

**A DIGITAL DNA SEQUENCING ENGINE FOR  
RANSOMWARE ANALYSIS USING A MACHINE  
LEARNING NETWORK**

محرك تسلسل رقمي للحمض النووي لتحليل برامج طلب الفدية باستخدام شبكة تعلم  
الآلات

by

**FIROZ KHAN**

**A thesis submitted in fulfilment  
of the requirements for the degree of  
DOCTOR OF PHILOSOPHY IN COMPUTER SCIENCE**

at

**The British University in Dubai**

**February 2020**



# **A DIGITAL DNA SEQUENCING ENGINE FOR RANSOMWARE ANALYSIS USING A MACHINE LEARNING NETWORK**

محرك تسلسل رقمي للحمض النووي لتحليل برامج طلب الفدية باستخدام شبكة تعلم الآلات

by

**FIROZ KHAN**

**A thesis submitted to the Faculty of Computer Science in  
fulfilment of the requirements for the degree of**

**PhD in Computer Science**

**at**

**The British University in Dubai**

**February 2020**

**Thesis Supervisor  
Dr Cornelius Ncube**

**Approved for award:**

\_\_\_\_\_  
Name  
Designation

\_\_\_\_\_  
Name  
Designation

\_\_\_\_\_  
Name  
Designation

\_\_\_\_\_  
Name  
Designation

Date: \_\_\_\_\_

## **DECLARATION**

I warrant that the content of this research is the direct result of my own work and that any use made in it of published or unpublished copyright material falls within the limits permitted by international copyright conventions.

I understand that a copy of my research will be deposited in the University Library for permanent retention.

I hereby agree that the material mentioned above for which I am author and copyright holder may be copied and distributed by The British University in Dubai for the purposes of research, private study or education and that The British University in Dubai may recover from purchasers the costs incurred in such copying and distribution, where appropriate.

I understand that The British University in Dubai may make a digital copy available in the institutional repository.

I understand that I may apply to the University to retain the right to withhold or to restrict access to my thesis for a period which shall not normally exceed four calendar years from the congregation at which the degree is conferred, the length of the period to be specified in the application, together with the precise reasons for making that application.

---

Signature of the student

## **COPYRIGHT AND INFORMATION TO USERS**

The author whose copyright is declared on the title page of the work has granted to the British University in Dubai the right to lend his/her research work to users of its library and to make partial or single copies for educational and research use.

The author has also granted permission to the University to keep or make a digital copy for similar use and for the purpose of preservation of the work digitally.

Multiple copying of this work for scholarly purposes may be granted by either the author, the Registrar or the Dean only.

Copying for financial gain shall only be allowed with the author's express permission.

Any use of this work in whole or in part shall respect the moral rights of the author to be acknowledged and to reflect in good faith and without detriment the meaning of the content, and the original authorship.

## **Abstract**

The research work proposes a novel detection mechanism for ransomware using machine learning approach using Digital DNA sequencing. The proposed work contains three significant phases: Preprocessing and Feature Selection, DNA Sequence Generation and Ransomware Detection.

In the first phase, data preprocessing and feature selection technique is applied to the collected dataset. The preprocessing of data includes remove missing value records and remove columns that have a negligible impact. The feature selection uses Grey Wolf Optimisation and Binary Search algorithms for choosing the best features out of the dataset. In the DNA Sequence generation phase uses design constraints of DNA sequence and k-mer frequency vector. A newly collected dataset after feature selection is used to generate the DNA sequence. In the final phase, the new dataset is trained using active learning concept, and the test data is generated using a random DNA sequence method. The data is finally classified as either ransomware or goodware using the learning methodologies.

The results are found to be promising and reconfirm the fact that the developed method has efficiently detected ransomware when compared to other methods. The thesis concludes by a discussion of future work to advance the proposed method and future directions of research on the use of Digital DNA sequencing engine for general malware detection.

## نبذة مختصرة

يقترح هذا البحث آلية جديدة للكشف عن برامج طلب الفدية باستخدام نهج التعلم الآلي باستخدام تسلسل الحمض النووي الرقمي. يحتوي العمل المقترح على ثلاث مراحل مهمة؛ المعالجة المسبقة وتحديد الميزات، توليد تسلسل الحمض النووي، وكشف برامج طلب الفدية.

في المرحلة الأولى ، يتم تطبيق تقنية المعالجة المسبقة للبيانات واختيار الميزة على مجموعة البيانات التي تم جمعها. تتضمن المعالجة المسبقة للبيانات إزالة سجلات القيمة المفقودة وإزالة الأعمدة التي لها تأثير ضئيل. يستخدم اختيار الميزة "تحسين غراي وولف" Grey Wolf Optimization وخوارزمية البحث الثنائي لاختيار أفضل الميزات من مجموعة البيانات. في مرحلة توليد تسلسل الحمض النووي ، تستخدم قيود تصميم تسلسل الحمض النووي ومتجه التردد k-mer. تستخدم مجموعة البيانات التي تم جمعها حديثاً بعد اختيار الميزة لإنشاء تسلسل الحمض النووي. استناداً إلى هذه الحسابات والمتجه، يحدث إنشاء مجموعة بيانات جديدة لمرحلة التدريب على الكشف عن برامج طلب الفدية. في المرحلة النهائية ، يتم تدريب مجموعة البيانات الجديدة باستخدام مفهوم التعلم النشط، ويتم إنشاء بيانات الاختبار باستخدام طريقة تسلسل الحمض النووي العشوائي. يتم تصنيف البيانات أخيراً على أنها إما برامج طلب الفدية أو برامج غير ضارة باستخدام خوارزمية التعلم.

النتائج واعدة، مؤكدة أن الطريقة المطورة قد اكتشفت برامج طلب الفدية على نحو فعال أكثر من الطرق الأخرى التي تم مقارنتها. تختتم الأطروحة بمناقشة العمل المستقبلي لتحسين الطريقة المقترحة واتجاهات البحوث المستقبلية بشأن استخدام محرك تسلسل الحمض النووي الرقمي للكشف عن البرامج الضارة العامة.

