Malware Materials Detection by Clustering the Sequence using Hidden Markov Model

Muhammed Mofe N AL Rwajah[a] and Ravi Rastogi[b]
[a] Master Student, Faculty of Computing and Information Technology, University of Bisha, Bisha, Saudi Arabia.
[b]Faculty of Computing and Information Technology, University of Bisha, Bisha, Saudi Arabia.

Article History: Received: 10 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 28 April 2021

Abstract: The exponential development in (malware)malicious development in current times and the growing scenario of challenges that malware poses to network atmospheres, like as the network and smart networks, highlight the need for further investigation in data technology and privacy forensics on computer network protection.The approach described in this scenario curve distinguishes "types" of malware groups that complex is extending, more obfuscated and more varied in nature. We suggest a hybrid strategy integrating signature recognition elements with machine learning-based approaches for classifying families of malware.The approach is performed using PHMM of the behavior features of the species of malware. This research article illustrates the method of modelling and learning developed PHMM using sequences derived from the discovery of the paramount aspects of each malware family, and the recognized orders produced during the development of Multiple Sequences Alignment (MSA). Since all file sets are not dangerous, and the aim would be to separate their malicious parts from the genuine ones along with put greater focus on them to raise the possibility of malware finding by ensuring the least case effect on the genuine parts. Founded on the "consensus series," the investigational findings indicate that even though minimal training data are usable, our proposed method outperforms other HMM-related strategies.

Keywords: Polymorphic Malware Materials, Metamorphic, Malware Detection, Static Analysis, Cyber Security, Profile Hidden Markov Models.

1. Introduction

At present, we seeing a massive rise in the use of technology in day-to-day life, further that network and Internet plays a significant role in regular activity of users by offering vital resources. Dramatic developments in information technologies and e-commerce have created exciting possibilities for corporate environments, but also raised questions of protection and secrecy of data in virtual worlds. Author [5] used the term "digital virus" at 1983, a vast rate of novel malware encryptions have appeared each year. The system virus refers to all like rootkits, worms, Trojans, spyware, adware, and ransomware; this malware is unauthorized computer programs [1-5]. Detection of ransomware is the very first step in protecting information networks. An antivirus program is really the standard way of dealing with viruses, but this software often uses signature-based identification methods and unable to distinguish new viruses and the null-day attacks.Malware tactics now are published more insidiously and deeper than it has ever been. Compared with the old generations of malware, certain types of malware are harder to identify; new versions of viruses (i.e., symmetric and polymorphic) using authentication and misdirection techniques [6].They generate various strains about themselves while maintaining the underlying features the same to build a completely new version, which after replication barely resembles the original version.In general, two key steps are malware identification techniques. The first is the study of vulnerabilities and retrieval of features and the second is the process of detection. The two traditional techniques, static and dynamic analyze, and modern hybrid approaches are defined in Phase II for the first step.In the second stage, the classification strategies are classified as statistical-based identification, way of behave based identification, and signature-based identification. The authors are dealt with in phase 3.This research was in the security sector, where we concentrated on malware identification. Provided that duration is one aspect of the operation of processes, the generating processes are thus a sequence of results. We may also use method designs and classification methods to work with such a series.The Hidden Marko Models and particularly the Profile-HMM are presented as effective tools in model development, classification, and its issues related. These have several applications in computer engineering, including nearly all problems in this area that are organized by occurrences or have a time-based function.Recognition of voice processing and voice, natural hand written and language processing, identification of data breach-attacks and network infiltration, analysis of diagnostic tool or prediction of associated proteins, identification and avoidance of fraud or retrieval of consumer activity trends in banking dealings, and estimating of money markets and asset values are some cases that can be listed.The HMM methods have already been emphasized and have played a crucial part in the correct solutions to these problems. With all that in mind, updating and enhancing the HMM-based methods, resulting in improved performance and increasing identification & accuracy rate of classification, would have a big effect on certain apps and glyph problems that have been briefly described earlier. Relevant HMM designs are capable of restricting the sophistication of the HMM training process, thereby reducing time. In addition, a simple structure means that the relevant model does have a few variables and can therefore be trained on small quantities of data. These HMM designs, whether they eloquently measure the experimental setup, will lead to suitable methods.One example of these architectures is Profile HMM, which is suitable as a pattern

description for fascinating sequence locations. For two purposes, the profile HMM structure is suitable as a depiction of the motif. Firstly, the form is linear to complement that of a role and secondly, it aims to restore the evolutionary process.The base of an HMM model is a series of states, so-called "play stages," which depicts the standard series for the considered kin. Both of the match conditions match a single location in the canonical series. A location weight matrix is the same as the sequence of states since each stage has a confidence interval over the entire signs. HMM, Profile also provides two different types of structures (plugin and remove states) that model the mechanism of evolution. The delete stage is opposite its accompanying match stage, which helps to avoid the match level. Connect states of self-loops were juxtaposed among match levels, enabling the addition of one or more bases within two match levels.We build a mathematical model initially implemented by [1] to detect blurred malware and suggest a new technique for profiling consider main data attributes. The aim of this article is to develop a Cutting-edge profile model hidden Markov (CPHMM) which enhances the accuracy of the article approach. In addition, we develop a model that implements mathematical testing methods for evaluating virus behave using minimal data and adjust it to manage 0 or 1 functions during retaining high identification rates of illegal purposes.The balance of this work is structured the following way. Related review work to it in Section 2. Section 3 briefly describes a summary of malware identification strategies. The suggested solution and the experimental findings are listed in phase 4.In phase 5 we describe our tests as well as the datasets used. The article ends with phase 6 where we explore prospective future system.

