Information Gain Score Computation for N-Grams Using Multiprocessing Model

Shiva Darshan S.L.[1], Ajay Kumara M.A.[2], and Jaidhar C.D.[3]
Department of Information Technology
National Institute of Technology Karnataka
Surathkal, Mangalore, India
it15f02.shivadarshan@nitk.edu.in[1], ajayit13f01@nitk.edu.in[2], jaidharcd@nitk.edu.in[3]

***Abstract*—Currently, the Internet faces serious threat from malwares, and its propagation may cause great havoc on computers and network security solutions. Several existing anti-malware defensive solutions detect known malware accurately. However, they fail to recognize unseen malware, since most of them rely on signature-based techniques, which are easily evadable using obfuscation or polymorphism technique. Therefore, there is immediate requirement of new techniques that can detect and classify the new malwares. In this context, heuristic analysis is found to be promising, since it is capable of detecting unknown malwares and new variants of current malwares. The N-Gram extraction technique is one such heuristic method commonly used in malware detection. Previous works have witnessed that shorter length N-Grams are easier to extract. In order to identify and remove noisy N-Grams, a popular Feature Selection Technique (FST), namely, Information Gain (IG), which computes score for each N-Gram (feature) in the dataset has been used in this work. N-Grams with the highest IG score are considered as best features, while the remaining N-Grams are neglected. The IG-FST (Information Gain-Feature Selection Technique) is computational resource demanding and takes time to generate IG scores for larger N-Gram datasets, if the processing is to be accomplished in the sequential mode. To address this issue, the present work presents a multiprocessing model that computes IG scores rapidly for larger N-Gram datasets. The proposed model has been designed, implemented, and compared with the sequential mode of IG score computation. The experimental results demonstrate that the proposed multiprocessing model performance is 80% faster than the sequential model of IG score computation.**

***Keywords*—Malware Detection, Machine Learning, N-Grams, Information Gain.**

I. Introduction

Malware is one of the biggest threats to modern computers and cyber infrastructures, as its severity is increasing year by year and infecting computer systems or computer networks, which are susceptible. Generally, the detection of an unknown malware is difficult since many of the modern malwares are built with obfuscation characteristics, which can elude many standard signature-based anti-malware systems. Due to this, malware analysts fail to identify unknown malwares. To assist them, new malware detection methods are being invented by computer researchers [1].

The malware recognition strategy is broadly categorized into two types: signature-based approach and behavior-based approach. The signature-based approach employs a pattern matching technique to match the sequence of binary instructions with the regular expression. However, it can be easily evaded by the obfuscations technique, if the computer system is highly vulnerable [2]. On the other hand, the behavior-based approach utilizes machine learning techniques to detect malwares and is capable of detecting both static and dynamic behaviors of the malware, even if the code is obfuscated. Thus, it is more widely adopted than the signature-based approach [3]. Nevertheless, it fails to balance between false positive rate and malware detection rate [4].

Soon after the idea of heuristic-based method was used to detect malwares through N-Grams generated from a given set of benign and malware files. The N-Grams were used as a basic feature by the classifier to distinguish between benign and malware files [5]. However, all N-Grams cannot be used as prominent features to prepare a classifier understandable training or testing file. This is due to the fact that the malware detection rate of the classifier purely depends on the final features supplied in the training file. FST plays an important role in generating a score for each N-Gram, which helps to select the best features based on the score. FST takes an enormous amount of time to produce a score, when the total numbers of N-Grams are more in the original feature set.

To address the issue of computing the score quickly, we present in this paper, a multiprocessing model that computes an IG score for each N-Gram. The proposed multiprocessing model is capable of computing the IG score of any large N-Gram dataset of any length of N-Grams, for example, N=2, N=3, N=4, etc. The proposed multiprocessing model is implemented, validated, and compared with the sequential model of IG score computation.

The rest of the paper is structured as follows: In Section II, we study the background of N-Gram techniques used to detect the malware, and review earlier research works related to malicious executable file detection. In Section III, we describe the proposed multiprocessing model. In section IV, experimental results are discussed. Finally, in Section V, we arrive at the conclusion.

II. Background and Related Work

Malware analysis is the process of determining the presence of malicious substances in the testing input file. However, the timely discovery of new malware is still a critical issue.