# Table of Contents

Chapter 1: Overview: A Digital DNA Sequencing Engine for Ransomware Analysis .....	1
1.1 Introduction to Ransomware .....	1
1.1.1 Ransomware Basics.....	4
1.1.2 Ransomware attack stages .....	8
1.2 Motivation and Problem Statement .....	10
1.3 Thesis scope .....	11
1.4 Thesis objectives and hypotheses .....	12
1.5 Research contributions.....	13
1.6 Thesis outline.....	13
Chapter 2: State of the Art Review .....	16
2.1 Introduction .....	16
2.2 Ransomware .....	20
2.3 Ransomware Family .....	21
2.4 Command-and-Control .....	29
2.5 Feature Selection .....	32
2.6 DNA Sequences.....	35
2.7 Ransomware Detection Methods.....	38
2.7.1 Signature-based Detection .....	38
2.7.2 Honeypot.....	40
2.7.3 Hashing .....	42
2.7.4 Shannon's Entropy .....	43
2.7.5 Machine Learning .....	44
2.8 Survey on Feature Selection Methods .....	46
2.9 Survey on DNA Sequencing .....	50
2.10 Survey on Online/Active Learning Algorithm .....	54
2.11 Ransomware Detection based on Machine Learning .....	59
2.12 Ransomware Detection mechanism from web .....	66
2.13 Summary .....	72
Chapter 3: Preprocessing and Feature Selection .....	73
3.1 Introduction .....	73
3.2 Overall System Architecture .....	74
3.3 Data Preprocessing .....	76
3.3.1 Data Cleaning.....	76
3.3.2 Data Integration.....	79
3.3.3 Data Transformation.....	80
3.3.4 Dimensionality Reduction .....	80
3.4 Feature Selection .....	81
3.5 Proposed Feature Selection Methodology .....	89
3.5.1 Grey Wolf Optimization (GWO) .....	89
3.5.2 Cuckoo Search .....	92

3.5.3	Proposed Feature Selection.....	95
3.6	Product: Pre-processing and Feature Selection .....	100
3.7	Summary .....	104
Chapter 4: Digital DNA Sequence Generation.....		105
4.1	Introduction .....	105
4.2	Basis of Human Genome Sequence .....	106
4.2.1	Human DNA Sequence Design.....	106
4.2.2	Criteria for Human DNA Sequence Design .....	108
4.2.3	DNA Sequence Application.....	114
4.3	Methods of DNA Sequence .....	114
4.4	Proposed Digital DNA Sequencing Methodology .....	119
4.4.1	DNA Sequence Encoding .....	120
4.4.2	DNA Sequence Design Criteria .....	123
4.4.3	K-mer Frequency.....	126
4.5	Product: DNA Sequence Generation .....	128
4.6	Summary .....	131
Chapter 5: Ransomware Detection .....		132
5.1	Introduction .....	132
5.2	Machine Learning for Malware Detection .....	133
5.2.1	Malware Analysis.....	133
5.2.2	Machine Learning Concept.....	135
5.2.3	Malware Analysis using Machine learning .....	137
5.3	Active Learning .....	139
5.3.1	Active Learning Cycle.....	140
5.3.2	Active Learning for Classification .....	141
5.3.3	Online Active Learning .....	143
5.3.4	Pool based active learning .....	143
5.4	Ransomware analysis .....	144
5.5	Proposed Ransomware Detection Methodology .....	147
5.5.1	Linear Regression Model.....	148
5.5.2	Active Learning Model.....	149
5.5.3	Ransomware Family Classification.....	152
5.6	Product: Ransomware Detection .....	155
5.7	Summary .....	157
Chapter 6: Results and Discussions .....		158
6.1	Introduction .....	158
6.2	Dataset Description.....	158
6.3	Evaluation Metrics .....	160
6.3.1	Confusion Matrix .....	160
6.3.2	Positive and Negative Rate .....	161
6.3.3	Recall, Precision, Accuracy and F-measure .....	162
6.4	Reliability Test .....	163



6.5	Experimental Results .....	164
6.5.1	Feature Selection Result .....	164
6.5.2	DNA Sequence Result .....	170
6.5.3	Prediction Result .....	174
6.5.4	Classification Result .....	181
6.6	Testing.....	183
6.7	Summary .....	185
Chapter 7: Conclusion and Future Enhancement Works .....		186
7.1	Introduction .....	186
7.2	Summary of the Research .....	186
7.3	Testing the thesis hypotheses .....	187
7.3.1	Hypothesis H1 and H2.....	187
7.3.2	Testing hypotheses H3 and H4 .....	187
7.4	Findings.....	188
7.5	Limitation of proposed work.....	188
7.6	Recommendations for future works.....	188
7.7	Summary .....	189
References .....		190
Appendix A Product- Ransomware Detector Software .....		213
Appendix B Snippets of Code used .....		214
Feature Selection .....		214
Digital DNA Sequence Generation .....		216
Ransomware Detection.....		217

