The Impact of Feature Selection on Malware Classification Using Chi-Square and Machine Learning

Abstract—**The Internet of Things (IoT) is a network of physical objects, automobiles, household appliances, and other items that are integrated with sensors, software, and connections to gather and share data via the Internet. The rapid proliferation of Internet of Things (IoT) devices has ushered in a wave of new security challenges, specifically in the realm of malware detection. These challenges necessitate innovative solutions. Consequently, the primary objective of this study is to develop an advanced malware detection system leveraging machine learning algorithms in tandem with Natural language processing (NLP) techniques like tokenization and vectorization in addition to a feature selection method named Chi-Square. The new method is tested using the IoTPot dataset and compared with recent research in the field, where it outperformed the current work with respect to accuracy, F1-score, recall, and precision. Furthermore, the new method was compared with time-based consulting and demonstrated superior performance with NLP and Chi-square than without, making it more suitable for resources constrained by such IoT systems. We also provide the code for the proposed method to foster transparency [1].**

Index Terms—**NLP, Machine learning, Malware detection, Chi-Square, feature selection**

I. Introduction

The Internet of Things (IoT) is a rapidly growing technology that enables physical devices and everyday objects to connect to the Internet, facilitating the transmission and reception of data [2], [14]. IoT devices encompass a wide range of items, including smartphones, smart homes, wearable, and even industrial machines [7]. The ultimate goal of IoT is to enhance the efficiency, convenience, and precision of both personal and business operations. However, with the rapid expansion of IoT, the associated challenges have become increasingly prominent [8]. One of the most critical challenges is the escalating security threats that target IoT devices. The proliferation of various types of attacks poses a significant risk to the integrity and privacy of IoT systems [1]. Consequently, there is an urgent need to develop highly secure and efficient systems capable of safeguarding IoT environments from these threats.

In recent years, numerous IoT researchers have been working on designing and implementing effective malware classification systems. In [15], Ruchi and Ankit proposed a honeypot-based approach using machine learning techniques for malware detection. The approach utilises data generated by the IoT honeypot as a dataset for dynamic and effective machine learning model training. This approach shows promise in combating zero-day DDoS Attacks, which pose an open challenge in defending IoT against DDoS attacks. In [17], the authors propose a system that uses a convolution neural network to monitor and detect malicious behaviour in specific IoT nodes within a network. They used a simple CNN with three convolution layers, two max pooling and two dense layers. They tested the proposed system using different datasets and achieved an overall F1-score of 97.77%. In [3], a lightweight malware feature detector was designed and implemented for static and dynamic feature detection. The detector is capable of detecting low- and high-level features in labelled data and can be trained to avoid overfitting. To test its accuracy, the feature detector was trained and tested on byte-code-based datasets of opcodes using the embedding model of Chars2Vec. The detected features were then fed into two different architectures: a fully connected network (FCN) with softmax activation and a long-short-term memory neural network. They achieved 98% accuracy. In [16] Yuan and et al pro-

[1] The link will be available after the anonymity period

vide a method for IoT malware classification using lightweight convolution neural networks (LCNN), which reduces trainable parameters while maintaining high accuracy. The LCNN model is only 1MB, and the proposed method achieves high accuracy of over 95% on multiple IoT malware datasets and outperforms low-level feature-based methods. In [10] they proposed a method that uses both static and dynamic features, integrated with machine learning classifiers, to distinguish IoT botnets from benign samples. Their experiments show that their approach achieves an accuracy of over 99% on 6520 samples, with 4644 IoT botnet samples.

While the mentioned works have achieved a high accuracy rate of over 99%, this level of accuracy may still be considered inadequate for certain applications, especially in the context of IoT healthcare systems, where any error or change in the information transmitted or received can pose a high risk to the patient's life. Furthermore, many authors have focused on reducing the computational complexity of their models in order to decrease training time and prediction speed, but this can have a negative impact on the overall performance of the system. The aim of this work is to design a new model that is faster, more secure, and less complex for classifying malware using modern AI techniques. We will employ natural language processing and the chi-square method to extract the most relevant features from malware files and then use machine learning algorithms to classify these features.