2. Existing Works

As stated previously in phase 1, approaches for virus classification could be categorized hooked on two groups of methods for staticanalys is and dynamic analysis of malware. The static analysis is extra common as it evaluates malicious code arrangement and identifies virus before consecutively it, as compared to dynamic analysis that generally involves emulation and malicious code assessment in rendering. In malware analysis established three general approaches, namely static behaviors, dynamic behaviors, and hybrid behaviors [7]. Some methods of malware identification are founded on the static analysis described in [8-18] and focus solely on the features data derived from virus or unknown source files before execution [3-6]. In the other hand, for malware detection, the hierarchical approach mentioned in [5, 19-27], in which runtime control occurs, was successfully utilized. Hybrid methods have been explored recently in [7, 28-30], everywhere static characteristics and dynamic characteristics have been utilized for exact recognition. Earlier research is defined in the following parts depending as to whether they depend on static analysis, dynamic analysis further more static-dynamic hybrid analysis research.

A. Static Validation

Malware is statically evaluated without the use of code execution. The set of operating codes and control flow diagrams are some valuable knowledge that we can obtain through static analysis. Their techniques, called G3MD, perform graph extraction to the metamorphosed malware category opcode graphs to retrieve the regular subsections. A classifier [31] is taught to differentiate between the neutral and harmful documents using these sub-graphs.In this work [4] a malware identification strategy was proposed that relies on static analysis and flow charts of operation. The method focuses on understanding subterfuge trends of reasonable precision in malicious programs and their effects. PCA (Principal Component Analysis) [15] has been used for malware detection, and SVM (Support Vector Machines) [3] was used. A sequence mining method to explore harmful frequent data as well as the Nearest Neighbor Classification model which is designed based on the orgin for virus identification discovered [32]. Decided to hire clustering technique based on the functions of malware detection invoke from the static analysis [8]. Function call of graphs to identify malicious software when evaluating opcode-based semantic similarity that depends on cryptanalysis methods for substitution[16, 17]. API call and functional code sequences were used to determine any particular malicious programs if a section of the code are similar [18].A KAMAS (knowledge-aided visual malware analysis system) designed and built for BBMA [33]. For a subsequence of features, the essential functions APIrequestsshould be intended to obtain private information and to use device resources are named. In [34] created anget-together code replacement matching platform named ScalClone which aims to recognize target malware code clones.

B. Dynamic Analysis

The samples input system is implemented in complex analytics. Typically it is done in an isolated framework like Virtual Systems [35] or methods [36] that are known as sand boxes. Any [7]details that dynamic interpretation can generate are calls API function, device request, tutoring records, registry updates, and storage programs. The outcomes of these simulations are denoted in the procedure of diagrams; where to even vertex means machine calls as well as the boundaries mean the calls are associated with each other. A runtime analysis method that obtains statistical features in the API call logs based on spatiotemporal information[27].In [23] Extracted applications' API requests, and then blended to construct an API-CFG prototype with regular expression graphs. In an updated version[30], the API calls were collected using n-gram methods. Malware analyzed based on API call sequences frequency analysis[21]. Implemented a tool identify malware that uses logs to track executables from instructions. The tracks are analyzed as graphs, whereby directions are represented by the decisions relating, and guidance track statistics are used for calculating transfer probabilities[24].

C. Hybrid Approach

Hybrid approaches use elements of both analytics observation. We examine some latest works that kind in this section. Moreover to established a method of classifying to malware using static analysis. They [29] use a tool called Malware DNA to remove the malware functionality. The key objective of this approach is behavior analysis that occurs in dynamic character extraction. A method called the HDM analyzer was developed[30]. In this approach, both static and dynamic analysis is implemented during a learning process when the only basis for the evaluation step is static analyses. The expected result is to take benefit of the seemingly greater dynamic study during the training process while retaining the efficiency advantage of static analysis at the scoring level. They prove, for example, that the HDM detector has greater precision, accuracy, and complexity than other techniques of static classify and dynamic classify. The dynamic analysis remained used, based on the API call sequence extracted. Presented a novel mobile malware detection method, called mad4a that utilizes the benefits of both static and dynamic analytical tools and discovers some unidentified malware properties detection by the methodologies used[38].

3. Proposed Approach

The uses of machine learning techniques are mainly have two challenges aimed at malware discovery. The primary is to create the ML technique that is discussed during the practices process of this proposed system, and the another task is to define and choose a suitable set of functions. In the procedure of developing several sub sequence arrangements the accord sequences are generated. A plurality category is a collection based on the average DAG formed by the POA technique, containing all the sequences of MSA. If the sequence family members are different a PHMM can getting extra sequence of consensus.

mov	-	mov	push	add	push	mov	jz	call	pop	-	push
mov	-	mov	push	-	push	mov	jz	call	pop	mov	push
lea	mov	mov	push	-	push	mov	jz	call	pop	mov	push
call	pop	mov	push	-	push	mov	jz	call	pop	push	xor
call	pop	mov	push	mov	mov	mov	-	-	-	-	push
call	pop	mov	push	-	-	mov	-	call	pop	mov	push
-	-	mov	mov	-	push	mov	-	call	pop	push	-
mov	-	mov	push	-	push	mov	jz	call	pop	mov	push
call	pop	mov	push	-	push	mov	-	call	pop	push	-

Figure1: MSA and the Consensus Sequence

As seen in Figure 1, to get an APHMM the consensus sequence is applied to a developed MSA. Furthermore, the attributes relevant to every group are illuminated taking ahead to progress in to the efficiency of this mathematical prototype. Remember that in form of enhanced MSA used in constructing the APHMM, the new method is better to normal PHMMs, and the model structure remains unchanged. Recently, several experiments have concentrated on the second problem, and the current methods have yielded reasonable results. The selection state of the proposed model is equal to scoring any sequence of opcodes against the designed PHMM corresponding to the malware group. Prevention and identification of high precision and speed of malware is one of the researchers' most significant targets in the world of information security. Our suggested approach to detecting malicious code is focused on static behavior and identification of statistical patterns. In this process, this article used PHMMs to classify the malicious software groups. We're able to obtain favorable outcomes in this model. The proposed approach yield accuracy 100 %, simplicity, LFPR and low computing above.