## List of Illustrations

Figure 1.1: Ransomware Attack Stages .....	10
Figure 1.2: The structure and relationships between the thesis chapters .....	155
Figure 2.1: Ransomware attack using command & control .....	311
Figure 2.2: Feature selection category .....	333
Figure 2.3: Ranking Software Modules by Severity using Digital DNA Sequencing.....	38
Figure 2.4: Behavioral Traits.....	38
Figure 3.1: Process Diagram highlighting the overall System Architecture .....	75
Figure 3.2: Various process and steps involved in Data Preprocessing.....	77
Figure 3.3: Filter Approach in Feature Selection .....	83
Figure 3.4: Wrapper Approach in Feature Selection .....	85
Figure 3.5: Hybrid Approach in Feature Selection.....	86
Figure 3.6: Preprocessing and Feature Selection Architecture.....	96
Figure 3.7: Dataset Loading .....	1000
Figure 3.8: Preprocessed Data .....	1011
Figure 3.9: GWO and Binary Search -Feature Selection.....	10202
Figure 3.10: Feature Selection.....	10303
Figure 3.11: Dataset after Feature Selection .....	10404
Figure 4.1: Continuity Measure .....	<b>Error! Bookmark not defined.</b>
Figure 4.2: Similarity Measure Example .....	<b>Error! Bookmark not defined.</b> 11
Figure 4.3: H-measure example.....	<b>Error! Bookmark not defined.</b> 12
Figure 4.4: Principle of Sanger sequencing.....	11515
Figure 4.5: DNA Sequence Generation .....	12020
Figure 4.6: DNA Sequence.....	12828
Figure 4.7: DNA Sequence Measurement.....	129
Figure 4.8: k-mer Frequency Vector.....	13030
Figure 4.9: Trained Dataset .....	13131
Figure 5.1: Taxonomy of machine learning techniques for malware analysis .....	13838
Figure 5.2: The pool-based active learning cycle .....	14040
Figure 5.3: Ransomware detection architecture .....	<b>Error! Bookmark not defined.</b> 47
Figure 5.4: Test Data- DNA .....	15656
Figure 5.5: Detection Result .....	15757
Figure 6.1: Feature Selection Comparison .....	166
Figure 6.2: Execution Time Comparison .....	16767
Figure 6.3: Evaluation Metric for Proposed Work .....	17575
Figure 6.4: Classification Accuracy Comparison .....	17676
Figure 6.5: Accuracy Result-1 .....	17878
Figure 6.6: Accuracy Result-2.....	179
Figure 6.7: Accuracy Result-3.....	180
Figure 6.8: TP and FP Rate Comparison.....	181
Figure 6.9: Precision, Recall and F-measure Comparison .....	182

Figure 6.10: NB, RF and SMO Accuracy Comparison .....	183
Figure 6.11: Validation Testing Screen -1 .....	184
Figure 6.12: Validation Testing Screen-2 .....	185

## List of Tables

Table 1.1: Evolution of Ransomware (Tandon and Nayyar, 2019).....	1
Table 2.1: List of other ransomware families.....	27
Table 4.1: DNA Character Mapping.....	121
Table 4.2: Proposed DNA Character Mapping .....	122
Table 4.3: Sample Data .....	122
Table 4.4: Sample Mapping .....	122
Table 4.5: DNA representation.....	123
Table 4.6: DNA Sequence Constraints .....	125
Table 4.7: Sample K-mer Frequency .....	127
Table 5.1: Ransomware Family Code.....	152
Table 6.1: Dataset- Sample Features.....	158
Table 6.2: Features Description Code.....	159
Table 6.3: Sample Dataset with only 40 features .....	159
Table 6.4: Sample Confusion Matrix .....	161
Table 6.5: Features Count .....	165
Table 6.6: Feature Selection .....	165
Table 6.7: Execution time of Preprocessing and Feature selection .....	166
Table 6.8: Selected Features.....	167
Table 6.9: Sample Data set after Preprocessing and Feature Selection .....	169
Table 6.10: DNA Sequence for sample data .....	170
Table 6.11: Sample DNA Design Constraints.....	171
Table 6.12: k-mer frequency for sample dataset .....	172
Table 6.13: Sample Training Dataset.....	173
Table 6.14: Trained Dataset .....	174
Table 6.15: Parameters Details .....	174
Table 6.16: Evaluation Metrics.....	175
Table 6.17: Accuracy Comparison .....	176
Table 6.18: Accuracy for different parameter result-1 .....	177
Table 6.19: Accuracy for different parameter result-2 .....	178
Table 6.20: Accuracy for different parameter result-3 .....	179
Table 6.21: TP and FP Rate.....	181
Table 6.22: Precision, Recall, and F-measure .....	182
Table 6.23: Classification Accuracy .....	182

## Chapter 1:

### Overview: A Digital DNA Sequencing Engine for Ransomware Analysis

#### 1.1 Introduction to Ransomware

Ransomware is a category of malware that is created to force clients to pay a payoff to have the option to recover full access to their system. PC Cyborg Trojan was the first ransomware developed by a biologist with a PhD from Harvard, Dr Joseph Popp, in 1989. The ransomware was also called the AIDS Info Disk Trojan.

A floppy disk was the contamination vector for physically spreading the ransomware. Consumers of the contaminated computer consequently became unaware victims, saving their information to a floppy, along with the PC Cyborg Trojan. Each time clients inserted a contaminated floppy into a new computer, the ransomware infected the new machine, and the number of infected computers multiplied. The process of the ransomware infection is as follows. The code first conducts installation and execution, then the code replaces the autoexec.bat file on the target computer and keeps track of the number of times the boot process of the machine executes.

The action of the ransomware is triggered when the boot count reached 90. The action of the code is to hide all folders and consequently to encrypt all files on the machine's C drive. The action also locks the user out of the machine and makes the system entire unusable. The affected user was required to make a payment of \$189 to an organisation named PC Cyborg Corporation at a post box in Panama to reestablish functionality (Deloitte, 2016). The development of ransomware through the years is shown in Table 1.1.

**Table 1.1: Evolution of Ransomware (Tandon and Nayyar, 2019)**

Year	Name	Information
1989-90	AIDS Trojan (PC Cyborg)	The initial unidentified ransomware

2005-06	Gpcode, TROJ.RANSOM.A, Archiveus, Krotten, Cryzip and MayArchive	First to use encryption algorithms
2008	Gpcode.AK	Utilised 1024-bit RSA keys
2010	WinLock	Begun in Russia, flashed pornography content on the screen until the client would make a \$10 phone call to a premium-rate telephone number
2011	Unnamed ransomware Trojan	Locked the user's computer and directed the visitor to a fake list of phone numbers which they could call to reactivate their operating system.
2012	Reveton	Informs the user that their machine has been used to download either copyrighted material or child pornography and scare the user to pay a fine. A type of scareware
2013	CryptoLocker	Very infamous ransomware. Had expanded encryption, and was tremendously challenging to prevent.
	Locker	The message is displayed and demands a ransom payment of \$150 which the user had 72 hours to pay
	CryptoLocker 2.0	Used The Onion Router (TOR) project to enhance obscurity for payment.
	Cryptobit	Used TOR project and would encode the first 1024 bits of every file it encoded. Installed a Bitcoin miner on the victim's machine to create more profit.
2014	CTB-Locker (Curve, Tor, Bitcoin)	Leveraged the elliptical curve cryptography. TOR project for secrecy and Bitcoin for payment
	CryptoWall	Notorious CryptoLocker clone that was in charge of infecting billions of files worldwide utilising infected emails.
	Cryptoblocker	Did not encode windows documents that were over 100MB in size. Utilised AES for encryption

