Feature Selection and Performance Improvement of Malware Detection System using Cuckoo Search Optimization and Rough Sets

Ravi Kiran Varma P[1], PLN Raju[2], K V Subba Raju[3], Akhila Kalidindi[4]
MVGR College of Engineering
Vizianagaram, AP, India

***Abstract*—The proliferation of malware is a severe threat to host and network-based systems. Design and evaluation of efficient malware detection methods is the need of the hour. Windows Portable Executable (PE) files are a primary source of windows based malware. Static malware detection involves an analysis of several PE header file features and can be done with the help of machine learning tools. In the design of efficient machine learning models for malware detection, feature reduction plays a crucial role. Rough set dependency degree is a proven tool for feature reduction. However, quick reduct using rough sets is an NP-hard problem. This paper proposes a hybrid Rough Set Feature Selection using Cuckoo Search Optimization, RSFSCSO, in finding the best collection of reduced features for malware detection. Random forest classifier is used to evaluate the proposed algorithm; the analysis of results proves that the proposed method is highly efficient.**

Keywords—Cuckoo search; rough sets; feature optimization; malware analysis; malware detection; feature reduction; clamp dataset

I. Introduction

From the past, many years' malware has become a significant security threat for systems and networks. Malware is defined as software or malicious code injected into a target system or network to make the system work abnormally [1]. Virus, Trojans, backdoors, worms, rootkits, spyware, adware, etc. are several forms of malware. In general, any malware is commonly termed as a virus, which was first framed by Fred Cohen [2] in the year 1983. Every malware is designed with a common intention of destroying or doing some illegitimate access on the system or gain access or retrieve some sensitive information from the system. The type of malware and the anti-malware or malware detection systems depends on the hardware/software platforms and the operating system. The main goal of attackers is to infect or morph malware to evade from the malware detectors.

The increase in the volume of the datasets has resulted in a decrease in performance and increased the complexity of the classification model, thereby resulting in need of feature reduction (FR). Feature reduction was defined by [3] as the "subset of features for enhancing the accuracy and at the same time decreasing the complexity of the classification model." A reduced subset is proved to be a useful subset if the number of features is reduced without the decrease of accuracy. In general, any FR techniques follow mainly two steps [4]. In the initial step, the candidate subset is selected using a search technique, and in the final step, the selected subsets are evaluated using an objective function.

The existing feature selection algorithms are classified into two approaches based on the Objective function used. They are wrapper-based [5] and filter-based [6]. Filter-based techniques use statistical methods like the dependency degree or information measurement to evaluate the candidate subset and do not depend on the classification algorithm. In contrast, wrapper-based algorithms depend on the classification algorithm to evaluate the selected features. Hall & Lloyd, [7] applied both wrapper-based methods and filter-based methods and proved that filter based technique is faster and utilizes less CPU utilization. But the main drawback is that they did not provide any accuracy after implementing the techniques.

Shabtai et al. [8] extracted features from the Linux OS and compared a total of 10 datasets. The authors implemented a feature reduction method. He compared selection methods such as chi-square, fishers score, and information gain, and at last, they proved that information gain would produce better results and obtained an accuracy of 96.8%. It used a filter-based technique, which is the main advantage of that project. The work proposed by [9] makes use of Ant Colony Optimization and Rough Sets as a filter-based feature reduction for web phishing detection and achieved good results.

Rough sets (RS) is a mathematical approach that was first discovered by [10] in the year 1982. Because of its unique method, RS has become the most widely used technique in many fields of information technology [11]. The working strategy of RS is that it first generates all possible subsets, and from among those subsets, it selects the one with the minimum number of features and, at the same time, having a maximum dependency.

The author of [12] and [13] has included in their work the advantages and primary reasons for RS being used extensively. The authors of [14] have developed a malware dataset whose features are extracted from the API call sequences. The author used RS as the feature selection algorithm along with SVM as the classifier and has achieved more significant results. Many researchers have combined meta-heuristic algorithms along with RS to improve the accuracy and to obtain an effectively reduced subset.

Optimization of a dataset is also very essential to reduce the complexity of the classification model. Optimization algorithms are classified mainly into Traditional and heuristic methods. Most of the malware detection systems use heuristic methods. Meta-heuristic algorithms are divided again into different categories. One that is inspired by natural behaviors such as Ant colony optimization [15], Bee colony optimization [16] that is inspired by the natural behavior of bees, Cuckoo search (CS) [17] technique which is inspired from the natural behavior of Cuckoo bird. Second are evolutionary algorithms and, finally, logical search algorithms. A taxonomy of optimization algorithms is shown in Fig. 1.

