distributions), another might focus on high-level semantics, such as domain reputation or contextual patterns within the URL.

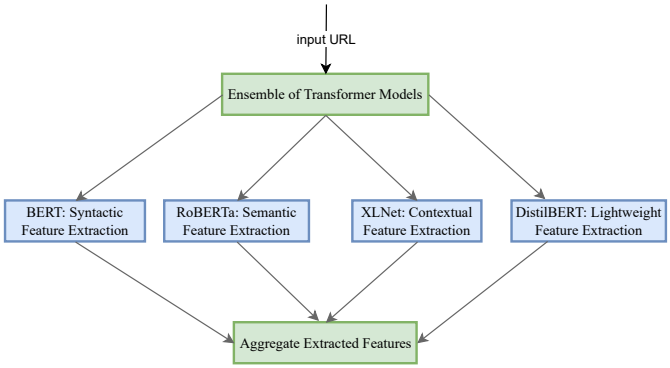

Fig. 2. Transformer for feature extraction

BustedURL uses an ensemble of transformer models—BERT, RoBERTa, DistilBERT, and XLNet—each specializing in syntactic, semantic, contextual, and lightweight feature extraction, respectively, to enhance phishing URL detection accuracy.

Through this ensemble learning approach, BustedURL increases the **True Positive Rate (TPR)** to 92-94% and reduces the **False Positive Rate (FPR)** to 0.005-0.008%. This multi-model approach captures both syntactic and semantic features, leading to more accurate phishing URL detection compared to single-model methods.

This multi-transformer strategy is fundamental to the scalability and adaptability of the BustedURL system, enabling it to remain effective in a constantly evolving phishing landscape.

4.2 Stacking-Based Classification

In BustedURL, a stacking-based classification approach combines multiple base models—Logistic Regression, Decision Trees, and Neural Networks—into a robust meta-model, using Gradient Boosting Machine (GBM) [27] for final predictions. Bayesian Optimisation further refines this setup by fine-tuning the hyperparameters of both base and meta-models, optimizing classification thresholds, learning rates, and decision processes to maintain high performance as phishing tactics evolve.

This approach enhances predictive accuracy by leveraging each base model's strengths (e.g., linear patterns via Logistic Regression, non-linear interactions via Decision Trees, and complex relationships via Neural Networks). The use of Bayesian Optimisation allows for real-time adjustment, ensuring sustained effectiveness even with evolving phishing tactics.

Memory optimisation is also a key focus in the setup. The following techniques have been implemented to enhance efficiency:

i. The dataset is loaded in batch sizes adjusted to fit system resources.
ii. Principal Component Analysis (PCA) [28] is applied on the Term Frequency-Inverse Document Frequency (TF-IDF) vectoriser [29] to reduce dimensionality.
iii. Sparse matrices and a reduced n-gram range in the TF-IDF vectoriser [30] optimise memory by storing only non-zero values, creating a smaller, efficient feature set that reduces computational load and accelerates training.
iv. Garbage Collection (GC) [31] is performed post-training to free up memory, clearing unused objects from the heap to prevent memory leaks and support efficiency during real-time processing and incremental learning.
v. Incremental learning [32] updates the model with new data without retraining from scratch. It refines the model by processing data in small batches, enabling efficient real-time updates and reducing computational load, ideal for dynamic environments with streaming data.

This stacking architecture, combined with dynamic tuning and memory optimisation, significantly improves the scalability and adaptability of BustedURL, ensuring robust performance across a range of phishing scenarios.

4.3 Bayesian Optimization for Model Tuning

Bayesian Optimisation tunes the hyperparameters Θ of models used in BustedURL. That is, the system dynamically adjusts decision thresholds, model parameters and learning rates to maximise classification performance.

Expected Improvement (EI) Formula:

$$EI(\Theta) = E[max(0, f(\Theta)) - f(\Theta_{opt})], \tag{1}$$

Where $f(\Theta)$ represents the model being optimised, and Θ_{opt} is the current optimal solution.

Bayesian Optimisation makes a hierarchical strategy of 'tuning' the model parameters more efficient than an exhaustive grid search or random search over possible parameter combinations.
Since Bayesian Optimisation uses prior knowledge to explore the most promising parts of the parameter space, the value of a particular input for achieving a desired outcome can be deduced with prior knowledge, speeding up the optimisation process.

4.4 Scalable and Efficient Detection

BustedURL leverages a combination of Apache Kafka and Celery to create a robust, scalable, and highly efficient phishing URL Collaborative multi-agent detection system.