Several attempts were made with different techniques for malware detection [6]. Currently, the analysis of malware is performed in two ways: static analysis and dynamic analysis. In static analysis, the malware is analyzed without executing the malware. Static features such as Application Programming Interface (API) calls, system calls, etc. are used to identify the malware [7] [8]. These extracted features are used as basic parameters by most of the anti-malware defensive solutions to spot the malware. Static analysis is largely ineffective and can be easily evaded by sophisticated malware, and sometimes, it can miss some important behavior parameters of the malware [9].

Dynamic malware analysis overcomes the limitations of static malware analysis. It is also called as behavior analysis. Dynamic malware analysis is normally performed in a sandbox environment to avoid propagation of malware infection in the productive system. There have been several dynamic malware sandbox approaches proposed in literature that perform dynamic malware analysis using the sandbox technology [10] [11] [12].

The common N-Gram analysis was one of the most successful methods in case of text classification and text clustering [5]. Inspired by the common N-Gram analysis method, the N-Grams in the present context are used in digital forensics for malware detection since it captures implicit features [16]. Byte-level N-Gram is found to be significant to extract relevant features for malware classification, due to its robust functionality. However, N-Gram underperforms from high feature space and large computation time because of many generated features, which cannot be accommodated in the main memory. So, to advance the feature selection process, a new partition-based method was introduced [13]. This method efficiently converts the original dataset to separate partitions and enables parallelism on different partitions. Consequently, an attempt was made to improve the feature selection process by using classifier called Elastic Net regularized Logistic Regression [14]. This approach significantly addresses most issues on N-Grams for feature selection process, i.e., N-Grams are computationally expensive and manifest low performance with more data.

In order to detect malware dynamically, many of the recent techniques utilize N-Grams as the basic feature [15]. The idea of N-Gram was stimulated by a commonly known method called word N-Gram. Word N-Gram is most widely used in applications involving natural language processing, speech recognition, text mining, and information retrieval. Subsequently, a novel approach was put forth for effective detection of malware based on invoked system call sequence [10]. The collected system call sequence structured in the form of N-Grams and N-Gram feature extraction technique is widely used for different input sources [5] [15] [17]. The N-Grams were selected from malware and benign files. The N-Grams based approach outperforms when the experiment is carried with a larger feature set. Recent reports [5] have shown that IG-FST has produced best results in classifying malicious executables files and benign executable files correctly.

In our proposed work, we have implemented a multiprocessing model for computing the IG score for N-Grams of larger datasets. The computation time taken by the multiprocessing model and the sequential mode1 was measured to compare the processing efficiency. The proposed approach proves to be generic to significantly reduce the computation time involved, while recommending the unique feature based feature selection technique. We believe that the current evaluation of this proposed approach validated against IG-FST will also work for other FSTs to measure the score of unique feature computation as required by a classifier.

III. PROPOSED WORK

In this work, the proposed multiprocessing model is designed, implemented, and validated with a larger N-Gram dataset. Benign N-Gram Files (BNFs) and Malware N-Gram Files (MNFs) were both included in the N-Gram dataset, where each N-gram file consists of more number of N-Grams. For each N-Gram, the IG score was computed using the steps depicted in Fig. 2. The IG score generation steps are grouped into four phases: 1) Pre-processing Phase, 2) Chunks Construction Phase, 3) Chunks to Process Allocation Phase, and 4) IG Score Computation Phase

A. *Pre-processing Phase*

In this phase, the N-grams were generated by extracting the binary instructions from both the benign and malware executable files. This step is referred as the intermediate stage. In this stage, the extracted binary instructions are continuously placed in a row. Later, N-Grams of different feature lengths are generated such as NG_L = 2 bytes, NG_L = 3 bytes, NG_L = 4 bytes, etc. based on the length specified by the user to create BNFs and MNFs. Subsequently, in order to get the highest order sequence and better selection of N-Grams, the created BNF and MNF were arranged in non-increasing order by removing any duplicates, if found. The generation of BNF and MNF are illustrated in Fig. 1.

The generation of BNF and MNF is a very important step since it is the start-up step or is defined as the input for the proposed multiprocessing model. The next step involves the union operation on the BNFs (B1, B2, B3 . . . Bn) and MNFs (M1, M2, M3 . . . Mn) individually on both the categories, benign and malware, to identify and remove redundant N-Grams. After the union operation, Unique Benign N-Gram File (UBNF) and Unique Malware N-Gram File (UMNF) are obtained as shown in Fig. 2.