Contributions of our work

The following are the main contributions of our work

1) We propose a novel method for classifying malware that combines natural language processing, Chi-square, and machine learning algorithms. Our approach offers a more accurate and efficient way to classify malware compared to existing methods.
2) To evaluate the effectiveness of our new method, we conduct a comparative analysis with several state-of-the-art machine learning techniques, including SVM, KNN, Gradient Boosting, Naive Bayes, and Random Forest.
3) Our new method significantly reduces the time required for classifying a task by selecting only the relevant features instead of processing entire files.

The remainder of this paper is structured as follows: In Section II, we provide a description of the dataset used in our study. Section III presents an overview of the natural language processing techniques and machine learning algorithms employed in our approach. In Section IV, we describe the proposed system architecture and its components. Next, in Section V, we present an evaluation of the performance of our proposed system. Finally, in Section VI, provide the discussion conclusion of the work.

II. Dataset

The dataset used in this work was collected by [12] and due to security concerns, it is not available online. To obtain access to the dataset, interested parties must request it directly from the authors at Yokohama National University. The dataset comprises two folders: the first folder contains text files of malware opcode instructions, while the second folder contains benignware text files. In total, the dataset contains 4,000 files, which were collected using the IoTPOT honeypot system. Notably, the dataset includes four types of DDoS attacks targeting Telnet-enabled IoT devices, affecting over nine CPU architectures.

III. Natural language processing and machine learning techniques

This work employs two artificial intelligence concepts: natural language processing for preparing the dataset files, and machine learning algorithms for the classification process. In the following sections, we provide a brief explanation of each of these concepts.

A. *Natural language processing*

Natural language processing, or NLP for short, is an interdisciplinary topic that combines computational linguistics with artificial intelligence [5]. Its main goal is to give robots the capacity to decipher, interpret, and produce human language. Machine learning, deep learning, and statistical analysis are only a few of the many approaches used in NLP. NLP is an effective technology that is used to analyse the opcode sequences found in malware code when classifying it. The main idea is to convert the opcode sequences into a format that machine learning algorithms can analyse. This work employs a variety of NLP methods, including tokenization and vectorization [18].

B. Machine learning algorithms

In this paper, we have utilised most of the machine learning algorithms for analysis and compared their performance with and without the feature selection method. The algorithms will be briefly explained as follows:

1) Random Forest: a popular ensemble learning algorithm that combines multiple decision trees to improve the accuracy and robustness of predictions. Random Forest randomly selects a subset of features and a subset of training examples for each decision tree. This randomness helps to reduce overfitting and improve generalisation [4].
2) Naive Bayes: a simple probabilistic algorithm that is often used in classification tasks. It is based on Bayes' theorem, which calculates the probability of a hypothesis being true given some observed evidence. Naive Bayes assumes that all features are independent of each other, which is why it is called "naive". It is easy to implement and can work well with small datasets [13].
3) KNN (K-Nearest Neighbors): a simple algorithm that assigns a class label to a data point based on the class labels of its k-nearest neighbours in the feature space. The value of k is a hyperparameter that needs to be tuned [19].
4) Gradient Boosting: a powerful ensemble learning algorithm that creates a sequence of weak prediction models (e.g., decision trees) and combines them to make a final prediction. Gradient Boosting works by iteratively fitting new models to the residuals of the previous model, gradually reducing the error [9].
5) Support Vector Machine (SVM) is a powerful and widely used machine learning algorithm for both classification and regression tasks. It works by finding the hyperplane that maximally separates the classes in the input data. SVM is particularly effective when working with high-dimensional and complex data, and it is often used in applications such as image classification, text classification, and bioinformatics [6].

IV. Proposed system architecture

The proposed system architecture involves multiple stages. The following explanation outlines each of these stages.