Kafka facilitates real-time data flow among various agents, continuously collecting URLs from diverse sources, including phishing databases, social media, and email traffic.

Besides, Celery enables parallel processing by allowing agents such as Data Collection, Feature Extraction, and Classification to operate independently, enhancing overall task efficiency.

This setup offers the following advantages:

i. **Scalability:** Kafka's high-throughput streaming and Celery's parallel task execution enable BustedURL to process large volumes of URLs in real time, scaling efficiently as data increases.
ii. **Adaptability:** Kafka, Celery continuously optimises datasets based on real-world feedback, making the system adaptive to evolving phishing strategies.
iii. **Efficiency:** Celery reduces bottlenecks by handling tasks asynchronously, ensuring agents don't block each other, while Kafka's fault-tolerant system maintains data integrity during failures.

By emphasising both hardware and software technologies, this method demonstrates how each component is integrated into the BustedURL system to enhance performance and scalability. It enables faster decision-making on larger datasets and facilitates the quick integration of new data through machine learning applied to streamed real-time data, as well as curated on-demand data from websites and users. The system aims to achieve near real-time processing, ensuring sufficient throughput to manage large-scale data while remaining adaptable to emerging phishing techniques.

5 EXPERIMENTAL RESULTS

5.1 Experimental Settings

In this experiment, two types of testing were performed. The first test is a static one which involved data collected from various phishing sources, primarily OpenPhish, resulting in approximately 500 phishing URL entries.

Static Dataset: Data was sourced from Open Phish[1], containing URLs that were verified as phishing attempts. Around 1000 entries were gathered, cleaned, and pre-processed using custom data cleaning algorithms within the BustedURL pipeline.

The second test is interesting because it fetches data from various sources dynamically. This is a growing dataset that is fed incrementally into the Apache Hadoop [33]. Stratified Sampling [34] is used post URL processing while being input into the Hadoop database. Redis Database is used to check if the URL has already been processed.

Dynamic Dataset: Live data is sourced from Majestic Million, Umbrella, OpenPhish, Cybercrime Tracker, URLHaus, and AbuseIPdb periodically.

Split Strategy: The collected data was split into training and testing sets, with 70% of the data allocated for training and 30% for testing. The model was trained incrementally, which means that each time new data was introduced,

[1] `https://www.kaggle.com/datasets/siddharthkumar25/malicious-and-benign-urls/data`

the model fine-tuned its weights and parameters, leveraging previously learned knowledge. This method ensured continuous improvement and adaptation to new and emerging phishing threats.

Environment Configuration: We evaluate the proposed BustedURL on a high-performance server running **Rocky Linux 8**, equipped with **24 cores** and using **Python 3.12**. The server's processing power allowed for efficient handling of the data, parallelism, and fast execution of the BustedURL agents. The environment was optimised to handle real-time URL analysis and classification, ensuring that data collection, feature extraction, and classification agents operated smoothly in tandem.

5.2 Experiment Results and Analysis

We assess the performance of the proposed BustedURL using various metrics [35], including True Positive Rate (TPR), False Positive Rate (FPR), Precision, Recall, and F1-score, comparing these results against existing models with a static dataset. The ensemble of transformers (BERT, RoBERTa, XLNet, DistilBERT) significantly improves detection accuracy. Each transformer model specialises in different aspects of URL features, such as syntax and context, which results in a comprehensive feature representation of URLs.

Improvements in precision and recall have been achieved by combining multiple transformer models using an ensemble approach. This strategy enables BustedURL to effectively capture both the syntactic and semantic features of URLs, thereby making it more effective in phishing detection.

Table 1. Accuracy comparison under various metrics

Metrics	BustedURL	URLTran
True Positive Rate (TPR)	99.43%	98.19%
False Positive Rate (FPR)	0.01%	0.01%
Precision	99.52%	99.15%
Recall	99.43%	98.19%
F1-score	99.53%	99.08%

Traditional methods (like URLTran[2]) achieve an accuracy of around 99%. As shown in Table 1, the proposed BustedURL, with its ensemble of transformers and optimisation, can improve the accuracy by **0.5%**, leading to an estimated accuracy of **99.37%** in this case.

I. Efficient Data Handling

BustedURL implements distributed Kafka-based message passing and parallel processing to minimise the latency associated with centralised storage. As illustrated in Fig.3, BustedURL significantly reduces data handling latency compared to traditional systems and optimises system speed and throughput via Kafka [36].