A. Training

This research article has opcode sequences for preparation and levels, which are derived from runnable files by using static classify. Many academic works on malware identification have used opcode sequences, as described in Section 2.We utilize IDA new version to take apart data and remove from the result of the static opcode sequences. Assume we take apart a compact executable file format, and acquire the ASM file format. In Figure 2 taken from ASM file.

```
.text:00401017    call sub_401098
.text:0040101C    push 8
.text:0040101E    lea ecx, [esp+24h+var_14]
.text:00401022    push offset xyz
.text:00401027    push ecx
.text:00401028    call sub_401060
.text:0040102D    add esp, 18h
.text:00401030    test eax, eax
.text:00401032    jz short loc_401045
```

Figure 2: Disassemble arrangement

Figure3: NGVCK category residues samples

The below sequences were created from ASM files belonging to the NGVCK malware family, which are the starting of certain quantities shown fig.2, Each base throughout the residue may display different kinds of N-grams. As can be seen in Figure3, the residues including the coding sequence represent the MSA matrix.

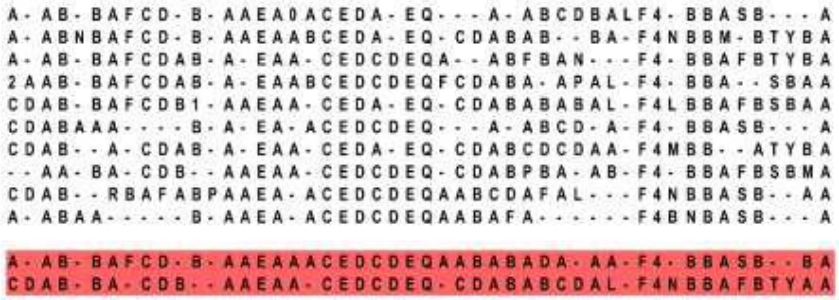

Figure3: MSA snapshot taken from NGVCK category

We delete everything operands including labels and guidelines, and maintain only the mnemonics. N separate symbols were used during the top N regular source codes, rest of the symbol for others as just a wild card. These all are primary signs could be produced from unique source codes or general N amalgamations of each sequential N (N-gram) opcodes. Patterns will then be used to sequence a PHMM, along with finding design is had to rank the documents in the test sets of virus and friendly groups. A scatterplot is constructed using the scores from a given experiment.

b. clustering process

System call sequence is an important resource for dynamic malware detection, but it is too fine-grained, so we map a system call sequence S = {s1,s2, . . . ,sn} to a high-level action sequence AS = {v1, v2, . . . , vT} where each action vi is a subsequence of system calls. We therefore partition all action sequences of malware set to some clusters (i.e., C = {c1, c2, . . . , ck}) using complete-link agglomerative hierarchical clustering method, where each cluster has high cohesion (i.e., high similarity) and less similarity with other clusters. In the process of clustering, we compute the normalized similarity of every two action sequences ASi and ASj based on Equation (1). Here, ED(ASi , ASj) is the edit distance of ASi and ASj , which is computed by the Levenshtein technique.

$$\text{Sim(ASi , ASj)} = \frac{\text{ED(ASi ,ASj)}}{\max(\text{length(ASi)},\text{length(ASj)})}$$

In the hierarchical clustering method, we plot the reconstruction error as a function of k and look for an elbow. Then, we set k at elbow. K-Set denotes the set of all candidate k values that points to the one of the elbow points. In HM3 alD, the principle rule is to select the smallest possible k that leads to set of suitable HMMs. We select the smallest k such that: (1) k ∈ k_Set; and (2) there is only at most one cluster ci where its members are not similar enough and has just few action sequences. Note that, according to our experience on HMM, if the average similarity of a cluster is less than 0.5, then the corresponding HMM is not converged accurately and leads to a high false alarm rate.

c. classification by threshold process

The first layer using the threshold approach excludes the certainly benign files from the set which lack any similarity to malware files. The similarity of each frame to benign files is evaluated only in order to exclude the files which are more similar to benign files and contain less important sequences, ultimately to detect important sequences of malware files. Therefore, benign files which show more similarity to malware files are required. The second layer aims at distinguishing different malware commands and training an HMM based on significant commands in order to acquire better results. The significance or importance of commands is evaluated based on their lack of similarity to benign files because not every part of malware program does not signify destructive essence and this justifies the elimination of certainly benign files in the first layer.

- Malware and benign files remaining in the test set are fed to the new HMM which trained only on specific parts of malware in order to determine the similarity of each file to malware files.
- Then, exploiting the defined classification threshold, benign files are separated from malware and so they are classified.

4. Experimental Results

We first address in this section the methods that can be used to derive functions from binary format. Then we offer a summary of the list of malware used for the study. After this, we reflect on the studies which have been carried out.The findings of our tests are given in Section 4.3. All additional resources including coding, datasets, and reports are provided on the Web[3] for public access.

A. Tools for Static Analysis

Advanced IDA is disassembler produces extremely exact association code belongs to runnable format. Even it exist code validation. The IDA has a versatile method that facilitates procedure, finding of features, finding of instructions, sorting of instructions, etc.

PE Explorer: This platform was made by user easy accessibility as compared to many other disassemblers. Some of the components of other disassemblers have been omitted to accomplish this purpose which has culminated in an easy and quick operation. This is similar to other popular and patented disassembling software. It also emphasizes on the transparency, user friendliness and interaction smoothness. It is commonly used to access, access and delete a range of executable Windows file formats.