	SynoLocker	Targeted sinology NAS devices and would encrypt all files.
2015	Cryptowall 2.0	TOR Project is used for anonymity and was delivered through multiple attack vectors.
	TeslaCrypt, VaultCrypt	Originally targeted computers that had certain games installed. Newer variants targeted non-gaming machined
	CryptoWall 3.0	Shared some of the same features as its predecessor but added additional features such as Anti-VM check and was delivered via exploit kits
	Crypto Wall 4.0	Would not only encrypt the data in the files but the file names as well. It also would disable any system to restore functionality and shadow volume copies.
	Chimera	It was more of scareware ransomware that not only encrypted files but also threatened the user that it would publish them online when ransoms are not paid. Also known as doxing.
2016	Lockey	Not only encrypt the users' files but would first scramble the files and then rename file extensions to .locky
	SamSam	Targets servers instead of end-users; The ransomware exploits vulnerabilities in JBoss application servers and compromises the server to gain shell access. SamSam then proceeds to spread to Windows machines and encrypts their files.
2017	WannaCry	Considered to be the worst ransomware attack in history. Utilised leaked hacking tools from the NSA, called EternalBlue, an exploit that takes advantage of a defect in Microsoft's implementation of the SMB protocol.

2018	Ryuk	Can disable the Windows System Restore option on infected computers, making it challenging to retrieve encrypted data without paying a ransom.
------	------	--

### 1.1.1 Ransomware Basics

Ransomware is generally classified into two primary forms: Locker ransomware and crypto-ransomware. The objective of the locker ransomware is to lock the target machine, by any mechanisms possible, to make it difficult for the user to get access back to the infected machine (Alhawi, Baldwin and Dehghantanha, 2018). The ransomware, furthermore, displays a message on the screen demanding a ransom. The user gets access back to the infected machine again only after the user has made a ransom payment. WinLock is an early example of locker ransomware. The ransomware demanded a payment of \$10 to be made through an SMS, to get the unblock code (Sgandurra et al., 2016).

Likewise, crypto-ransomware searches for and encrypts data files present on a target machine quietly and then provides a decryption key to reaccess the encrypted files after the user has paid a ransom. The normal action of crypto-ransomware is not to encrypt the entire hard disk, but encrypt data files with specific extensions only, for example, .xls, .doc, .pdf, .jpg. The targets are often files containing spreadsheets, databases, text documents, presentations, images, files which typically contain valuable and personal user data – especially the ones that affect the users the most if data is lost (Sgandurra et al., 2016).

Usually, symmetric encryption is the preferred method of encryption for computational efficiency. However, in some cases, both asymmetric and symmetric encryption is used as a hybrid mechanism. In this methodology, a symmetric key is used to encrypt the files and additionally, a public key from the asymmetric encryption is used to encrypt the symmetric keys (Klijnsma, 2015).



In all ransomware attacks, social engineering and scare tactics are used to pressurise the affected victims to pay the ransom as soon as possible. The pressure tactic can include messages being displayed as coming from law enforcement agencies with the victim being accused of downloading illegal music or movies from the Internet.

Several techniques are frequently used by cybercriminals to install ransomware on a user's computer, as illustrated below:

***Phishing or SPAM e-mails:*** these are e-mails that have a malicious link (for example, to Dropbox (Zorz, 2015)), or an attachment (using official titles for the file, such as 'invoice', 'internal') to a malware. Alternatively, .src (Microsoft Windows Screensaver) files are veiled as .pdf with a fake symbol, to trick the targets and make them open the attachment or link. The effect of the process is such that the ransomware is downloaded onto the target machine. Peer to peer and IRC networks are also infected using similar techniques.

***Downloader and Trojan Botnets:*** These are software downloaders present on application hosting sites, with the primary objective of providing users with downloads of original files, and a hidden malicious code that downloads malware unknown to the end-user.

***Exploit kits:*** The exploit kits are used to develop weaknesses in a target system. The vulnerabilities can be created on targeted browsers to enable drive-by downloads of ransomware. The exploit kits like Angler, Nuclear, Magnitude and Neutrino (Chen and Li, 2015) are injected into the browser by redirect user traffic using rouge advertisements that are implanted to an advertising network. Adobe Flash is the most vulnerable software, with 75% of exploits in the Angler Exploit Kit and 20% of exploits related to Internet Explorer.

***Social engineering tactics:*** tactics include methodologies encompassing tricking users into downloading and purchasing fake antivirus that shows fake results of antivirus scans supposedly showing malicious programs on the end-users machine.

***Traffic Distribution Systems (TDS) (Goncharov, 2011):*** this is similar to malvertising and exploit kits where cybercriminals purchase traffic redirection to malicious sites that enable the download of exploit kit which triggers the drive-by download of the malware.

Once a system is infected by locker ransomware, there is a specific process to recover from this attack. The infected system first needs to be rebooted in Windows safe mode and then execute the installed antivirus scanner. In extreme cases, a system restores or last known good configuration can be chosen to revert to a previous clean state of the machine.

To recover from a crypto-ransomware attack is a much more challenging task as data files are encrypted by the ransomware code (Sgandurra et al., 2016). Usually, once the affected user pays the ransom, the ransomware creator would supply the victim with the decryption key as they need to preserve their reputation. If the key is not obtained, then it becomes challenging to decrypt the encrypted data files without a key.

Usually, the private key is stored on the C&C server, which monitors the ransomware. In other instances, the encrypted key is stored on the local machine itself after being encrypted using the public key technologies. There are chances of weak encryption being used by potential ransomware creators who are script kiddies. There could be instances in some ransomware where a short length key or weak encryption protocols are being used. The above instances of weak encryption are sporadic as most of the ransomware, similar to TeslaCrypt 2.0, uses sophisticated key generation techniques (Sinitsyn, 2015).