Thanushkodi and Suguna [18] have combined rough sets along with Bee-Colony Optimization (BCO) for analyzing a medical dataset. The author also applied various combinations with rough sets and proved in the results that BCO, along with RS, work best for his dataset. Though BCO, along with RS would produce effective results than others, the only weakness is that it consumes more time in finding the reductant subset. Liang [19] have employed a Genetic algorithm, along with RS in the marketing application. The author concluded that this hybrid approach would work effectively for clustering datasets but failed to provide the results after the application of the algorithm. Rough sets can also be used along with other meta-heuristic algorithms such as Ant Colony Optimization, Cuckoo Search, Ant bee colony optimization, etc.

Cuckoo Search (CS) is one of the optimization technique which is extensively used by the present researchers. This Optimization technique was developed by Yang and Deb [20] in the year 2009, which is based on the reproduction strategy of a cuckoo bird. The first cuckoo search algorithm was modified by [17], works effectively for non-linear problems; they used the cuckoo search algorithm along with rough sets. The author used different datasets and different optimization techniques to evaluate the datasets and provided the results. But the major drawback is that it is a wrapper based method and consumes longer run times. The author proved through his results that the modified algorithm, when used with SVM as the classifier, resulted in an average accuracy of 93.94%. Kumar and Shampa [21] used cuckoo search in the multi-reliable objective function and concluded that CS is effective when compared with that of other heuristic algorithms.

Ajit et al. [22] have used the claMP dataset to create an integrated feature set. The author tried a total of 5 classifiers and compared against each other and obtained a result of 98% by using the J48 decision tree. However, not many features are reduced. Mouhammad & Samail [23] have also used the claMP dataset and made a comparison of the results of all the classifiers. Among all the classifiers, this author has proved that the decision tree would give the best results. The author, at the end of his work, recommended that the extraction can be done much more effectively by using a genetic algorithm.

In this work the features are extracted from the PE header, which is present in all Windows executable files. The PE header has four sections embedded within it [24].

- The DOS header.
- PE file header or The Common object file format (COFF) header (also known as the File Header).
- The optional header.
- The section header.

The following are the features that can be extracted from different headers in the PE header.