This comes with several approaches which have been exported to support disassemble binary records. Most of its capabilities are: multi-architectures, disassembler specifics and approximately semantics to the dis-assembled code, and Natural support aimed at all system software.

B. Datasets

Sequential development and execution in the production of novel malware sign, single standard should not be used for long levels of time because it would not be feasible to use the methods trained for these exists data categories in the malware detection sector. And as such, not a standard dataset is available to those researchers as an appraisal guide.We were using a set of genuine data from network related documents in this report. This article accrued a dataset comprising four virus classes that had been utilized in some prior similar articles. The IDA Pro discovered the opcodes if not present. There have been 50 samples of building kits available for each family on the VX Heaven platform, except maybe the 200-sample NGVCK package, that is among the most mysterious malicious pieces of malware detection techniques available because of their complicated conversions. These datasets are often included in assessments of different malware identification approaches dealt with in a variety of related works [1, 14, 31, 41, 51-55]. As per similarity measurement is taken on the datasets by [14], samples in each family are matched two by two after calculating the similarity score using the framework mentioned in [41]. Table 1 indicates the least, most and mean number of specimens pair-wise compare in each kit for virus formation.

	NGVCK	**G2**	**VCL32**	**MPCGEN**	**Gygwin**
Low	0.01504	0.63956	0.35446	0.45012	0.14503
High	0.22029	0.85765	0.9356	0.94532	0.94412
Average	0.10098	0.75592	0.60453	0.63505	0.35342

Table 1: Low, high and average kits similarity

With an average of around 10.0 %, NGVCK produced viruses with variations ranging from 1.5 to 21.0 %. This is a significantly smaller degree of resemblance than two of the three other sources. The correlation between two versions of the same virus ranges from 34.4 to 96.6 percent for non-NGVCK generators, as well as the average G2, VCL32, and MPCGEN rates are 74.5, 60.6 and 62.7 % each method.But at the other hand, regular files offer an average 34.7 % similarity. From all of these findings we could see that the NGVCK malware vary considerably from each other, whereas the virus variants produced by the other producers are more similar to regular data. We believe that the non-NGVCK generators we evaluated really aren't nearly as accurate at metamorphic generation as NGVCK.

5. Results

To test this proposed model simulations, we applied a technique of algebraic driven polymorphic malware identification focused on HMM of profiles. We had comparison our findings aimed to a study of n-gram (with various parameters N). Proposed a PHMM based solution. Since they used a separate datasets, we applied their process using Python language and simulated the findings on the List. See Figure 4 for the effects of this process. The approach was not effective in classifying NGVCK families, as can be understood from the graph and also mentioned in the original article. The LLPO has used during the evaluation process to calculate similarity measure to identify a part of malware groups (designs). The LLPO is determined by dividing the finding of the log probability procedure by the length of a document to build an independent estimate of the volume.

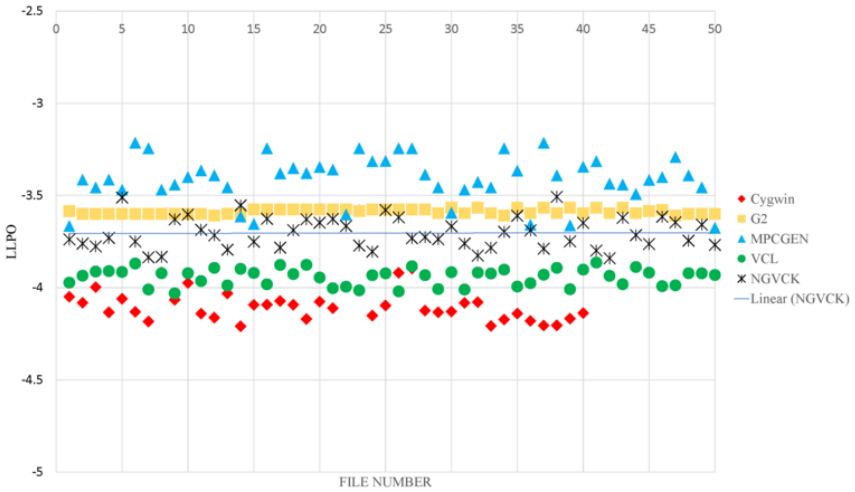

Figure 4: The Finding results of proposed method of without consensus samples

The power of the solution suggested lies in introducing consensus sequences for constructing APHMMs to the MSA. Through including the MSA from consensus serials, we are creating an advanced PHMM which places greater focus on the frequent trends that occur in every malware family. In the studies recorded in the different statistics, the influence of quantity of consensus sequences is analyzed. Figures 5 and 6, respectively, show the product of categorizingNGVCK group of un known source (malware) combined one and four consensus arrangements. As already stated, we have 200 samples for this class, where the 150 samples were used for learning and 50 samples were used for research [56-58]. The findings for the classification of all other families are presented in supplementary content IV. There were only 50 sequences usable for the majority of malware groups where the 40 are being used for learning and 10 samples used for processing. We only had 40 samples for a benevolent group.