It is sometimes seen that the private keys are stored as clear text without any encryption on the infected machine itself. This weak approach to designing ransomware are infrequent. A famous example of this case is the GPCoder variants, in the ransomware family. In this variant, the keys are stored as a registry key in the infected machine itself and can lead to secure recovery of the decryption key to decrypt the locked data.

There are several measures that users and corporation can incorporate into their security measures to diminish the danger of getting infected by ransomware attacks are discussed. The four essential security practices to have in any business are highlighted below (De Groot, 2019):

- **Frequent, Tested Backups:** Having a robust backup policy is essential at any organisation. The backup process is very important dense against ransomware attacks. If a periodic incremental backup is being done, then all-important user information can be returned to an earlier save point, and any impact of a ransomware attack can be negated. The backup policy needs to have a testing process to ensure that data is backed up is complete.
- **Sensible Restrictions:** In case of external entities using the organisation's network, specific restrictions must be placed on contractors and temporary staff who use mobile devices which have company data, documents and applications. Care needs to be taken to check all mobile devices that can bring vulnerabilities to the organisation's network.
- **Structured, Regular Updates:** The software creator commonly updates most software utilised by businesses. The software update patches vulnerabilities discovered on applications and software, along with known vulnerabilities.
- **Proper Credential Tracking:** Every staff member using the network is a potential for introducing vulnerability to the system. If credentials used to access the network are not properly managed, there are possibilities of these credentials being leaked and disclosed

to people with malicious intent. Other mechanisms that can result in even higher probabilities of attack are, having weak passwords, failure to update passwords and improper implementation of restrictions.

### 1.1.2 Ransomware attack stages

Cybercriminals create ransomware to get access to a victim's system, restricting access to crucial data files, and then extorting the user to regain control of their files. Ransomware attack includes the following stages, as can be seen in figure 1.1: deployment, installation, destruction, and extortion.

All the above stages have been further explained in the below section.

#### *- Deployment*

In this stage, the malware seeks to gain access to the target device. Typical attack vectors include the following.

- **Drive-by downloads:** Malicious code is placed on websites to exploit a visitor's browser in an attempt to download malware onto the targeted machines
- **Phishing attacks:** Phishing uses social engineering methods to attract victims to spoofed websites and links. Often trustworthy sites are impersonated to get victims to provide personal information or download malicious software (Zuhair and Selamat, 2017).
- **Exploit kits:** These software tools were created to take advantage of vulnerabilities in a victim's machine automatically. These kits can be used to create code that targets applications such as Adobe Flash Player and Java Runtime Environment. Because of their automated nature, exploit kits have lowered the technical ability required to create and use malicious code.

- **Strategic web compromise:** These rely on strategic surveillance of the end-users and is also called watering hole attacks. The attacks are often used for more specific targeted attacks.

#### ***- Installation***

Once the malware has been successfully downloaded to a target system, the ransomware attempts to gain control over the system's functions. In many cases with ransomware, the infected machine attempts to establish a connection to a domain over the Internet to communicate information of the victim's computer as well as encryption keys generated to be used on the device (Chadha and Kumar 2017). Generally, at this stage, the malware also scans the local connections for other vulnerable systems.

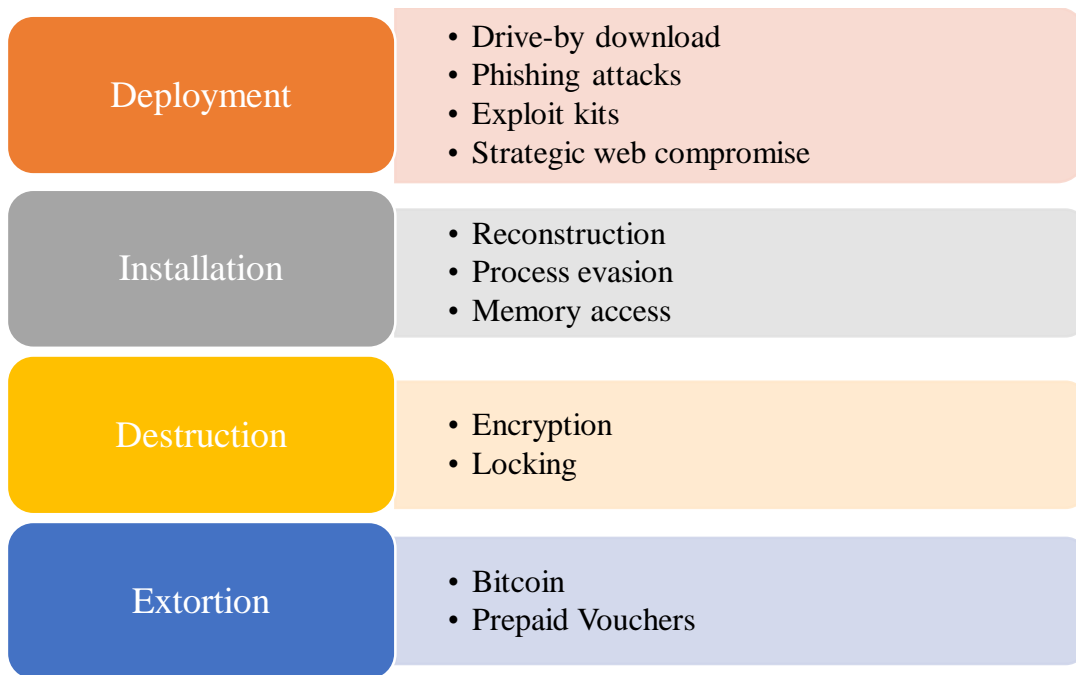
#### ***- Destruction***

In this stage, the malware encrypts files on the victim's system. Encryption methods range from using asymmetric algorithms, such as RSA, to symmetric algorithms, such as Data Encryption Standard (DES) and Advanced Encryption Standard (AES). Asymmetric algorithms use two keys to encrypt data, whereas symmetric algorithms only use one key. There are even families of ransomware that use self-designed encryption system (Gonzalez and Hayajneh, 2017).