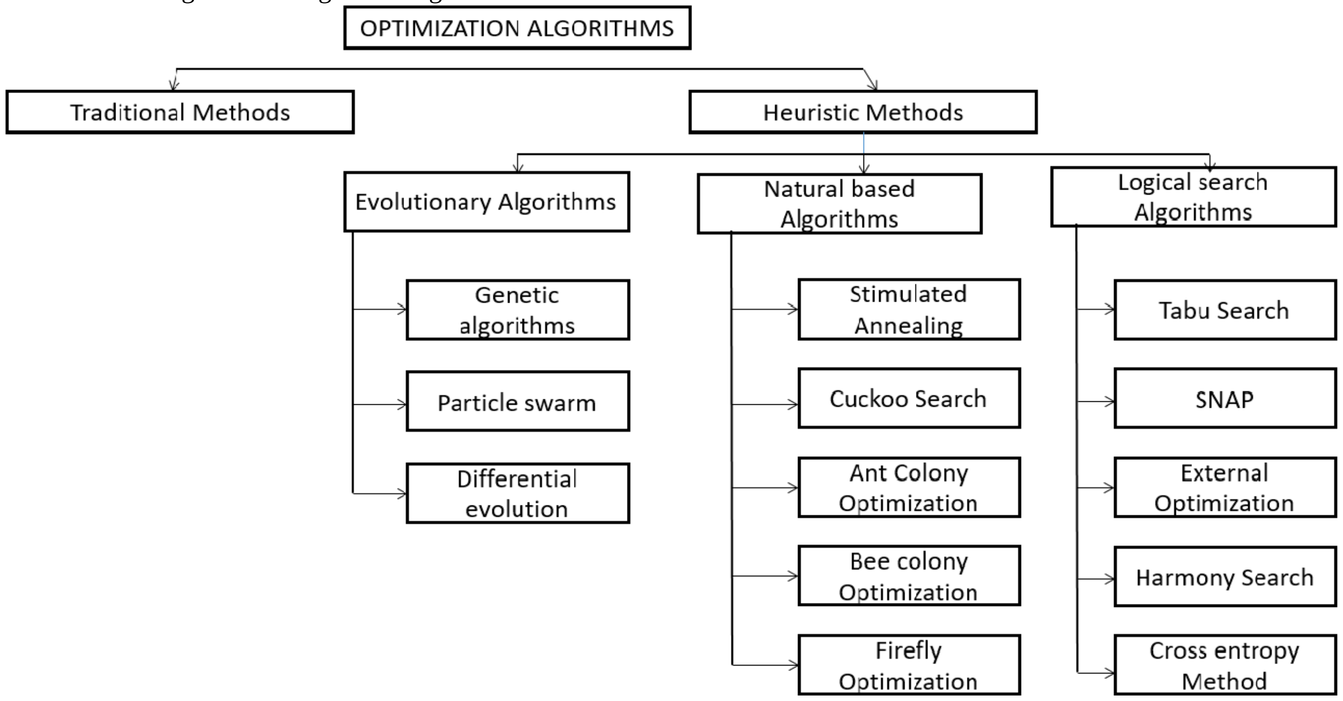

Fig. 1. A Taxonomy of Optimization Algorithms.

A. DOS Header Features

Table I describes the features that can be identified with the help of the DOS header. The feature e_magic is a fundamental feature that generally starts with the hex value 4D5A, which means 'MZ' [25] at the beginning and indicates that the file is an MS-Dos executable file.

B. PE Header Features

This header is present in front of the object file or immediately after the signature of the image file [26]. Table II describes the features which can be extracted from the PE file header. The principal analysis lies in the feature Timedatestamp. The people responsible for making a malware file first tries to change this feature.

C. Optional Header Features

This header field describes the logical structure of the PE header. Every image file would have this optional header and would provide with the loader information. This header is not optional in case of image files whereas it is optional in case of other files. This header is again subdivided into the version and size attributes. The optional header contains a version attribute that is further subdivided. They are briefly described in Table III. The optional header contains a size attribute that is divided into the features as described in Table IV [27]. The optional header includes a location attribute that is subdivided into the features that are described in Table V.

D. Section Header

The number of entries in the section table is predefined in the PE header by the Number of sections feature [28]. The features that can be extracted from a section header are mentioned in Table VI.

TABLE I. FEATURES EXTRACTED FROM DOS HEADER

Feature	*Description*	*Type*
e_magic	Magic number.	Numeric
e_cblp	Bytes on the last page of file	Numeric
e_cp	Number of pages in the file	Numeric
e_cparhdr	Header size in paragraphs	Numeric
E_maxalloc	Maximum extra number of paragraphs needed	Numeric
E_sp	Initial sp value	Numeric
E_lfanew	File address of new exe header	Numeric
e_csum	Checksum value	Numeric
e_minalloc	Minimum extra number of paragraphs needed	Numeric

TABLE II. FEATURES THAT CAN BE EXTRACTED FROM THE PE FILE HEADER

Feature	*Description*	*Type*
Timedatestamp	Date and time of file creation	Numeric
Numberofsections	Size of the section table. Windows limits this size to 96	Numeric
Symbol attribute	Define location and size of COFF header	Numeric
DLL Characteristics	This field contains a combination of 16 different features	Numeric

TABLE III. OPTIONAL HEADER VERSION ATTRIBUTES FEATURES

Feature	*Description*	*Type*
MajorLinkVersion	Linkers major version number	Numeric
MajorLinkVersion	Linkers minor version number	Numeric
MajorOperatingsystemVersion	Major version number of OS	Numeric
MajorOperatingsystemVersion	Minor version number of OS	Numeric
Majorsubsystem	Major version number of Subsystem	Numeric
Majorsubsystem	Minor version number of Subsystem	Numeric

TABLE IV. OPTIONAL HEADER SIZE ATTRIBUTES FEATURES

Feature	*Description*	*Type*
SizeofCode	Size of code sections. If multiple sections are present then sum of all those sections	Numeric
SizeofIntializedData	Size of initialized data or if multiple data sections are present then the sum of all those sections	Numeric
SizeofuninitializedData	Size of uninitialized data or if multiple data sections are present then the sum of all those sections	Numeric
SizeofImage	Size of the image	Numeric
Sizeofheader	Size of all section headers	Numeric
SizeofStackreserve	Number of bytes reserved by stack	Numeric
SizeofStackcommit	Number of bytes required to commit the stack	Numeric
SizeofHeapreserve	Number of bytes reserved for heap	Numeric
SizeofHeapcommit	Number of bytes required to commit the heap	Numeric
SizeofOptionalHeader	Indicates the size of the optional header. This size is not fixed.	Numeric