1) Pre-processing: Preprocessing is an important step in obtaining better performance [11], in this step, the dataset is first stripped of special characters such as !@#% . This is done for all files in the dataset. Following that, all the words in the dataset are converted to lowercase, and then all the stop words like "the", "and", "of", "to", etc., are removed.
2) Tokenization: The second step is to break down the text files of the dataset into smaller units called tokens. Tokenization helps in transforming unstructured text data into a structured format that can be readily processed.
3) Vectorization: After tokenization, the frequency of each word is counted to identify the most common and relevant terms within the document. Finally, a vectorization technique is applied to represent the frequency count of each word as a numerical value, which can be used as a feature vector in machine learning algorithms.
4) Chi-square: The chi-square test is often used as a feature selection method, where it is applied to determine the most important features in a dataset. This is done by calculating the chi-square statistic for each feature and selecting the top k features with the highest chi-square values. The chi-square statistic is calculated as follows
   a) Count the number of occurrences of each feature (word) in each class separately (i.e., malware and benign files). This will give you two frequency tables, one for each class. Let $N_{w,c}$ represent the count of feature w in class c.

b) Compute the total number of documents (text files) in each class, denoted as N_c. In this case, N_c is 4000 files.

c) Compute the total number of occurrences of each feature (word) in both classes combined, denoted as N_w.

d) For each feature (word), compute the expected frequency (EF) in each class by multiplying the total number of occurrences of that feature (word) in both classes by the proportion of documents (text files) in each class. The expected frequency in class c for feature w, denoted as $EF_{w,c}$, is calculated as:

$$EF_{w,c} = \frac{N_w \cdot N_c}{N_{\text{total}}} \tag{1}$$

where N_{total} represents the total number of documents across all classes.

e) For each feature (word), compute the observed frequency (OF) in each class. The observed frequency in class c for feature w, denoted as $OF_{w,c}$, is the same as the count $N_{w,c}$.

f) For each feature (word), compute the chi-square statistic using the equation:

$$\text{Chi-square}w, c = \frac{(OFw, c - EF_{w,c})^2}{EF_{w,c}} \tag{2}$$

This equation measures the difference between the observed and expected frequencies for each feature and class.

g) Sort the features based on their chi-square values and select the highest K features for further analysis.

5) Splitting the dataset: In order to train and evaluate our model, we divided the dataset into training and testing sets. We experimented with different percentages for the split and trained our model on various percentages of the data.

6) The final stage of our proposed system is to input the pre-processed data into the machine learning algorithm. In this study, five different machine learning methods were employed for classification purposes, namely SVM, KNN, Random Forest, Gradient Boosting, and Naive Bayes. Figure 1 show the proposed system architecture .

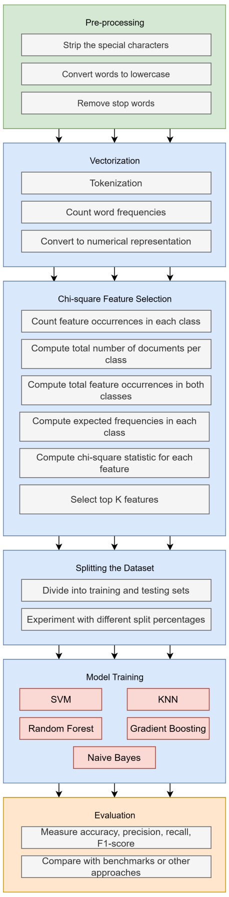
Pre-processing
Strip the special characters
Convert words to lowercase
Remove stop words
Vectorization
Tokenization
Count word frequencies
Convert to numerical representation
Chi-square Feature Selection
Count feature occurrences in each class
Compute total number of documents per class
Compute total feature occurrences in both classes
Compute expected frequencies in each class
Compute chi-square statistic for each feature
Select top K features
Splitting the Dataset
Divide into training and testing sets
Experiment with different split percentages
Model Training
SVM
KNN
Random Forest
Gradient Boosting
Naive Bayes
Evaluation
Measure accuracy, precision, recall, F1-score
Compare with benchmarks or other approaches

Fig. 1. Proposed System architecture

V. Performance Evaluation of the Proposed System

As explained in the previous section, the dataset undergoes preprocessing, and the chi-square of each feature is computed. The features with the highest chi-square values are selected as features for training the machine learning algorithm. The dataset is then fed to the machine learning algorithm with a specified value of K, which represents the number of features used in training the system. Each of the machine learning algorithms is trained and tested using different percentages of the dataset, including 20%-80%, 22.5%-77.5%, 25%-75%, 27.5%-72.5%, and 30%-70%. As shown the minimum testing is 20% used to prevent the overfitting problem. Each set of training and testing is evaluated with a different K, starting from 1 feature up to 1000. The hyperparameters that achieved the highest performance for each algorithm are recorded and illustrated in Table I. These hyperparameters include the number of features K used for training, as well as the percentage of the dataset used for training and testing.