#### ***- Extortion***

Once the encryption of the victim's files is completed, the malware displays information explaining that the device has been compromised. Payment, typically in the form of a cryptocurrency because of its anonymity features, is requested for the service to decrypt the victim's data. Most ransom demands fall in the range of \$300 and \$500 worth of bitcoins for unlocking a system. However,

some extreme variants target organisations and life-critical establishments where the ransom demand are sometimes in the range of tens of thousands of dollars.



**Figure 1.1: Phases of ransomware and the relevant methodologies that may be involved in the phase.**

## 1.2 Motivation and Problem Statement

Ransomware is defined as a type of scareware that takes control of a targeted system until the victim makes a payment to regain access (Gangwar et al., 2017). Ransomware operation varies according to the family of ransomware. The operation can be denied access to data files of users by encrypting them or taking over the boot process of a system and disable access to the system entirely. Ransomware is a subcategory under malware, and there are several behavioural differences between malware and ransomware. A significant difference is that malware aims to remain hidden from the user for as long as possible, to continue its functions in a stealth mode. Whereas, the primary purpose of ransomware is to be shown to the user and make him aware of the infection.

The most major ransomware attack in recent years was the WannaCry ransomware attack. The ransomware exploited the vulnerability named EternalBlue present in the SMB protocol of Windows systems and infected multitudes of users all over the world. User data was encrypted, and bitcoin payments were demanded in exchange for the decryption key (Mimoso, 2017). A large number of ransomware attacks occurring in the current years has prompted organisations and users to secure and create backups of their critical data. Even if a user is attacked with ransomware, having a reliable data backup policy and the presence of backups of critical data minimises the expense related to paying the ransom and potential data loss. A useful data backup policy is an integral part of the IT management process. However, the general population of unsophisticated users do not usually follow these recommendations and are becoming victims of a large number of ransomware attacks prevalent in today's world. Due to the highly profitable nature of such attacks, the newer ransomware strains are evolving, and authors continue to create more new sophisticated ransomware every day (Monrose et al., 2016).

The current defence mechanisms to detect, analyse and defend against ransomware are not very efficient and unable to protect the user from attacks. This thesis proposes an efficient machine learning algorithm for detecting ransomware to solve this problem.

### **1.3 Thesis scope**

The research work covers the following objectives:

- The thesis proposes an efficient feature selection algorithm for selecting the best features. DNA Sequence using selected features have been generated.
- The thesis also proposes an active learning algorithm for detecting ransomware. The proposed system as a whole improves the accuracy of ransomware detection. This research presents a ransomware detection method.

- The machine learning algorithm is effectively applied for ransomware detection. A real-time dataset is employed to verify the effectiveness and efficiency of the proposed work,
- A set of evaluation measures were utilised to evaluate the proposed work. The proposed research work was compared with several existing algorithms. The proposed active learning algorithm is compared to Naïve Bayes, Decision Stump and AdaBoost classification algorithms.

The proposed method has been successful in ransomware detection compared to other methods with the help of experiments and authentications. However, the methodology proposed in this thesis can be improved further. Some recommendations proposed for improving ransomware detection performance are highlighted below.

- Increase the size of the dataset
- Use the meta-heuristic algorithm for feature selection
- Explore the possibility to use other novel and classification algorithms for detection.

## **1.4 Thesis objectives and hypotheses**

The primary purpose of this research is to create a new method for detecting ransomware attack along with the associated processes, techniques, and algorithms. The research further proposes an active learning algorithm for detecting ransomware with methodologies to improve the accuracy of detection. The hypotheses that structure the research are:

**H1** It is possible to detect the malicious behaviour of ransomware by sequencing its Digital DNA.

**H2** Digital DNA mapping of software can be used for classifying and identifying ransomware.

**H3** It is possible to use active learning algorithm for detecting ransomware.

**H4** The proposed method can increase the accuracy of detection of ransomware.



## 1.5 Research contributions

This thesis has implications for detecting and classifying ransomware:

- a new method that uses Digital DNA Sequencing Engine for Ransomware Analysis with the aid of Machine Learning Network to detect and classify ransomware;
- several features from the dataset using a novel, innovative combination of Grey Wolf Optimization and Binary Cuckoo Search algorithm are selected, which improves the data quality;
- based on the selected features, Digital DNA Sequences are generated. DNA sequence constraints are computed, and k-mer frequency vector found based on the DNA sequences;
- a requirement model, a software product model, and compliance model that models compliance relationships for ransomware detection is proposed;
- an Active Learning-based Classification methodology that uniquely identifies the detected ransomware into well-known families;
- a concept demonstrator prototype which encompasses all contributions of the research and successfully detects and classifies ransomware using an active learning algorithm

## 1.6 Thesis outline

A description of how the aims of the research have been met is covered in the remaining six chapters. Chapter 2 presents a review of literature relevant to the thesis. The chapter includes topics of ransomware detection, types of ransomware attacks, and some prevention methods are presented. This literature review has provided an understanding of fundamental concepts of various topics and different perspectives provided by different experts. The chapter concludes with an assertion that there is a lack of relevant methodologies using Digital DNA sequencing engine in the field of ransomware.