TABLE V. SECTION HEADER FEATURES

Feature	*Description*	*Type*
Raw size	Size of section when stored on disk	Numeric
Virtual size	Size of section when stored on memory	Numeric
Virtual address	Address of virtual memory	Numeric
Physical address	Address of physical memory	Numeric
Entropy	This value is not present in PE file but calculated by an external header	Numeric

TABLE VI. FEATURES EXTRACTED FROM THE OPTIONAL HEADER

Feature	*Description*	*Type*
Section Alignment	Alignment of section loaded into memory	Numeric
File Alignment	Alignment of raw data section in the image file	Numeric
Baseofcode	A pointer at the beginning of code section	Numeric
BaseofData	Pointer at the beginning of data section	Numeric
Image Base	Address of first byte when image is loaded in the memory	Numeric

This work aims at reducing the feature set to optimize the Malware detection system. Through feature reduction we can automatically minimize the classification model complexity and lower the computational complexity.

II. Rough Set feature Significance and Cuckoo search Optimization (RSFSCSO)

A. Rough Set Theory (RST)

RST is an effective mathematical approach for selecting the best candidate feature subset [29]. RST has its advantages and disadvantages. RST is a pair of upper approximation and lower-approximation. These approximations can be calculated by using the equation.

$$\hat{a}p = \{p[p]_a \subseteq p\} \quad (1)$$

$$\hat{a}p = \{p[p]_a \cap p \neq 0 \quad (2)$$

The certainty of samples and the uncertainty of samples is defined by the positive region and negative region. The sum of both regions is defined by the bounded region.

$$\text{RP(D)} = \dot{U}_{a \in u \ddot{|}D}\, \hat{a}p \quad (3)$$

$$\text{RN(D)} = \ddot{U} - U_{a \in \ddot{u}}\, \hat{a}p \quad (4)$$

$$\text{RB(D)} = U_{a \in \ddot{u}}\, \hat{a}p - \dot{U}_{a \in u \ddot{|}D}\, \hat{a}p \quad (5)$$

Before performing any feature selection algorithm, it is very important to calculate how much an attribute depends on another attribute. Dependency degree helps in calculating the amount of dependency of an attribute on another. In this work, the dependency degree plays a vital role in the calculation of the objective function.

$$\mathfrak{D}d = \curlyvee(d) = \frac{|RP(D)|}{|\ddot{U}|} \quad (6)$$

B. Cuckoo Search Optimization Technique

Cuckoos have a different style of breeding behavior, as shown in Fig. 2. Cuckoo search is inspired by the natural response of the cuckoo bird. Cuckoo lay their eggs in host nests of other birds and depend on the host birds for hosting their eggs. Sometimes the host bird discovers the cuckoo bird eggs and either abound them or change their nests to a new place. But cuckoos have the talent of producing eggs of the same color and shape as that of the host eggs. In general, cuckoo bird eggs hatch first and have more probability of getting more food. Levy's flight [4]is one mechanism that is used by the cuckoo bird to effectively search for their food. This levy's flight depends upon the levy's distribution function which is calculated by using the equation.

$$levy \sim U = v^{-\curlywedge} \quad (7)$$

Where $\curlywedge$ ranges from $0 < \curlywedge < 3$ and v are the step size.

In cuckoo search initially, the number of population nests and the iterations are initialized. After each iteration, the hybrid search is used to update the population of nests. It updates the population nest based on the conditions that the lowest quality nests are updated randomly based on the global search, and the remaining nests are updated locally using levy's flight. The new nest is updated using equation 8.

$$a^{t+1} = a^t + \propto \odot\, levy(\curlywedge) \quad (8)$$

Where a^{t+1} is the updated solution

$\propto$ is the step size which is equal to 1 in most of the cases.

$levy(\curlywedge)$ is given by the equation (7)

Binary cuckoo search [30] is a modified version of the cuckoo search which uses binary vectors in which 1 represents the selected features, and 0 represents the remaining. In this search, the nests represent the solution, and each egg represents a feature. The search initially starts by generating an initial population randomly, and then after each iteration, it updates the worst nests using levy's flight. To develop a binary vector, the equations (9) and (10) are used,

$$v\left(a_{x,y}^t\right) = \frac{1}{1+e^{-at}_{(x,y)}} \quad (9)$$

$$a_{(x,y)}^{t+1} = \begin{cases} 1, v\left(a_{x,y}^t\right) > \intercal \\ 0, otherwise \end{cases} \quad (10)$$

Where $\intercal$ belongs to [0,1] and $a_{(x,y)}^{t+1}$ represents new eggs.

Every cuckoo search algorithm makes use of a fitness function, which helps in evaluating the fitness of cuckoo as well as nests.