B. *Chunks Construction Phase*

The UBNF and UMNF consist of a large number of N-Grams, and computing the IG score for each N-Gram in the UBNF and UMNF sequentially is computational resource demanding. To reduce this, a multiprocessing model is proposed in this work that computes the IG score for each N-Gram. In this phase, the UBNF is divided into files of smaller benign N-Gram files [Benign N-Gram Chunk (BNC)], namely, BNC1, BNC2, BNC3, BNC4. . . BNCn, and similarly, the UMNF is

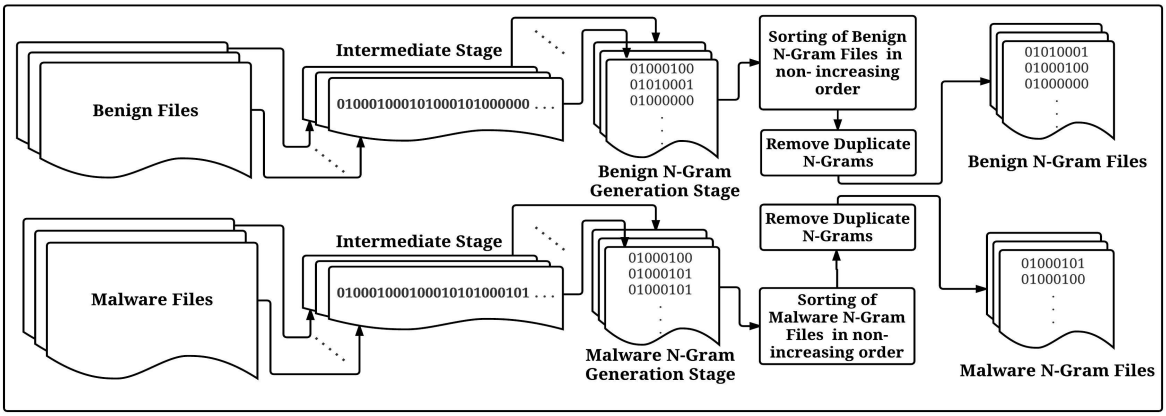


Fig. 1: Generation of Benign N-Gram Files and Malware N-Gram Files

divided into smaller malware N Gram files [Malware N-Gram Chunk (MNC)], namely, MNC1, MNC2, MNC3, MNC4. . . MNCn as shown in Fig. 2. A pair of chunks (BNC1, MNC1), (BNC2, MNC2) . . . (BNCn, MNCn) is assigned to a created process. Each benign chunk N-Gram file and malware chunk N-Gram file consists of a fixed number of N-Grams. The number of N-Grams to be present in each BNC file and MNC file is to be of a constant value as specified by the user, and accordingly, the generation of chunk files takes place.

C. Chunks to Process Allocation Phase

In this phase, one chunk file from both benign and malware categories is accessed sequentially and assigned to the process. The distribution of the number of chunk pairs to each process is calculated as per the equation given below:

$$\begin{aligned} &Distribution\ of\ chunk\ pair\ to\ each\ process = \\ &\frac{Highest\ number\ of\ chunks\ made\ either\ from\ benign\ or\ malware\ category}{No.\ of\ process} \end{aligned} \tag{1}$$

D. IG Computation Phase

To compute the IG score, each process follows the following steps:

- The union operation is conducted on each pair of benign and malware chunk N-Gram files to recognize the occurrence of the same N-Gram.
- If observed, similar occurrences of N-Gram are removed after performing the union operation.
- Finally, a unique N-Gram file is prepared.

Each N-Gram from the unique N-Gram file is accessed to check its existence in all BNFs and MNFs to compute the IG score using the following equation:

$$IG(N-Gram) = \sum_{v_{N-Gram}\in\{0,1\}} \sum_{C\in\{C_i\}} P(v_{N-Gram}, C) \log \frac{P(v_{N-Gram}, C)}{P(v_{N-Gram}), P(C)} \tag{2}$$

Where, C is one of the two categories - benign or malware, v_{N-Gram} is the value of the N-Gram; v_{N-Gram} = 1 indicates the presence of N-Gram in either of the N-Gram files and v_{N-Gram} = 0, otherwise; P(v_{N-Gram}, C) is the proportion of N-Gram files in C in which case, the N-Gram takes on the value v_{N-Gram}; P(v_{N-Gram}) is the proportion of benign or malware N-Gram files in the entire training set such that N-Gram takes the value v_{N-Gram}; and P(C) is the proportion of datasets belonging to category C. The N-Grams are organized in non-increasing order based on the IG score and the topmost L numbers of N-Grams are extracted as best features for the purpose of classification.