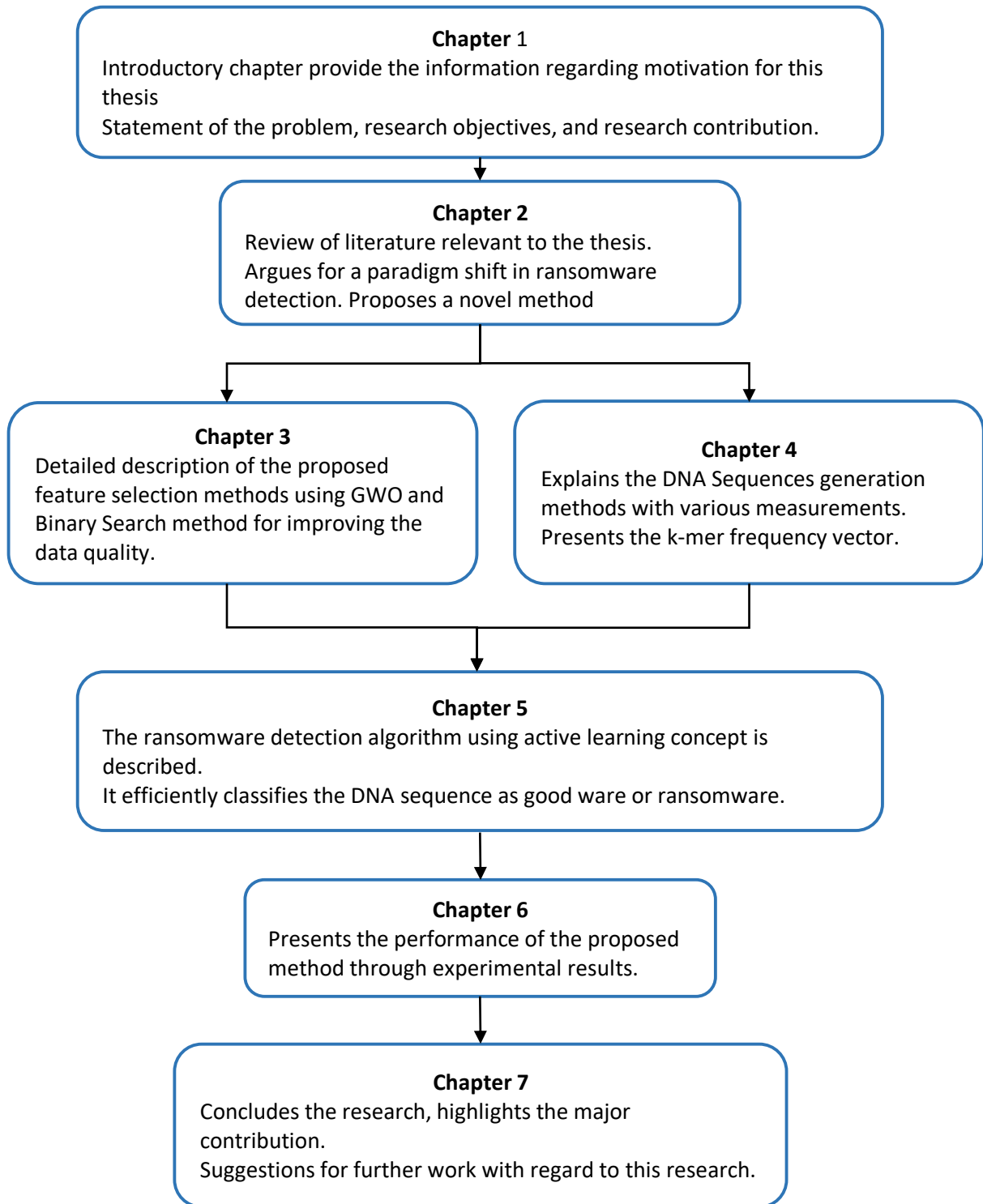
Chapter 3 provides a detailed description of the proposed feature selection methods using the Grey Wolf Optimization method and Binary Search method for improving the data quality and for selecting the best features from the dataset. The chapter explains the proposed feature selection algorithm for ransomware detection and summarises all the benefits of the proposed algorithms.

Chapter 4 explains the DNA Sequences generation methods with various measurements. This chapter also presents the k-mer frequency vector. Furthermore, the chapter uses the design constraints of DNA sequence and k-mer frequency vector. The chapter concludes with the formation of a new ransomware dataset using a k-mer frequency vector.

Chapter 5 describes the ransomware detection algorithm using active learning concept. The proposed active learning-based ransomware detection is explained in this chapter. Machine learning methods for malware detection and the concept of active learning is also explained. The process of efficiently classifying the DNA sequence as goodware or ransomware is achieved in this chapter.

Chapter 6 presents the performance of the proposed method through experimental results and verified against the proposed hypotheses. The experiments were performed with the set of sample data, and the input/output based on these sample data are explained. The results are discussed, and the interpretations based on the results are documented in this chapter.

Chapter 7 concludes and summarises the research, highlights the significant contributions and gives suggestions for further work concerning this research. The chapter also discusses how the Digital DNA sequencing engine can be further expanded to detect other malware based on the research conducted. Figure 1.2 shows the structure of the thesis.



**Figure 1.2: The structure and relationships between the thesis chapters**

## Chapter 2:

### State of the Art Review

#### 2.1 Introduction

A portion of software that is intended to destroy a system or network is called Malware. Malware is different from standard programs such that they can spread itself in the network, remain imperceptible, cause changes/harm to the impure system or network and be persistent. They can reduce the performance of the affected system and can cause the decimation of the network. A few warning signs of an infected system are as illustrated below (Nikam, 2011):

- The system may be unstable and reacts to user requests with delay as malware might be utilising system resources
- The system detects the presence of unknown new executables
- Presence of unexpected network traffic to sites that one does not hope to interface with
- The changing of system settings without user knowledge, similar to the change of a browser's homepage
- Random pop-ups are appearing in the system while using it.
- Alerts are shown by counterfeit applications which a user has never installed. Messages, for example, "Your PC is infected!" are shown, and it scares the client to download and install the program to remove the identified viruses.

Generally, the user machine displays unforeseen and unpredictable behaviour.

Malware is generally divided into several classes, depending upon the coding, how it infiltrates into the target machine and the damage that it is coded to do (Sharp, 2017) (Denning, 1988). The researcher has come up with a general classification of malware and is illustrated below:

**Virus:** Virus is a type of malware that propagates starting with a single Personal Computer (PC) then onto the next one by attaching duplicates of itself into files. These infected files need to be physically transferred from one system to another target. Transportation can be as emails attachments or using external disks and is called as the vector of the virus.

**Worm:** A type of malware that is coded to propagate automatically from a single system to other systems by transferring duplicates of itself through a network. The source system is generally a member of the network, and the worm infection can spread without user intervention.

**Trojan horse:** A type of malware that is projected to the end-user as good software, but will have a hidden malicious code embedded within the software. While the user is using the application, the hidden malicious code will operate as per the coding. The hidden coding actions may be data theft to using the machine as a Command and Control (C&C) server.

**Logic bomb:** A type of malware that is triggered by an external event. The trigger function can be date bound or a specific action in the infected system like the creation of a file or folder.