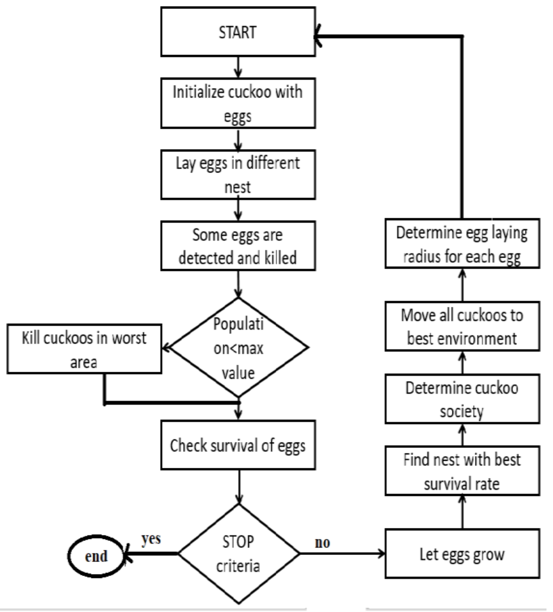
START
Initialize cuckoo with eggs
Lay eggs in different nest
Some eggs are detected and killed
Population<max value
Kill cuckoos in worst area
Check survival of eggs
STOP criteria
yes
end
no
Let eggs grow
Find nest with best survival rate
Determine cuckoo society
Move all cuckoos to best environment
Determine egg laying radius for each egg

Fig. 2. Flow Chart of Cuckoo Breeding Behaviour.

The fitness function corresponding to this work is represented in equation (11)

$$F(R) = \frac{\mathfrak{D}d}{|L|} \quad (11)$$

Where |L| represents the cardinality of the redundant set, and $\mathfrak{D}d$ is obtained from equation 7.

C. Rough Set Feature Significance and Cuckoo Search Optimization (RSFSCSO)

In this algorithm, p_a value is taken as 0.25 which means that the final solution depends 75% on the global best solution and 25% on the local best solution (lbest). The fitness function is calculated for every cuckoo and each value is compared with the global best solution (gbest). When the current fitness value is better than the global best solution then the gbest value is replaced and updated. This process is repeated iteratively until the stop criteria is met. The stop criteria in our algorithm is a maximum number of iterations which is given as input at the start of the algorithm. The max_iteration value is taken as 5;

RSFSCSO Algorithm

```
Input:
    1)  number of nests N
    2)  Maximum number of iterations i(max_iteration)
    3)  Step length α=1
    4)  $p_a = 0.25$
Output:
    5)  An optimized subset of features(RS)
                    Procedure:
    6)  The population of N host nests are initialized as
        $x_i, where\ i = 1,2,3 \dots N$
    7)  r=1; //random number
    8)  While stopping criteria not met do
    9)  for (i=0 to N)
    10) {
    11) RS= feature subset corresponding to $x_i$.
    12) // generate a new cuckoo($x_i$) with the help of levy
        flight using equation 7.
    13)         //evaluate the fitness function $f_{eval}(x_i)$ using
        the equation (11)
    14) if( $f_{eval}(x_i)$ < gbest)
    15)         {
    16)                 lbest=$x_i$
    17) gbest= ( $f_{eval}(x_i)$)
    18) RS= $x_i$
    19) Break;
    20) }// end if;
    21) } // end for;
    22) r=r+1;
    23) Sort $x_i$ by order of fitness function in descending
        order.
    24) Pick a random nest j such that j! =i;
    25) for all Abandon a fraction of worst nests by comparing
        with $p_a$ and update the nests using levy flight (equation
        7)
    26) Let the new egg generated is $x_l$
    27) RS= feature subset corresponding to $x_l$
    28) again evaluate the fitness function $f_{eval}(x_l)$ using
        equation 11.
    29) Sort the nests according to their fitness function in
        descending order.
    30)         Choose a random nest k
    31)         if($f_{eval}(x_i) >=$ $f_{eval}(x_k)$) // solutions are
        ranked according to current best.
    32)         {
    33) $x_k = x_l$
    34)         $f_{eval}(x_k) = f_{eval}(x_l)$
    35)         } //end if;
    36) end for;
    37) end while;
```

III. Experimental Results and Discussion

In this work, the analysis has been carried out on the clamp dataset [31] comprising of 55 features in the raw dataset and 72 features in the integrated dataset that are extracted from the PE header of an executable file. The proposed algorithm was implemented using 5184 samples Table VII makes a comparison of classification accuracies and Fig. 3 shows a graphical representation of accuracies in both integrated and raw datasets.