IV. EXPERIMENT RESULTS

The experimental datasets consists of two types: benign dataset and malware dataset. The benign dataset is a collection of non-malicious executable files, whereas the malware dataset is a collection of malicious executable files obtained from public source [1]. For the purpose of the experiments, ten benign files and ten malware files were considered. The length of the N-Grams was fixed for four bytes to generate the BNFs and MNFs. Three processes were created to demonstrate the multiprocessing model to compute the IG score.

Furthermore, as per the explanation in section III, the generated BNFs and MNFs underwent union operation, and then, the N-Grams were sorted in non-increasing order and duplicate N-Grams were removed, if observed to obtain UBNF and UMNF. The generated UBNF and UMNF in this experiment consists of 414606 N-Grams of the benign category and 2475931 N-Grams of the malware category.

[1] https://malwr.com/

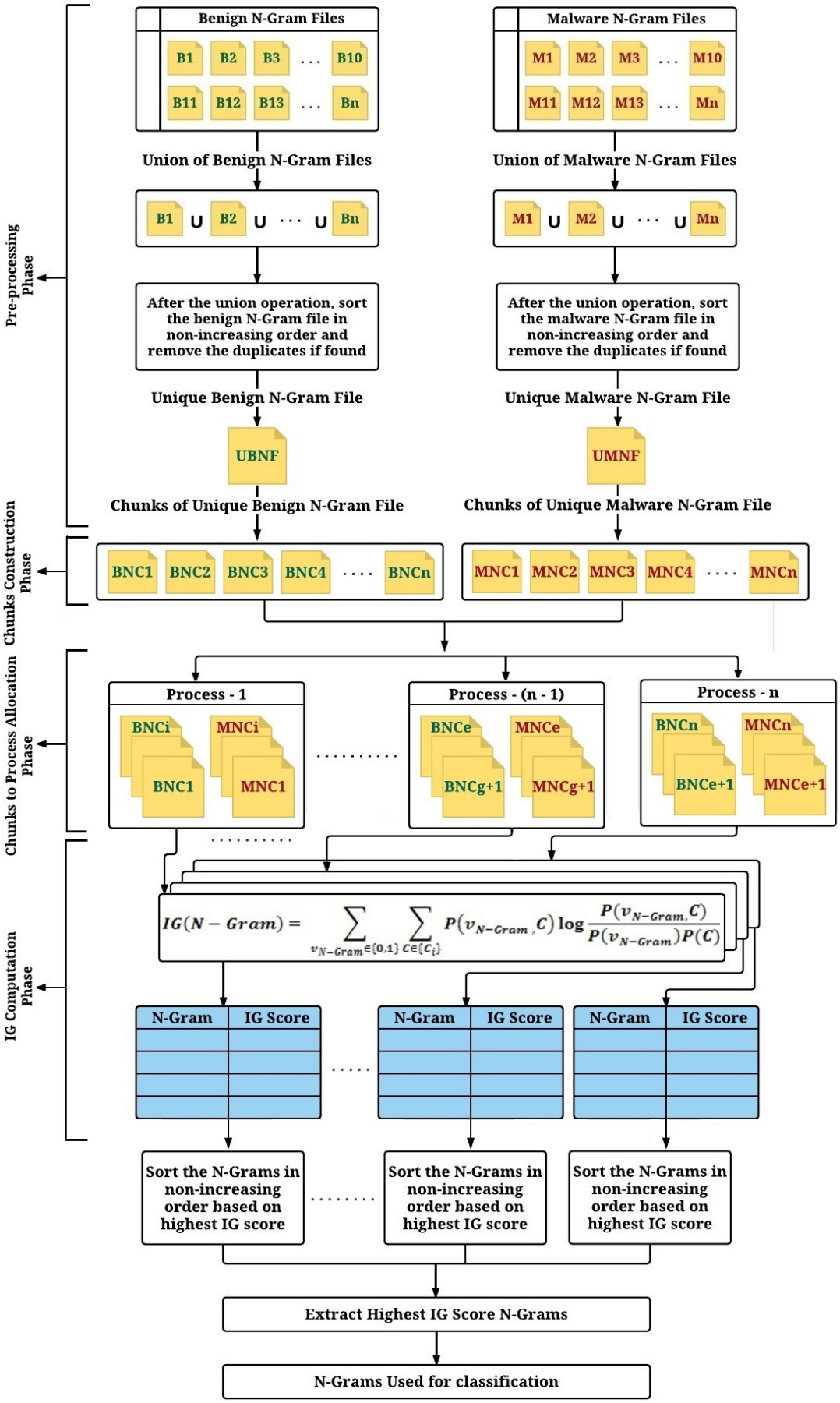