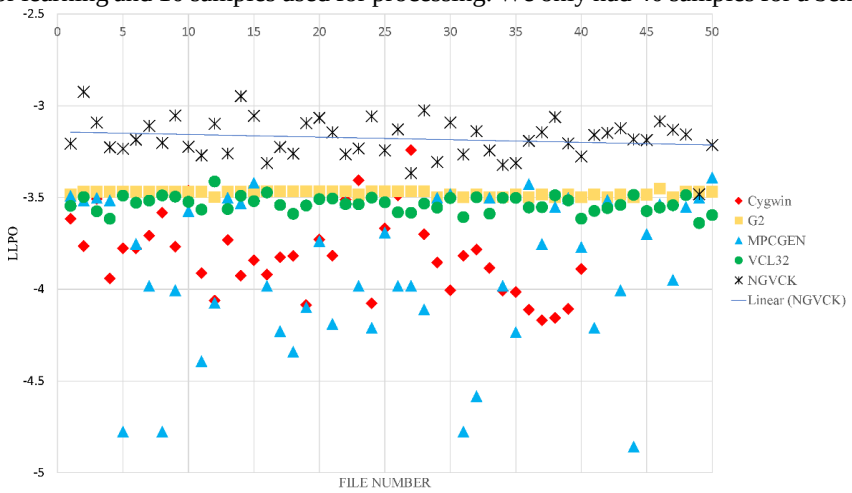

Figure 5: Proposed method findings for NGVCK family classifying by which mono consensus sample

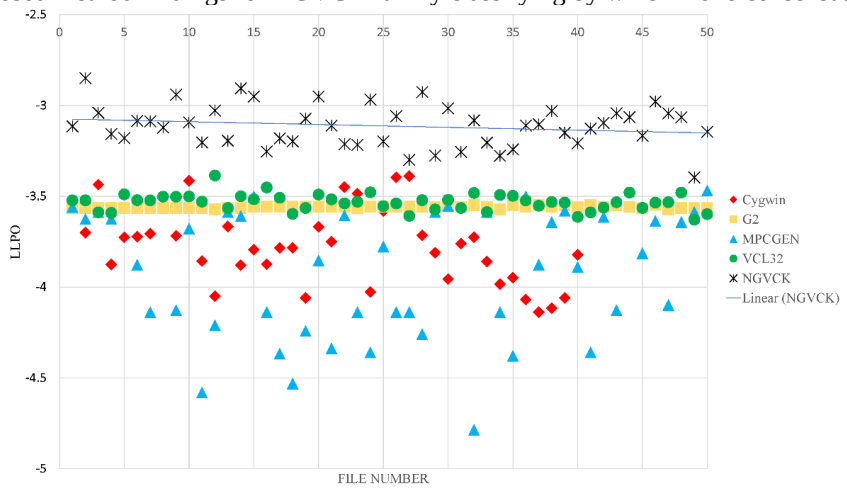

Figure 6: The proposed method finding for NGVCK groups classifying by which four consensus samples

For each regular opcode, we use a symbol during the extraction process of the function. Here we used one symbol for the remaining opcodes. Just the top 36 distinct opcodes have been used in the previous tests. The findings of the unigram method for feature extraction are seen for 37 and 200 signs in Figures 7 and 8 accordingly. In figure 9, the findings are shown on bigram with 100 symbols. While evaluating Figures 7 and 8, can see that using fewer signs with unigram features findings in stronger NGVCK class distinctions. Whenever we examine the occurrence of all probable unigram signs, this method find that the best 10 resource code represent 94% of any and all executable opcodes that occur throughout all directories, including 98% of those opcodes protected by the top 50 source code [59]. As used in the research we selected the top 36 opcodes. The number of regular ones, including bigram symbols, would increase comparatively. As we've seen in Figures 7 and 8, getting more signs will not enhance unigram function configuration results. The explanation for this is that the non-frequent signs influence the scoring emissions matrix and render excessively small score values. The signs are developed orders from the composition of N succeeding source codes for greater N values, which are special and insightful including very huge numbers of signs. The PHMM that location-specific design; use of Nitrous-grams had a confident influence on the success of the model cataloging. By combining Figures 8 and 9 with 100 signs, we notice this positive effect on bigrams.

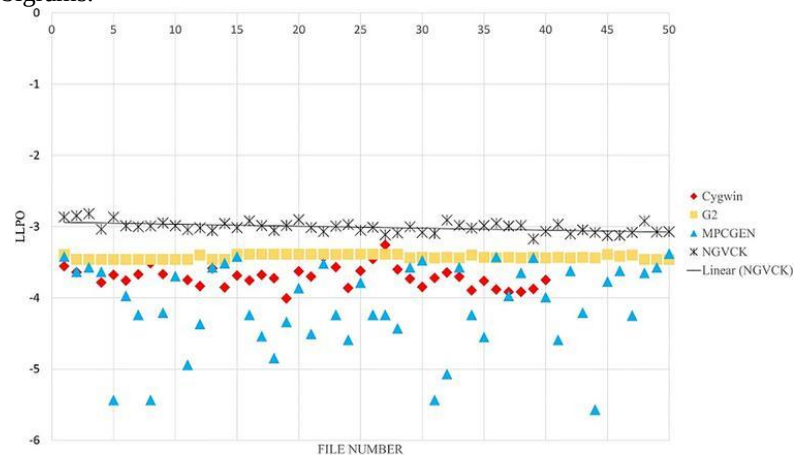

Figure 7:Theproposed method finding by which one consensus samples,37 samples, and unigram

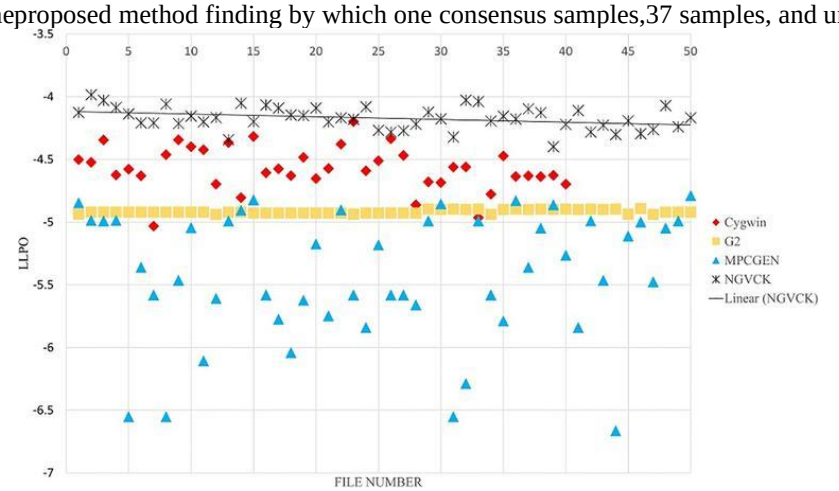

Figure 8: With one consensus sequence, unigram and 200 symbols findings for proposed method