The proposed algorithm was implemented on the Java platform, and WEKA 3.9 tool is used for classification. A Windows ten operating system with 8GB RAM is used for experimentation.

The dataset used in our work, before feature selection, is analyzed with other feature selection algorithms along with the proposed algorithm. The resulted accuracies are tabulated in Table VII, and the pictorial representation of these accuracies is represented in Fig. 3. Using random forest classifiers on both raw and integrated datasets, we have obtained an accuracy of 98.3% and 99.25%, respectively. Therefore random forest classifier is selected for evaluation of our feature reduction algorithm.

TABLE VII. COMPARISON OF CLASSIFICATION ACCURACY BEFORE FEATURE SELECTION

classifier	*Accuracy (Raw Dataset)*	*Accuracy (integrated Dataset)*
J48	97.2%	97.8%
Random forest	**98.3%**	**99.25%**
Naïve Bayes	62.78%	65.20%
Random Tree	96.27%	96.18%
Decision table	93.5%	95.0%
IBK	96.9%	97.98%

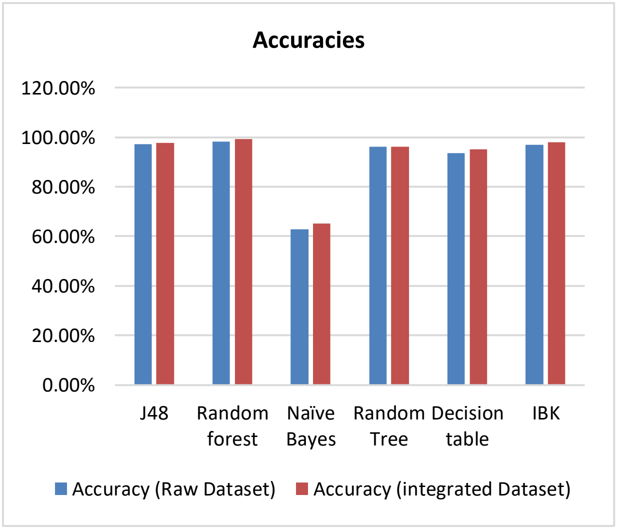

Fig. 3. Graph Representing Accuracies.

The proposed algorithm has just produced three features for the raw dataset and only two features for the integrated dataset. The critical features that were identified for the Integrated dataset are FH_characteristics and OH_Dll characteristics. The characteristics field of the file header is recognized as an essential feature by the proposed algorithm. Malware is differentiated by calculating the mean value of files. The mean value of malware files is lesser than that of benign data. The second feature identified is the DLL characteristic feature of the optional header. This feature is linked with the import table which consists of the names of files that are imported and exported. Malware files will have strange import tables when compared with regular data. Table VIII gives detailed results of accuracies after feature selection, and Fig. 4 shows the pictorial representation of the comparison of accuracies.

The features that were identified crucial for Raw dataset are characteristics, checksum and DLL characteristics. As mentioned above characteristics and DLL characteristics play a very effective role in the identification of malware file. The checksum is one other characteristic extracted from the DOS-Header. The checksum is a crucial feature because it validates at load time. It helps in preventing the entry of any damaged files or binaries. To evaluate and compare the performance of the proposed algorithm with other existing feature reduction algorithm, the same number of features are considered in all the cases. Table IX gives a brief description of other related works which used the same clamp dataset in their work. According to table IX, the proposed work produced better results than others.

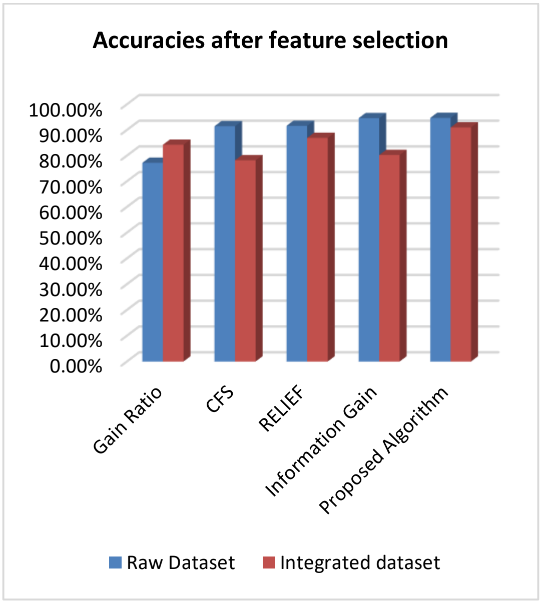

Fig. 4. Comparison of Accuracies after Feature Selection.