Fig. 2: Multiprocessing model to compute Information Gain score

TABLE I: Information Gain score computation time

Benign N-Gram Files	Malware N-Gram Files	Unique N-Grams File		Threshold Value for Chunk Construction		No. of Chunk Files Constructed		N-Gram Size (No. of N-Grams x No. of Chunks)		Time Taken for IG Score Computation in Sequential Mode1 (Sec)	Time Taken for IG Score Computation in Multiprocessing Mode1 (Sec)
		Benign	Malware	Benign	Malware	Benign	Malware	Benign	Malware		
10	10	414606	2475931	10000	10000	42	248	10000 x 3	10000 x 3	5244.857	1065.764
10	10	414606	2475931	15000	15000	28	166	15000 x 3	15000 x 3	7989.225	1513.592
10	10	414606	2475931	20000	20000	21	124	20000 x 3	20000 x 3	11006.653	2081.264
10	10	414606	2475931	25000	25000	17	100	25000 x 3	25000 x 3	13722.352	2684.321
10	10	414606	2475931	30000	30000	14	83	30000 x 3	30000 x 3	16688.499	3364.588

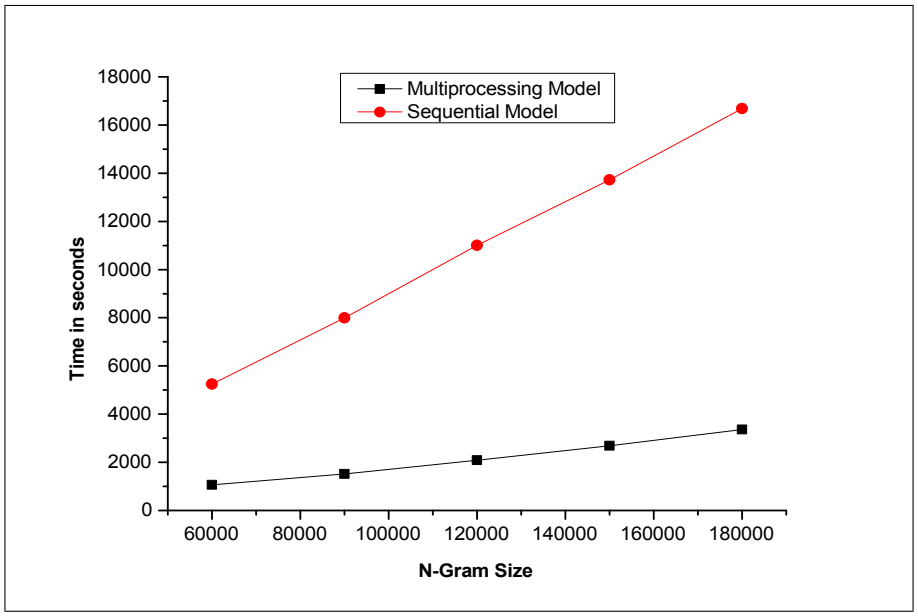
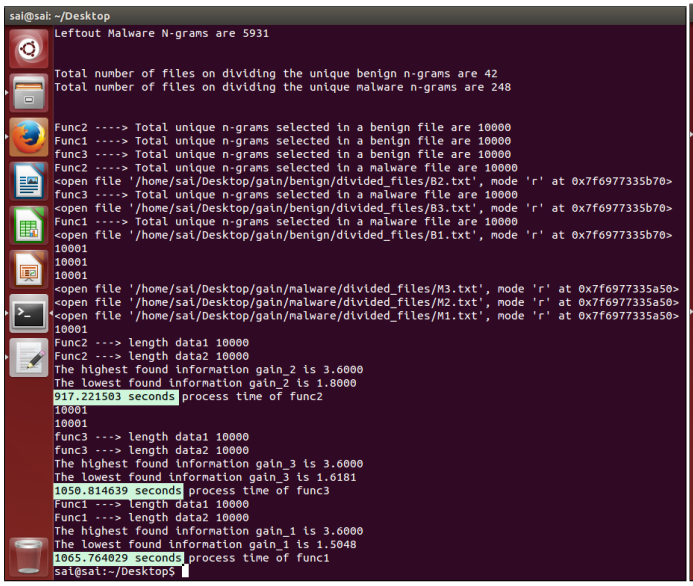