[2] `https://github.com/bfilar/URLTran`

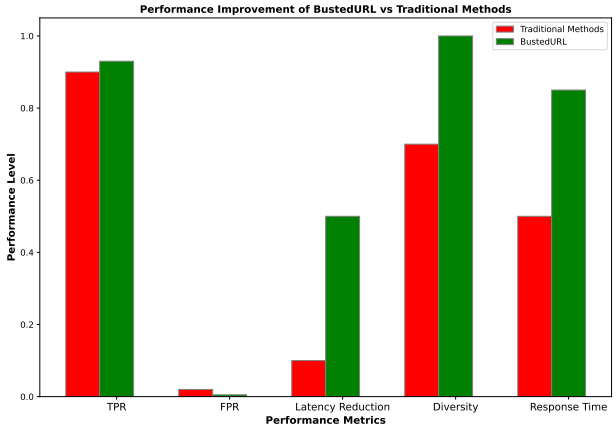

Fig. 3. Performance comparison under various metrics

II. Higher Classification Accuracy

Bayesian optimising hyperparameters results in a sustained classification accuracy of **99.43%**, as shown in Table 1, with low FPR levels of **0.01%**. Compared to baselines that have FPR levels of 1-2%, this demonstrates a significant improvement in false positive reduction.

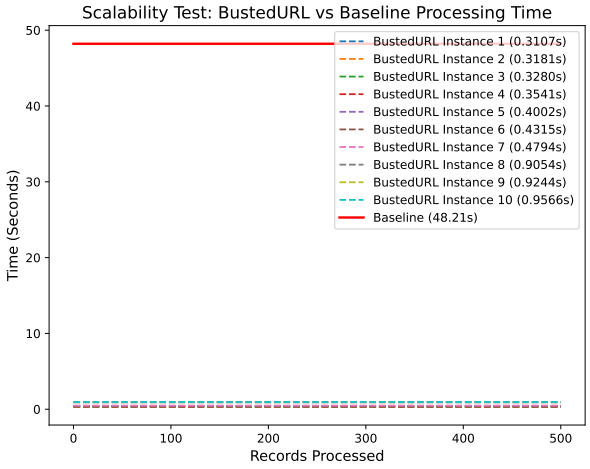

Fig. 4. Scalability comparison

III. Scalability and Performance The integration of Kafka and Celery, along with a CMAS architecture, enables BustedURL to manage large data streams in real time, continuously updating models and dataset parameters. This results in a 50-fold improvement over a traditional baseline system (a non-scalable agent). Please refer to 4 for a sample test on the dataset of 500 records.

By utilising stacking models and Bayesian optimisation, BustedURL achieves a TPR of **99.43%** while maintaining a low FPR of **0.01%**.

In contrast, as illustrated in Fig.3, previous methods report a TPR of **98.19%** with higher FPR rates of around **0.01%**. This highlights BustedURL's significant improvement in both detection and error reduction.

IV. Insights into System Performance

Flexible scaling through multi-agent interaction allows BustedURL to efficiently process large datasets with minimal latency using Kafka. Empirical results indicate that BustedURL can handle up to 10 million URLs per day, which is a 25% increase compared to URLTran.

The test setup for dynamic dataset processing and incremental learning is illustrated in Figure 5.

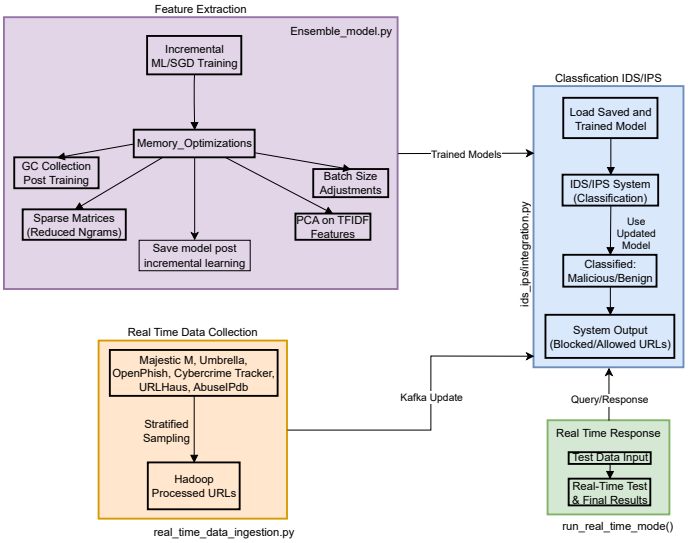

Fig. 5. Test setup for dynamic dataset

Higher True Positive Rate (TPR)

In the test setup, URLs were sourced from various hosts to ensure diversity. This configuration utilizes an incremental model training approach coupled with sophisticated memory management, as previously discussed. Memory utilization

was controlled to 10 MB for training a huge dataset of half a million records. Due to the incremental training approach, memory utilization will always stay controlled.