TABLE VIII. COMPARISON OF ACCURACIES ON RANDOM FOREST CLASSIFIER, AFTER FEATURE SELECTION

Feature selection algorithm	*Accuracy (Raw Dataset)*	*Accuracy (Integrated Dataset)*
Gain Ratio	77.19%	84.24%
CFS	91.45%	78.21%
RELIEF	91.62%	86.92%
Information Gain	94.6%	80.24%
Proposed Algorithm	**94.71%**	**91%**

TABLE IX. COMPARISON OF PROPOSED WORK WITH PREVIOUS WORKS

Dataset Reference	*No of features after reduction*	*Feature selection Algorithm(if used any)*	*Accuracy (before feature selection)*	*Accuracy (after feature selection)*
Mouhammad and Samail [23]	NA	NA	99.1%(integrate dataset)	NA
Kumar et al. [22]	15 (raw dataset)	ExtraTree Classifier	98.3% (raw dataset)	98.3%
Proposed algorithm	**3(raw dataset) 2 (Integrated Dataset)**	**Rough Set feature significance and Cuckoo search Optimization(RSFSCSO)**	**98.3%(Raw dataset) 99.25%(Integrated dataset)**	**94.71%(Raw dataset) 92%(Integrated dataset)**

V. Conclusion

The PE file header features are extracted from Windows executables in the process of identifying malware using machine learning techniques. Feature reduction is a quite essential pre-processing phase in machine learning to improve the performance and reduce the space complexity. This paper presents the implementation of a rough-set based dependency degree as an objective function in cuckoo search optimization applied to the malware detection system. A massive 94.54% reduction of data size concerning raw dataset and 97.22% reduction of data size concerning the integrated dataset is achieved at a loss of marginal 3.59% and 7.52% accuracies for raw and integrated datasets, respectively. The advantage of RSFSCSO is that it is a filter-based feature reduction, and the final model does not depend on the classifier for feature reduction. However, since the cuckoo search optimization is a population-based solution selection, it normally takes more run time than dynamic search techniques like the ACO. A comparison of various optimization techniques with RS for feature selection can be made in future work.

References

[1] M. Christodorescu, S. Jha, S.A. Seshia, D. Song and Bryant R.E, "Semantics-aware malware detection," in 2005 IEEE Symposium on Security and Privacy (S&P'05), Oakland, CA, USA, USA, 2005.

[2] Sadia Noreen, Shafaq Murtaza, M. Zubair Shafiq and Muddassar Farooq, "Evolvable Malware," in 11th Annual conference on Genetic and evolutionary computation, Canada, 2009.

[3] Koller Daphne and Sahami Mehran, "Toward Optimal Feature Selection," in Proceedings of the 13th International Conference on Machine Learning, 1996.

[4] Ahmed F. Alia and Adel Taweel, "Feature Selection based on Hybrid Binary Cuckoo Search and Rough Set Theory in Classification for Nominal Datasets," I.J. Information Technology and Computer Science, vol. 9, no. 4, pp. 63-72, 2017.

[5] Sebastián Maldonado and Richard Weber, "A wrapper method for feature selection using Support Vector Machines," Information Sciences, vol. 179, no. 13, pp. 2208-2217, 2009.

[6] S L Shiva Darshan and C D Jaidhar, "Performance Evaluation of Filter-based Feature Selection Techniques in Classifying Portable Executable Files," Procedia Computer Science, vol. 125, pp. 346-356, 2018.

[7] Mark A Hall and Lloyd A Smith, "Practical Feature Subset Selection for Machine Learning," in In C. McDonald(Ed.), Computer Science ’98 Proceedings of the 21st Australasian Computer Science Conference ACSC’98, Perth, 1998.

[8] Asaf Shabtai, Uri Kanonov, Yuval Elovici, Chanan Glezer and Yael Weiss, "“Andromaly”: a behavioral malware detection framework for android devices," Journal of Intelligent Information Systems, vol. 38, p. 161–190, 2012.

[9] Ravi Kiran Varma P and Padmaprabha K, "Web phishing detection: feature selection using rough sets and ant colony optimisation," International Journal of Intelligent Systems Design and Computing, vol. 2, no. 2, pp. 102-113, 2018.

[10] Zdzisław Pawlak, "Rough sets," International Journal of Computer & Information Sciences, vol. 11, no. 5, p. 341–356, 1982.

[11] Qinghua Zhang, Qin Xie and Guoyin Wang, "A survey on rough set theory and its applications," CAAI Transactions on Intelligence Technology, vol. 1, no. 4, pp. 323-333, 2016.