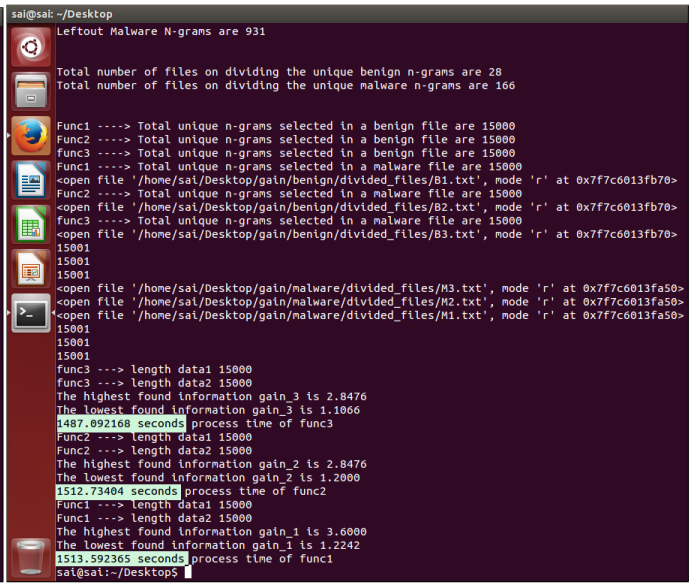
Fig. 3: Comparison between multiprocessing model and sequential model to compute Information Gain score

The proposed multiprocessing model was implemented using the Python programing language. It constructs the chunks separately from the UNBF and UNMF based on the threshold value set by the user. Then, the IG score for each N-Gram is computed. In this work, N-grams chunks with varying threshold values such as 10000, 15000, 20000, 25000, and 30000 were created to demonstrate the efficiency of the proposed model. For the threshold value 10000, 42 benign chunk files and 248 malware chunk files were obtained. For other threshold values like 15000, 20000, 25000, and 30000, benign chunks files of 28, 21, 17, and 14 and malware chunk files of 166, 124, 100, and 83 were constructed and tabulated in Table I. To demonstrate the efficiency of the multiprocessing model against the sequential model, the first three pair of chunk files from both categories of benign and malware files were obtained for different threshold values such as 10000, 15000, 20000, 25000, and 30000.

When the threshold value was set to 10000, each chunk file from both benign and malware categories consisted of a fixed number of 10000 N-Grams. If three benign chunk files and three malware chunk files were considered, then the number of N-Grams in the benign category was 30000, and similarly, the N-Grams size of the malware category was also 30000. The time taken to compute the IG score of all 60000 N-Grams was 5244.857 seconds in the sequential model and 1065.764 seconds in the multiprocessing model. In the same way, when the threshold value was set to 15000, each chunk file from both benign and malware categories consisted of a fixed number of 15000 N-Grams. For the first three chunk files, the N-Gram size of both benign and malware was 45000 and the computational time observed to compute the IG score for all 90000 N-Grams by the sequential model was 7989.225 seconds and for the multiprocessing model was 1513.592 seconds. Similarly, on increasing the threshold value to 20000, each benign chunk file and malware chunk file now consisted of a fixed number of 20000 N-Grams. For three benign chunk files and three malware chunk files, the N-Gram size increased to 120000 and the time taken to compute the IG score for all 120000 N-Grams was 11006.653 seconds by the sequential model and 2081.264 seconds by the multiprocessing model. Correspondingly, the computational time taken by the sequential model and the multiprocessing model to compute