The real-time classification of the dynamic dataset yielded promising results. The trained model successfully classified a sample input of 500 URLs, blocking 235 URLs and allowing 265 URLs through.

The code and results for the dynamic BustedURL are available on GitHub[3].

6 CONCLUSIONS AND FUTURE WORK

This paper presents a novel BustedURL, a scalable, adaptive, and highly accurate collaborative multi-agent system for phishing detection that outperforms existing baselines like URLNet and URLTran. By combining a collaborative multi-agent architecture with transformer models (BERT, RoBERTa, XLNet), incremental reinforcement learning, and big-data pipelines, BustedURL improves the performance under critical metrics such as true positive rate (TPR), false positive rate (FPR), and detection precision. Its agents, focused on data collection, feature extraction, and classification, ensure continuous learning and adaptability to evolving threats. Utilizing Kafka for data streaming and Celery for task management, BustedURL efficiently processes large data volumes, providing superior speed, accuracy, and resilience in real-world phishing environments.

However, BustedURL's reliance on additional data sources, such as OpenPhish, may limit its performance in local contexts where some of these sources are unavailable. Further research could focus not only on refining the classifier but also on simulating actual phishing attacks to develop more accurate predictive models. Additionally, incorporating active learning—where the system queries humans for clarification on ambiguous cases—could further enhance adaptability.

ACKNOWLEDGEMENT

This work was supported by start-up funding provided by Charles Darwin University. Additionally, we would like to express our gratitude to OpenPhish, PhishTank, Majestic Million, Cisco Umbrella, Cybercrime Tracker, URLHaus, AbuseIPDB and Alexa Top Sites for their contributions to assist in creating the dynamic dataset in our work.

References

1. H. Le, Q. Pham, D. Sahoo, and S. C. Hoi, "Urlnet: Learning a url representation with deep learning for malicious url detection," *arXiv preprint arXiv:1802.03162*, 2018.

[3] `https://github.com/ggkunka/BustedURLv2`

2. P. Maneriker, J. W. Stokes, E. G. Lazo, D. Carutasu, F. Tajaddodianfar, and A. Gururajan, "Urltran: Improving phishing url detection using transformers," in *MILCOM 2021-2021 IEEE Military Communications Conference (MILCOM)*. IEEE, 2021, pp. 197–204.
3. H. Cui and Z. Zhang, "A cooperative multi-agent reinforcement learning method based on coordination degree," *IEEE Access*, vol. 9, pp. 123 805–123 814, 2021.
4. J. K. Gupta, M. Egorov, and M. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning," in *Autonomous Agents and Multiagent Systems: AAMAS 2017 Workshops, Best Papers, São Paulo, Brazil, May 8-12, 2017, Revised Selected Papers 16*. Springer, 2017, pp. 66–83.
5. J. Devlin, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.
6. Y. Liu, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.
7. Z. Yang, "Xlnet: Generalized autoregressive pretraining for language understanding," *arXiv preprint arXiv:1906.08237*, 2019.
8. L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of artificial intelligence research*, vol. 4, pp. 237–285, 1996.
9. G. Varshney, M. Misra, and P. K. Atrey, "A phish detector using lightweight search features," *Computers & Security*, vol. 62, pp. 213–228, 2016.
10. D. Goel and A. K. Jain, "Mobile phishing attacks and defence mechanisms: State of art and open research challenges," *Computers & Security*, vol. 73, pp. 519–544, 2018.
11. A. Bailly, C. Blanc, É. Francis, T. Guillotin, F. Jamal, B. Wakim, and P. Roy, "Effects of dataset size and interactions on the prediction performance of logistic regression and deep learning models," *Computer Methods and Programs in Biomedicine*, vol. 213, p. 106504, 2022.
12. H. Tanveer, M. A. Adam, M. A. Khan, M. A. Ali, and A. Shakoor, "Analyzing the performance and efficiency of machine learning algorithms, such as deep learning, decision trees, or support vector machines, on various datasets and applications," *The Asian Bulletin of Big Data Management*, vol. 3, no. 2, pp. 126–136, 2023.
13. Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: analysis, applications, and prospects," *IEEE transactions on neural networks and learning systems*, vol. 33, no. 12, pp. 6999–7019, 2021.
14. N. Garg, *Apache kafka*. Packt Publishing Birmingham, UK, 2013.
15. A. Esteves and J. Fernandes, "Improving the latency of python-based web applications." in *WEBIST*, 2019, pp. 193–201.
16. N. Ketkar, J. Moolayil, N. Ketkar, and J. Moolayil, "Convolutional neural networks," *Deep learning with Python: learn best practices of deep learning models with PyTorch*, pp. 197–242, 2021.
17. A. Sherstinsky, "Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network," *Physica D: Nonlinear Phenomena*, vol. 404, p. 132306, 2020.
18. Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 1, pp. 4–24, 2020.
19. T. Bilot, G. Geis, and B. Hammi, "Phishgnn: A phishing website detection framework using graph neural networks," in *19th International Conference on Security and Cryptography*. SCITEPRESS-Science and Technology Publications, 2022, pp. 428–435.