[12] Mert Bal, "Rough Sets Theory as Symbolic Data Mining Method: An Application on Complete Decision Table," Information Science Letters, vol. 2, no. 1, pp. 35-47, 2013.

[13] Rafael Bello and Rafael Falcon, "Rough Sets in Machine Learning: A Review," in Thriving Rough Sets, Studies in Computational Intelligence, vol 708, Cham, Springer, 2017, pp. 87-118.

[14] Zhang Boyun, Yin Jianping, Wensheng Tang, Jinbo Hao and Dingxing Zhang, "Unknown Malicious Codes Detection Based on Rough Set Theory and Support Vector Machine," in The 2006 IEEE International Joint Conference on Neural Network Proceedings, Vancouver, 2006.

[15] Vittorio Maniezzo and Antonella Carbonaro, "Ant Colony Optimization: An Overview," Essays and Surveys in Metaheuristics. Operations Research/Computer Science Interfaces Series, vol. 15, pp. 469-492, 2003.

[16] Anil Kumar, Gaurav Kabra, Eswara Krishna Mussada, Manoj Kumar Dash and Prashant Singh Rana, "Combined artificial bee colony algorithm and machine learning techniques for prediction of online consumer repurchase intention," Neural Computing and Applications, vol. 31, pp. 877-890, 2017.

[17] Mohamed Abd El Aziz and Hassanien Aboul Ella, "Modified cuckoo search algorithm with rough sets for feature selection," Neural Computing and Applications, vol. 29, p. 925–934, 2016.

[18] N Suguna and K Thanushkodi, "A Novel Rough Set Reduct Algorithm for Medical Domain Based on Bee Colony Optimization," Journnal of Computing, vol. 2, no. 6, pp. 49-54, 2010.

[19] Wen-Yau Liang W, "The Genetic Algorithm Incorporates with Rough Set Theory—An Application in Marketing," International Journal of e-Education, e-Business, e-Management and e-Learning, vol. 3, no. 3, pp. 271-273, 2013.

[20] Xin-She Yang and Deb Suash, "Cuckoo Search via L´evy Flights," in 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC), Coimbatore, India, 2009.

[21] Anil Kumar and Shampa Chakarverty S, "Design Optimization for Reliable embedded system using Cuckoo Search," in IEEE 3rd International Conference on Electronics Computer Technology, Kanyakumari, India, 2011.

[22] Ajit Kumar, K.S. Kuppusamy and G. Aghila, "A learning model to detect maliciousness of portable executable using integrated feature set," Journal of King Saud University - Computer and Information Sciences, vol. 31, no. 2, pp. 252-265, 2017.

[23] Mouhammd Alkasassbeh and Samail Al-Daleen, "Classification of malware based on file content and characteristics," in arXiv:1810.07252, 2018.

[24] Yibin Liao, "PE-Header-Based Malware Study and Detection," Department of Computer Science, The University of Georgia, Athens, 2012.

[25] Zatloukal Filip and Znoj Jiri, "Malware Detection Based on Multiple PE Headers Identification and Optimization forSpecific Types of Files," Journal of Advanced Engineering and Computation (JAEC), vol. 1, no. 2, pp. 153-161, 2017.

[26] R. Vyas, X. Luo, N. McFarland and C. Justice, "Investigation of Malicious Portable Executable File Detection on the Network using Supervised Learning Techniques," in IFIP/IEEE Symposium on Integrated Network and Service Management (IM), Lisbon, Portugal, 2017.

[27] D. Devi and S. Nandi, "PE File Features in Detection of Packed Executables," International Journal of Computer Theory and Engineering, vol. 4, no. 3, 2012.

[28] Jakub Ács, "Static detection of malicious PE files," Department of Computer Systems, Czech Technical University in Prague, Prague, 2018.

[29] P.Ravi Kiran Varma, V. Valli Kumari and S. Srinivas Kumar, "A novel rough set attribute reduction based on ant colony optimisation," Int. J. Intelligent Systems Technologies and Applications, vol. 14, no. 3/4, pp. 330-353, 2015.

[30] L. A. M. Pereira, D. Rodrigues, T. N. S. Almeida, C. C. O. Ramos, A. N. Souza, X.-S. Yang and J. P. Papa, "A Binary Cuckoo Search and Its Application for Feature Selection," Studies in Computational Intelligence,Springer, vol. 516, pp. 141-154, 2014.

[31] "GitHub," [Online]. Available: https://github.com/. [Accessed 28 01 2019].