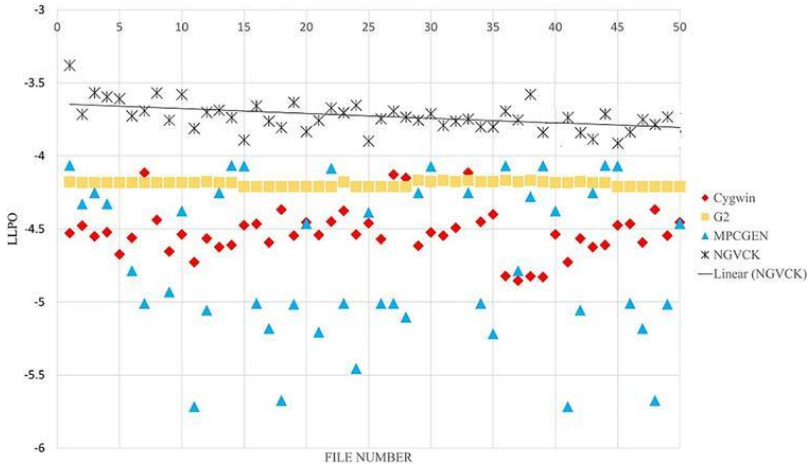

Figure 9: The proposed method finding by which one consensus sequence, bigram and 100 signs.

A description of all of the results is given in Table 2; we validate our findings with three parameters that influence on Accuracy, recall and precision directly. Begins is the quantity of consensus sequences which by highlighting the general features plays a significant role in the classification of malware families. The next one parameter is the number of various symbols used during MSA integration and the third input parameter indicates as an atomic element the number of consequential opcode classes. Although high Nitrogen-grams orders contain of more number of specific domains, the grouping may be greater. Equations both demonstrate accuracy and False Positive Rate (FPR) as following sections:

$$Accuracy = \frac{True\ Positive + True\ Negative}{True\ Positive + False\ Negative + True\ Negative + False\ Positve}$$

$$False\ Positve\ Rate = \frac{False\ Positive}{True\ Negative + False\ Positive}$$

Number of Censuses Signs	Accuracy %	False Positive Rate
One / 37 / unigram	97	0.008
One / 100 / unigram	97	0.014
One / 200 / unigram	95	0.002
Two / 37 / unigram	97	0.003
Two / 100 / bigram	100	0.001
Two / 200 / bigram	100	0.014
Four / 37 / unigram	100	0.002

Table 2: The proposed system findings experimental results on NGVCK groups, FPR and CS

For each and every malware datasets described in phase 4.2 we submitted a specific experiment for every one of the different combinations of circumstances demonstrated in table 2. We validated the impact of three parameters for each experiment, the amount of consensus signs, the sequence of n-grams as well as the number of signs.

Existing Work	Type	Accuracy	False Positive Rate
[31]	Fixed (Static)	0.98	0.16
[52]	Fixed (Static)	0.99	0.1
[54]	Fixed (Static)	0.98	0.19
[51]	Fixed (Static)	0.94	0.2
[14]	Fixed (Static)	0.94	0.0
[23]	Changeable (Dynamic)	0.98	0.2
[24]	Changeable (Dynamic)	1.00	0.0
[30]	Mixed(Hybrid)	0.97	0.2
[28]	Mixed(Hybrid)	0.97	0.5
Proposed System	Fixed (Static)	1.00	0.0

Table 3: The proposed system experimental results comparison with existing works

The empirical finding results show that even though there are a certain testing samples to construct the models, our classification technique achieves the perfect precision. We associate the success of our enhanced way to classification with that of other existing works. Since one of our key objectives was highest accuracy, we compared with just the existing research finding accuracy of 90% or higher on even a samples.

6. Conclusion

In this proposed article have produced and tested various machine code sequence based malware identification techniques. We practiced hidden Markov model profiles and correlated the identify rates for Nitro-grams based frameworks as well as consensus sequence techniques. In addition, in order to attain the ideal sensitivity rating, the FPR in model development is 0.01 while using 1 consensus sequences, and has been reduced to zero when it was used four consensus sequences. We had disassembly strategies in the pre-processing and data analysis processes in the proposed method provided in this study to derive the features required to build our computational model to improve the speed of the overall project. By interpreting byte-byte and in binary notation (hexadecimal) under formats under query files rather than obtaining data from opcodes, we can retrieve sufficient features to construct the PHMM. This service basically reduces the wants for non-machine interaction along with findings and improves the amount of processing speed particularly in the classification of large datasets.

References

1. Muhammad N. Sakib, Chin-Tser Huang, Ying-Dar Lin, Maximizing accuracy in multi-scanner malware detection systems,Computer Networks, Volume 169, 2020, 107027, ISSN 1389-1286,https://doi.org/10.1016/j.comnet.2019.107027.

2. Ö. A. Aslan and R. Samet, "A Comprehensive Review on Malware Detection Approaches," in IEEE Access, vol. 8, pp. 6249-6271, 2020, doi: 10.1109/ACCESS.2019.2963724.