To assess the effectiveness of the chi-square feature selection method, we compared the performance of machine learning algorithms with and without the feature selection applied. Specifically, we evaluated the recall, precision, F1-score, and accuracy of the models using both approaches. As shown in Table II Our results show that, across all performance metrics, the machine learning models that incorporated the chi-square feature selection method outperformed those that did not. This indicates that the chi-square method was effective in identifying the most relevant features for the classification task. Tables clearly show the superiority of the models with feature selection applied in terms of all performance metrics considered.

99.93

IoT devices are known for their resource constraints, requiring highly efficient models. In this study, we compare the efficiency of models with and without the application of feature selection with respect to time-consuming. The results clearly demonstrate that all the models performed significantly better when the feature selection method was applied compared to when it was not. Figure 2 illustrates the performance of each model. Among them, SVM showcased the best performance, while random forest and KNN performed comparatively less effectively with the two approaches applied.

Table III presents a comprehensive comparison between the proposed methods and recent works in malware detection. The results clearly demonstrate that the proposed method outperforms the existing approaches

VI. Discussion and consultation

In this paper, a machine learning system based on NLP is proposed for malware classification. The proposed system is evaluated using the IoT-POT dataset and compared without using NLP techniques. The findings show that applying NLP techniques, such as vectorization and chi-square feature selection, improves the performance of machine learning models in several ways. Firstly, by converting the data into numerical features through vectorization, the information becomes more efficiently understandable for machine learning algorithms. Secondly, the chi-square feature selection method focuses on selecting the most informative features, allowing the models to concentrate on the most discriminative aspects of the data. Additionally, the reduction of the number of features to a particular set rather than taking into account all features leads to faster processing, which reduces time consumption.

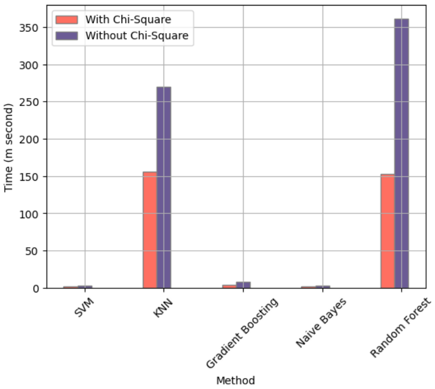

Fig. 2. Time Comparison of Machine Learning Methods with and without Chi-Square Feature Selection

References

[1] Shubair A Abdullah and Ahmed A Al Ashoor. Ipv6 security issues: A systematic review following prisma guidelines. *Baghdad Science Journal*, 19(6 (Suppl.)):1430–1430, 2022.

[2] Shadab Alam, Shams Tabrez Siddiqui, Ausaf Ahmad, Riaz Ahmad, and Mohammed Shuaib. Internet of things (iot) enabling technologies, requirements, and security challenges. In *Advances in Data and Information Sciences: Proceedings of ICDIS 2019*, pages 119–126. Springer, 2020.

[3] Muhammad Amin, Duri Shehwar, Abrar Ullah, Teresa Guarda, Tamleek Ali Tanveer, and Sajid Anwar. A deep learning system for health care iot and smartphone malware detection. *Neural Computing and Applications*, pages 1–12, 2020.

[4] Gérard Biau and Erwan Scornet. A random forest guided tour. *Test*, 25:197–227, 2016.

[5] Yue Kang, Zhao Cai, Chee-Wee Tan, Qian Huang, and Hefu Liu. Natural language processing (nlp) in management research: A literature review. *Journal of Management Analytics*, 7(2):139–172, 2020.