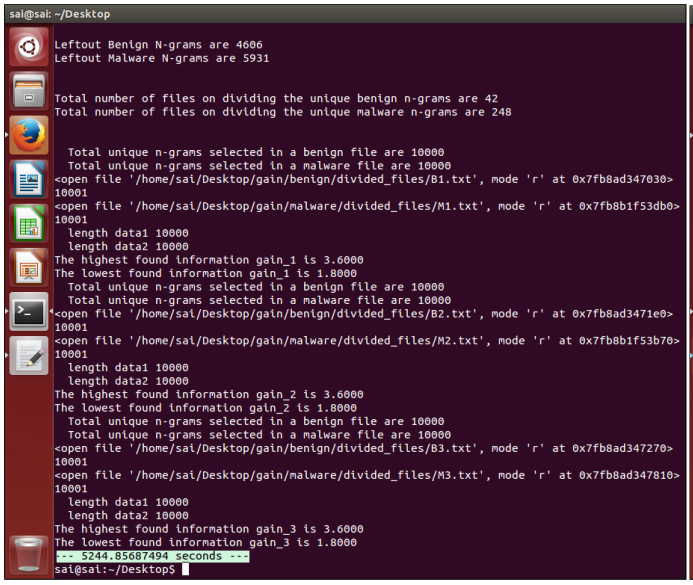


(a) Computation Time(sec) - Chunk of 60000 N-Grams.

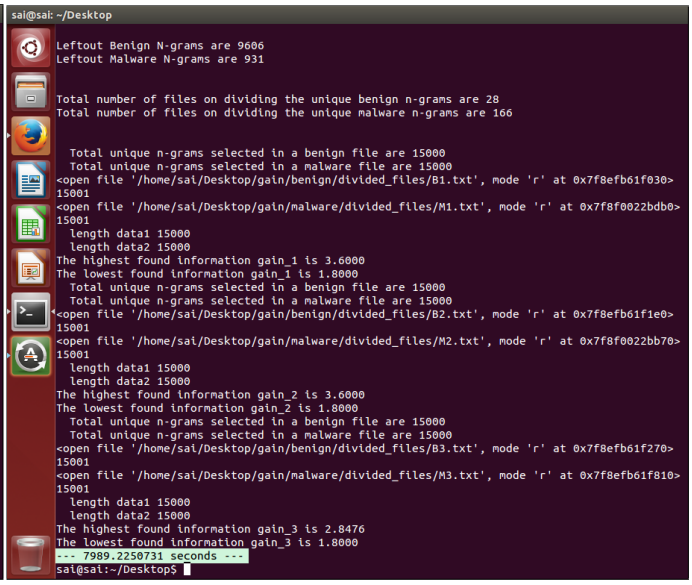


(b) Computation Time(sec) - Chunk of 90000 N-Grams.

Fig. 4: Information Gain score computation time - Multiprocessing model.



(a) Computation Time(sec) - Chunk of 60000 N-Grams.



(b) Computation Time(sec) - Chunk of 90000 N-Grams.

Fig. 5: Information Gain score computation time - Sequential model.

the IG score for 150000 N-Grams and 180000 N-Grams is tabulated in Table I.

The computation time taken to compute the IG score by the multiprocessing model for chunks of 60000 N-Grams and 90000 N-Grams is shown in Fig. 4a and 4b, where the highlighted part in Fig. 4a and 4b denotes the computation time taken by the process (FUNC1, FUNC2, and FUNC3 denotes Process1, Process2, and Process3, respectively) to compute the IG score. Similarly, experiments were carried out for other chunks such as 120000 N-Grams, 150000 N-Grams, and 180000 N-Grams. Fig. 5a and Fig. 5b represent the computational time taken to compute the IG score by the sequential model for chunks of 60000 N-Grams and 90000 N-Grams. The highlighted part signifies the overall computation time required by the sequential model to generate the IG score. Similar demonstrations were performed for chunks of 120000 N-Grams, 150000 N-Grams, and 180000 N-Grams in the sequential model. For each N-Gram, the computation time needed to generate the IG score by the sequential model as well as by the multiprocessing model was observed and is

depicted in Table I.

Looking at Table I, we can notice that there is a drastic improvement in the computation time of the IG score by the proposed multiprocessing model. For the chunk of 60000 N-Grams, the computation time of the IG score reduced by 79.68%, and chunks of 90000, 120000, 150000, and 180000 N-Grams, the computational time was reduced by 81.05%, 81.09%, 80.44%, and 79.84%, respectively, against the sequential model. On average, the proposed approach was 80% faster than the sequential model for the computation of the IG score. As observed in Fig. 3, analogizing the two parameters manifests that our multiprocessing model is faster than the sequential model.