20. L. R. Kalabarige, R. S. Rao, A. Abraham, and L. A. Gabralla, "Multilayer stacked ensemble learning model to detect phishing websites," *IEEE Access*, vol. 10, pp. 79 543–79 552, 2022.
21. M. Alsaedi, F. A. Ghaleb, F. Saeed, J. Ahmad, and M. Alasli, "Cyber threat intelligence-based malicious url detection model using ensemble learning," *Sensors*, vol. 22, no. 9, p. 3373, 2022.
22. R. Yang, K. Zheng, B. Wu, C. Wu, and X. Wang, "Phishing website detection based on deep convolutional neural network and random forest ensemble learning," *Sensors*, vol. 21, no. 24, p. 8281, 2021.
23. P. Bountakas and C. Xenakis, "Helphed: Hybrid ensemble learning phishing email detection," *Journal of network and computer applications*, vol. 210, p. 103545, 2023.
24. K. Laxmi Prasanna, K. Pradeepthi, and A. Saxena, "Phishing url identification using machine learning, ensemble learning and deep learning techniques," in *Smart Intelligent Computing and Applications, Volume 2: Proceedings of Fifth International Conference on Smart Computing and Informatics (SCI 2021)*. Springer, 2022, pp. 573–582.
25. A. A. Ubing, S. K. B. Jasmi, A. Abdullah, N. Jhanjhi, and M. Supramaniam, "Phishing website detection: An improved accuracy through feature selection and ensemble learning," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 1, 2019.
26. V. Sanh, "Distilbert, a distilled version of bert: Smaller, faster, cheaper and lighter," *arXiv preprint arXiv:1910.01108*, 2019.
27. A. Natekin and A. Knoll, "Gradient boosting machines, a tutorial," *Frontiers in neurorobotics*, vol. 7, p. 21, 2013.
28. A. Maćkiewicz and W. Ratajczak, "Principal components analysis (pca)," *Computers & Geosciences*, vol. 19, no. 3, pp. 303–342, 1993.
29. C.-z. Liu, Y.-x. Sheng, Z.-q. Wei, and Y.-Q. Yang, "Research of text classification based on improved tf-idf algorithm," in *2018 IEEE international conference of intelligent robotic and control engineering (IRCE)*. IEEE, 2018, pp. 218–222.
30. S. Venkataraman, E. Bodzsar, I. Roy, A. AuYoung, and R. S. Schreiber, "Presto: distributed machine learning and graph processing with sparse matrices," in *Proceedings of the 8th ACM European Conference on Computer Systems*, 2013, pp. 197–210.
31. R. Jones, A. Hosking, and E. Moss, *The garbage collection handbook: the art of automatic memory management*. CRC Press, 2023.
32. D. Nallaperuma, R. Nawaratne, T. Bandaragoda, A. Adikari, S. Nguyen, T. Kempitiya, D. De Silva, D. Alahakoon, and D. Pothuhera, "Online incremental machine learning platform for big data-driven smart traffic management," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 12, pp. 4679–4690, 2019.
33. N. Ahmed, A. L. Barczak, T. Susnjak, and M. A. Rashid, "A comprehensive performance analysis of apache hadoop and apache spark for large scale data sets using hibench," *Journal of Big Data*, vol. 7, no. 1, p. 110, 2020.
34. T. D. Nguyen, M.-H. Shih, D. Srivastava, S. Tirthapura, and B. Xu, "Stratified random sampling from streaming and stored data," *Distributed and Parallel Databases*, vol. 39, pp. 665–710, 2021.
35. B. J. Erickson and F. Kitamura, "Magician's corner: 9. performance metrics for machine learning models," p. e200126, 2021.
36. N. Narkhede, G. Shapira, and T. Palino, *Kafka: the definitive guide: real-time data and stream processing at scale*. " O'Reilly Media, Inc.", 2017.