TABLE I
HYPERPARAMETERS FOR HIGHEST PERFORMING

Method	Training Set (%)	Testing Set (%)	No of K
SVM	75	25	23
KNN	77.5	22.5	68
Gradient Boosting	70	30	41
Naive Bayes	75	25	82
Random Forest	80	20	26

TABLE II
COMPARISON OF MACHINE LEARNING METHODS WITH AND WITHOUT CHI-SQUARE FEATURE SELECTION

	With Chi-square				Without Chi-square			
Method	Recall	Precision	F1-score	Accuracy	Recall	Precision	F1-score	Accuracy
SVM	99.933	99.93	99.93	99.93	98.30	97.24	97.76	98.69
KNN	99.37	97.60	98.45	99.21	99.37	97.60	98.45	98.99
Gradient Boosting	99.72	99.72	99.72	99.83	99.71	99.71	99.71	99.83
Naive Bayes	94.00	85.00	89.00	91.94	93.38	81.05	84.76	89.12
Random forest	99.89	99.88	99.89	99.89	99.89	99.89	99.88	99.89

TABLE III
COMPARISON OF METHODS FOR MALWARE CLASSIFICATION

Method	Authors	Dataset	F1-score (%)
CNN	Ahmad et al [17]	IoTPOT [12]	97.77
SVM with chi-saqaure	propsed method	IoTPOT [12],	99.93

[6] Duggani Keerthana, Vipin Venugopal, Malaya Kumar Nath, and Madhusudhan Mishra. Hybrid convolutional neural networks with svm classifier for classification of skin cancer. *Biomedical Engineering Advances*, 5:100069, 2023.

[7] Abhishek Khanna and Sanmeet Kaur. Internet of things (iot), applications and challenges: a comprehensive review. *Wireless Personal Communications*, 114:1687–1762, 2020.

[8] Bhabendu Kumar Mohanta, Debasish Jena, Utkalika Satapathy, and Srikanta Patnaik. Survey on iot security: Challenges and solution using machine learning, artificial intelligence and blockchain technology. *Internet of Things*, 11:100227, 2020.

[9] Alexey Natekin and Alois Knoll. Gradient boosting machines, a tutorial. *Frontiers in neurorobotics*, 7:21, 2013.

[10] Quoc-Dung Ngo, Huy-Trung Nguyen, Hoang-Anh Tran, and Doan-Hieu Nguyen. Iot botnet detection based on the integration of static and dynamic vector features. In *2020 IEEE Eighth (ICCE)*, pages 540–545. IEEE, 2021.

[11] Hadeel S Obaid, Saad Ahmed Dheyab, and Sana Sabah Sabry. The impact of data pre-processing techniques and dimensionality reduction on the accuracy of machine learning. In *2019 9th(iemecon)*, pages 279–283. IEEE, 2019.

[12] Yin Minn Pa Pa, Shogo Suzuki, Katsunari Yoshioka, Tsutomu Matsumoto, Takahiro Kasama, and Christian Rossow. Iotpot: Analysing the rise of iot compromises. *Emu*, 9(1), 2015.

[13] Korab Rrmoku, Besnik Selimi, and Lule Ahmedi. Application of trust in recommender systems—utilizing naive bayes classifier. *Computation*, 10(1):6, 2022.

[14] Sana Sabah Sabry, Noor Ahmed Qarabash, and Hadeel S Obaid. The road to the internet of things: a survey. In *2019 9th Annual Information Technology, Electromechanical Engineering and Microelectronics Conference (IEMECON)*, pages 290–296. IEEE, 2019.

[15] Ruchi Vishwakarma and Ankit Kumar Jain. A honeypot with machine learning based detection framework for defending iot based botnet ddos attacks. In *2019 3rd (ICOEI)*, pages 1019–1024. IEEE, 2019.

[16] Baoguo Yuan, Junfeng Wang, Peng Wu, and Xianguo Qing. Iot malware classification based on lightweight convolutional neural networks. *IEEE Internet of Things Journal*, 9(5):3770–3783, 2021.

[17] Ahmad MN Zaza, Suleiman K Kharroub, and Khalid Abualsaud. Lightweight iot malware detection solution using cnn classification. In *2020 IEEE 3rd 5G World Forum (5GWF)*, pages 212–217. IEEE, 2020.

[18] Yujia Zhai, Wei Song, Xianjun Liu, Lizhen Liu, and Xinlei Zhao. A chi-square statistics based feature selection method in text classification. In *2018 IEEE 9th (ICSESS)*, pages 160–163. IEEE, 2018.

[19] Shichao Zhang, Xuelong Li, Ming Zong, Xiaofeng Zhu, and Debo Cheng. Learning k for knn classification. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 8(3):1–19, 2017.