3. Dong, Z. Ye, J. Su, S. Xie, Y. Cao and R. Kochan, "A Malware Detection Method Based on Improved Fireworks Algorithm and Support Vector Machine," 2020 IEEE 15th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET), Lviv-Slavske, Ukraine, 2020, pp. 846-851, doi: 10.1109/TCSET49122.2020.235556.

4. Hasan M.R., Begum A., Zamal F.B., Rawshan L., Bhuiyan T. (2020) Android Malware Detection by Machine Learning Apprehension and Static Feature Characterization. In: Bhuiyan T., Rahman M., Ali M. (eds) Cyber Security and Computer Science. ICONCS 2020. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol 325. Springer, Cham. https://doi.org/10.1007/978-3-030-52856-0_5.

5. Aghamohammadi, A., Faghih, F. Lightweight versus obfuscation-resilient malware detection in android applications. J ComputVirol Hack Tech 16, 125–139 (2020). https://doi.org/10.1007/s11416-019-00341-y.

6. Alshahwan, N.; Barr, E.T.; Clark, D.; Danezis, G.; Menéndez, H.D. Detecting Malware with Information Complexity. Entropy 2020, 22, 575.

7. Alipour, Alireza Abbas and Ansari, Ebrahim. 'An Advanced Profile Hidden Markov Model for Malware Detection'. 1 Jan. 2020 : 759 – 778.

8. Chaulagain et al., "Hybrid Analysis of Android Apps for Security Vetting using Deep Learning," 2020 IEEE Conference on Communications and Network Security (CNS), Avignon, France, 2020, pp. 1-9, doi: 10.1109/CNS48642.2020.9162341.

9. Amir NamavarJahromi, SattarHashemi, Ali Dehghantanha, Kim-Kwang Raymond Choo, HadisKarimipour, David Ellis Newton, Reza M. Parizi, An improved two-hidden-layer extreme learning machine for malware hunting, Computers &Security,Volume 89, 2020,101655, ISSN 0167-4048, https://doi.org/10.1016/j.cose.2019.101655.

10. X. Li, K. Qiu, C. Qian and G. Zhao, "An Adversarial Machine Learning Method Based on OpCode N-grams Feature in Malware Detection," 2020 IEEE Fifth International Conference on Data Science in Cyberspace (DSC), Hong Kong, Hong Kong, 2020, pp. 380-387, doi: 10.1109/DSC50466.2020.00066.

11. Jagsir Singh, Jaswinder Singh, A survey on machine learning-based malware detection in executable files, Journal of Systems Architecture,2020, 101861,ISSN 1383-7621,oi.org/10.1016/j.sysarc.2020.101861.

12. Jagsir Singh, Jaswinder Singh, A survey on machine learning-based malware detection in executable files, Journal of Systems Architecture,2020,101861,ISSN 1383-7621,doi.org/10.1016/j.sysarc.2020.101861.

13. Gharacheh, M., et al. Proposing an HMM-based approach to detect metamorphic malware. in Fuzzy andIntelligent Systems (CFIS), 2015 4th Iranian Joint Congress on. 2015.

14. Bostami B., Ahmed M. (2020) Deep Learning Meets Malware Detection: An Investigation. In: Fadlullah Z., Khan Pathan AS. (eds) Combating Security Challenges in the Age of Big Data. Advanced Sciences and Technologies for Security Applications. Springer, Cham. https://doi.org/10.1007/978-3-030-35642-2_7.

15. Jain, M., Andreopoulos, W. & Stamp, M. Convolutional neural networks and extreme learning machines for malware classification. J ComputVirol Hack Tech 16, 229–244 (2020). https://doi.org/10.1007/s11416-020-00354-y.