V. Conclusion

Many researchers incorporated N-Grams with the machine learning technique to classify whether the given instance (input file) belongs to the benign or malware class. FST is an important step that determines which features are best and which features are noisy. The IG- FST technique consumes more time in producing the IG score for each N-Gram, when the N-Gram dataset size is large. To tackle this issue, in this work, we have proposed, implemented, and validated a multiprocessing model that generates IG score rapidly for larger N-Gram datasets. Experiments were conducted with different sizes of N-Grams datasets such as 60000, 90000, 120000, 150000, and 180000. In each experiment, the performance of the proposed multiprocessing model in computing the IG score is high as compared to the sequential model. On average, the proposed approach is 80% faster than the sequential model of IG score computation.

References

[1] X. Jiang, X. Wang, and D. Xu, "Stealthy malware detection through vmm-based out-of-the-box semantic view reconstruction," in *Proceedings of the 14th ACM conference on Computer and communications security*. ACM, 2007, pp. 128–138.

[2] A. Moser, C. Kruegel, and E. Kirda, "Limits of static analysis for malware detection," in *Computer security applications conference, 2007. ACSAC 2007. Twenty-third annual*. IEEE, 2007, pp. 421–430.

[3] J. Z. Kolter and M. A. Maloof, "Learning to detect and classify malicious executables in the wild," *Journal of Machine Learning Research*, vol. 7, no. Dec, pp. 2721–2744, 2006.

[4] M. G. Schultz, E. Eskin, F. Zadok, and S. J. Stolfo, "Data mining methods for detection of new malicious executables," in *Security and Privacy, 2001. S&P 2001. Proceedings. 2001 IEEE Symposium on*. IEEE, 2001, pp. 38–49.

[5] D. K. S. Reddy and A. K. Pujari, "N-gram analysis for computer virus detection," *Journal in Computer Virology*, vol. 2, no. 3, pp. 231–239, 2006.

[6] Z. Bazrafshan, H. Hashemi, S. M. H. Fard, and A. Hamzeh, "A survey on heuristic malware detection techniques," in *Information and Knowledge Technology (IKT), 2013 5th Conference on*. IEEE, 2013, pp. 113–120.

[7] M. Christodorescu and S. Jha, "Static analysis of executables to detect malicious patterns," DTIC Document, Tech. Rep., 2006.

[8] M. Alazab, R. Layton, S. Venkataraman, and P. Watters, "Malware detection based on structural and behavioural features of api calls," 2010.

[9] P. OKane, S. Sezer, and K. McLaughlin, "Obfuscation: the hidden malware," *IEEE Security & Privacy*, vol. 9, no. 5, pp. 41–47, 2011.

[10] K. Rieck, P. Trinius, C. Willems, and T. Holz, "Automatic analysis of malware behavior using machine learning," *Journal of Computer Security*, vol. 19, no. 4, pp. 639–668, 2011.

[11] H. Bai, C.-z. Hu, X.-c. Jing, N. Li, and X.-y. Wang, "Approach for malware identification using dynamic behaviour and outcome triggering," *IET Information Security*, vol. 8, no. 2, pp. 140–151, 2014.

[12] J. T. Juwono, C. Lim, and A. Erwin, "A comparative study of behavior analysis sandboxes in malware detection," in *International Conference on New Media (CONMEDIA)*, 2015, p. 73.

[13] W. Hu and Y. Tan, "Partitioning based n-gram feature selection for malware classification," in *International Conference on Data Mining and Big Data*. Springer, 2016, pp. 187–195.

[14] E. Raff, R. Zak, R. Cox, J. Sylvester, P. Yacci, R. Ward, A. Tracy, M. McLean, and C. Nicholas, "An investigation of byte n-gram features for malware classification," *Journal of Computer Virology and Hacking Techniques*, pp. 1–20, 2016.

[15] S. Jain and Y. K. Meena, "Byte level n–gram analysis for malware detection," in *Computer Networks and Intelligent Computing*. Springer, 2011, pp. 51–59.

[16] H. Yin, D. Song, M. Egele, C. Kruegel, and E. Kirda, "Panorama: capturing system-wide information flow for malware detection and analysis," in *Proceedings of the 14th ACM conference on Computer and communications security*. ACM, 2007, pp. 116–127.

[17] H. Parvin, B. Minaei, H. Karshenas, and A. Beigi, "A new n-gram feature extraction-selection method for malicious code," in *International Conference on Adaptive and Natural Computing Algorithms*. Springer, 2011, pp. 98–107.