16. Yao, Yi et al. 'Combinational Metamorphic Testing in Integer Vulnerabilities Detection'. 1 Jan. 2020 : 1 – 9.
17. Y. Pan, X. Ge, C. Fang and Y. Fan, "A Systematic Literature Review of Android Malware Detection Using Static Analysis," in IEEE Access, vol. 8, pp. 116363-116379, 2020, doi: 10.1109/ACCESS.2020.3002842.
18. R. Surendran, T. Thomas and S. Emmanuel, "On Existence of Common Malicious System Call Codes in Android Malware Families," in IEEE Transactions on Reliability, doi: 10.1109/TR.2020.2982537.
19. Vemparala, S., et al. Malware Detection Using Dynamic Birthmarks. in Proceedings of the 2016 ACMon International Workshop on Security And Privacy Analytics. 2016.
20. Vemparala, S., Malware Detection Using Dynamic Analysis. 2015.
21. Qiao, Y., et al. Analyzing malware by abstracting the frequent itemsets in API call sequences. in Trust,Security and Privacy in Computing and Communications (TrustCom), 2013 12th IEEE International Conference on. 2013.
22. Eskandari, M., Z. Khorshidpur, and S. Hashemi. To incorporate sequential dynamic features in malwaredetection engines. in Intelligence and Security Informatics Conference (EISIC), 2012 European. 2012.
23. Eskandari, M. and S. Hashemi, A graph mining approach for detecting unknown malwares. Journal ofVisual Languages & Computing, 2012. 23(3): p. 154-162.
24. Anderson, B., et al., Graph-based malware detection using dynamic analysis. Journal in computerVirology, 2011. 7(4): p. 247-258.
25. Park, Y., et al. Fast malware classification by automated behavioral graph matching. in Proceedings ofthe Sixth Annual Workshop on Cyber Security and Information Intelligence Research. 2010.
26. Dai, J., R.K. Guha, and J. Lee, Efficient Virus Detection Using Dynamic Instruction Sequences. JCP,2009. 4(5): p. 405-414.
27. Ahmed, F., et al. Using spatio-temporal information in API calls with machine learning algorithms formalware detection. in Proceedings of the 2nd ACM workshop on Security and artificial intelligence. 2009.
28. Islam, R., et al., Classification of malware based on integrated static and dynamic features. Journal ofNetwork and Computer Applications, 2013. 36(2): p. 646-656.
29. Choi, Y.H., et al. Toward extracting malware features for classification using static and dynamic analysis.in Computing and Networking Technology (ICCNT), 2012 8th International Conference on. 2012.
30. Eskandari, M., Z. Khorshidpour, and S. Hashemi, HDM-Analyser: A hybrid analysis approach based ondata mining techniques for malware detection. 2013.
31. Khalilian, A., et al., G3MD: Mining frequent opcode sub-graphs for metamorphic malware detection ofexisting families. Expert Systems with Applications, 2018. 112: p. 15-33.
32. Fan, Y., Y. Ye, and L. Chen, Malicious sequential pattern mining for automatic malware detection.Expert Systems with Applications, 2016. 52: p. 16-25.
33. Wagner, M., et al., A knowledge-assisted visual malware analysis system: Design, validation, andreflection of KAMAS. computers& security, 2017. 67: p. 1-15.
34. Farhadi, M.R., et al., Scalable code clone search for malware analysis. Digital Investigation, 2015. 15:p. 46-60.
35. Egele, M., et al., A survey on automated dynamic malware-analysis techniques and tools. ACMcomputing surveys (CSUR), 2012. 44(2): p. 6.
36. Stiborek, J., T. Pevný, and M. Rehák, Multiple instance learning for malware classification. ExpertSystems with Applications, 2018. 93: p. 346-357.
37. Kolbitsch, C., et al. Effective and Efficient Malware Detection at the End Host. in USENIX securitysymposium. 2009.
38. Kabakus, A.T. and I.A. Dogru, An in-depth analysis of Android malware using hybrid techniques. DigitalInvestigation, 2018. 24: p. 25-33.
39. Kang, B., et al., N-gram opcode analysis for android malware detection. arXiv preprintarXiv:1612.01445, 2016.

40. Saradha, R. and S. Education, Malware Analysis using Profile Hidden Markov Models and IntrusionDetection in a stream learning setting. Supercomputer Education and Research Centre, Indian Instituteof Science, 2014.
41. Wong, W. and M. Stamp, Hunting for metamorphic engines. Journal in Computer Virology, 2006. 2(3):p. 211-229.
42. Rieck, K., et al. Learning and classification of malware behavior. in International Conference onDetection of Intrusions and Malware, and Vulnerability Assessment. 2008. Springer.
43. Huang, L. and M. Stamp, Masquerade detection using profile had hidden Markov models. computers&security, 2011. 30(8): p. 732-747.
44. Durbin, R., et al., Probabilistic models of proteins and nucleic acids. BiolSeq Anal, 1998. 14: p. 164-73.
45. Stamp, M., A revealing introduction to hidden Markov models. Department of Computer Science SanJose State University, 2004.
46. Ghahramani, Z., An introduction to hidden Markov models and Bayesian networks. International journalof pattern recognition and artificial intelligence, 2001. 15(01): p. 9-42.
47. Grasso, C. and C. Lee, Combining partial order alignment and progressive multiple sequence alignmentincreases alignment speed and scalability to very large alignment problems. Bioinformatics, 2004.20(10): p. 1546-1556.
48. Lee, C., Generating consensus sequences from partial order multiple sequence alignment graphs.Bioinformatics, 2003. 19(8): p. 999-1008.
49. Lee, C., C. Grasso, and M.F. Sharlow, Multiple sequence alignment using partial order graphs. Bioinformatics, 2002. 18(3): p. 452-464.
50. Lab, S. Understanding Partial Order Alignment for Multiple Sequence Alignment. 2017; Available from:http://simpsonlab.github.io/2015/05/01/understanding-poa/.
51. Canfora, G., A.N. Iannaccone, and C.A. Visaggio, Static analysis for the detection of metamorphiccomputer viruses using repeated-instructions counting heuristics. Journal of Computer Virology and Hacking Techniques, 2014. 10(1): p. 11-27.
52. Fazlali, M., et al., Metamorphic malware detection using opcode frequency rate and decision tree.International Journal of Information Security and Privacy (IJISP), 2016. 10(3): p. 67-86.
53. Govindaraju, A., Exhaustive statistical analysis for detection of metamorphic malware. 2010.
54. Khodamoradi, P., et al. Heuristic metamorphic malware detection based on statistics of assemblyinstructions using classification algorithms. in Computer Architecture and Digital Systems (CADS), 201518th CSI International Symposium on. 2015. IEEE.
55. Lin, D. and M. Stamp, Hunting for undetectable metamorphic viruses. Journal in computer virology,2011. 7(3): p. 201-214.
56. Sampathkumar. A, Vivekanandan. P "Gene Selection Using Multiple Queen Colonies In Large Scale Machine Learning" Journal of Electrical Engineering, 9 (6), 97-111.
57. Sampathkumar*, Vivekanandan. P "Gene Selection Using PLOA Method In Microarray Data For Cancer Classification" Journal of Medical Imaging and Health Informatics 9, 1294-1300
58. Sampathkumar, Rastogi, R., Arukonda, S. Achyut Shankar, SandeepKautish& M. Sivaram "An efficient hybrid methodology for detection of cancer-causing gene using CSC for micro array data" J Ambient Intell Human Comput (2020),Springer. https://doi.org/10.1007/s12652-020-01731-7
59. Bilar, D. Fingerprinting malicious code through statistical opcode analysis. in ICGeS'07: Proceedings ofthe 3rd International Conference on Global E-Security. 2007.