**HackTool:** A type of malware that is used by a hacker to collect information about a target system. The tool can further be used to exploit a target to gain unauthorised access to the machine — additionally, the tool aids in bypassing security mechanisms that are present on the target system. A famous example of HackTool is Netcat. Even though the network administrator uses Netcat for system analysis, the tool can be used by hackers to gain unauthorised access and to extract data from a network.

**Rabbit (Bacterium):** A type of malware which uses up all resource available on a specific system, Resources that are targeted include process control blocks, file space, or message buffers on a computer system.

**Rootkits:** They are a type of stealthy software that is intended to conceal specific processes or applications and get sustained privileged access to targeted systems. Rootkit strategies can be used at various system levels: they can trigger Application Programming Interface (API) calls in user mode or tamper with a kernel module or a device driver present in the operating system.

**Backdoor:** A backdoor is one of the significant steps present in a malware code for persistence. Once the malware infects a target machine, the code would create a user with escalated privileges on the machine to enable the malware creator to access the machine without any brute force hacking methodologies. This is known as a backdoor.

**Spyware:** A type of malware which harvests user data and user preferences from a target machine. These details are then transferred to the program creator for ulterior purposes. The malware spies on browsing habits and user activities without their knowledge or approval (Borders and Prakash, 2004). Malicious spyware creators can use the program to harvest sensitive data, observe user actions and gather username and passwords as keystrokes.

**Ransomware:** A type of malware that is prevalent worldwide in recent years. Ransomware is the most widespread and dangerous malicious programs in recent times (Symantec, 2016). The malware installs stealthily on a target system and conducts a crypto virology attack that causes a harmful effect. Once this malware infects a target system, the system or the data becomes inaccessible, and the affected party is required to pay a ransom amount to the malware creators to get access back to the system or data.

Malware can spread across the network to various user machines. The spreading mechanism used by malware is called contamination vectors (Quick Heal, 2014). The below list are some contamination vector associated with the relevant category or malware.

- Vulnerabilities: Operating system, Web browser and plugins, Adobe Reader vulnerabilities
- Boot Sector: Infecting Master Boot Record (MBR) of the physical disk
- File Infection: Parasitic infectors
- File Shares: Parasitic infectors, worms
- Email: Email worms
- IRC: Internet Relay Chat
- Network: Network worms, through vulnerabilities
- P2P Networks: IM, Kazaa
- Bluetooth: Worms for mobile devices
- Removable Media: Floppy, USB drives, optical discs
- Web Apps: Using cross-site scripting vulnerabilities

The speciality of capturing malware and analysing its activities for recognition and prevention is called malware investigation. Antivirus organisations perform malware examination to refresh signatures so that the malware can be detected and quarantined, which is the reason why using and updating an antivirus is required (Nikam, 2011).

**Static Analysis:** Reverse engineering the malicious code to understand the working of the malware and preparing its remediation, which in effect will not instigate any damages to the system, but it is tough to learn and master for the below reasons:

- Malware is coded, and not all security experts are coders or comprehend this stuff
- The user needs to realise what is being searched for, and that comes with experience.
- Getting the sample is hard for new malware; the user needs to have a honeypot to trap malware.

**Dynamic Analysis:** Relatively more straightforward but riskier than the static analysis and not very useful in the case of advanced malware. Dynamic analysis method involves creating an isolated environment and running the malware within that environment to recognise its behaviour.

The behaviour can be categorised into various categories:

- Network behaviour
- File-system behaviour
- Registry changes
- System changes

## 2.2 Ransomware

Ransomware is a type of malware that obstructs the regular usage of a computer or blocks access to data and applications until a monetary demand is fulfilled. An amount to be paid for restoring access to the computer or information is usually the demand. Some malware may be benign and may scare the user into paying the attackers by displaying a splash message on the screen, whereas other malware may be hazardous and even make modifications to the boot sector of the operating system and encrypt all files. Ransomware attacks have increased significantly since 2012. The increase is also due to the popularity of digital currency, which is the de-facto format of ransom payment. The anonymous nature of digital currency has increased the popularity of payments and enables hackers to escape prosecution, which has further increased the number of ransomware attacks (Sultan et al., 2018).

Ransomware is generally classified into two different types. The first type is known as the locker ransomware. This ransomware does not give access to the user to the system, and only a locked screen is viewable by the user where a message is displayed demanding payment to access the locked system. The payments are usually in the format of cryptocurrencies, vouchers or purchases



to be done on specific websites. The other type of ransomware is known as crypto-ransomware. This particular strain of ransomware targets user data and files and begins to encrypt predefined popular data files created and used by the end-user. The files can be documents, images, spreadsheets, among others. A ransom message is shown to the user after the encryption process is complete, which usually is a demand to pay a ransom amount to decrypt the files. Usually, a countdown timer is shown to the user along with the ransom message to create a sense of urgency and pressurise the user to make the payment within a specific time. If the time exceeds the presented deadline, the ransomware may delete the encryption key or in some worse cases delete all the encrypted files. This method is a form of scare tactic as the ransomware creator is only interested in getting the payment from the affected users.

This chapter reviews the related work of ransomware detection. The chapter outlines the topics of ransomware and their families and explains relevant survey for subjects of feature selection, DNA sequencing and active online learning. A description of the detailed study of various ransomware detection techniques is also presented.

### **2.3 Ransomware Family**

According to Becker's Hospital Review (Becker's Healthcare, 2016), the first primary target of a ransomware attack was the healthcare industry, and this attack occurred in 1989. Even after twenty-eight years, the vulnerable and human life-critical nature of the healthcare industry makes it a significant target for ransomware attacks. An Acquired Immune Deficiency Syndrome (AIDS) researcher, Joseph Popp, was the first known creator of ransomware. He distributed 20,000 floppy disks to other AIDS researchers from all over the world by stating that the program in the floppies could be used to detect the potential of an individual to acquire the AIDS virus by answering a set of questions. However, the floppies had a hidden malware program, known as the PC Cyborg or the AIDS Trojan, that infected the target computers and stayed hidden in the machines. The

program remained unactivated and would get activated only after the infected machines were powered on 90 times. Upon activation of the malware program, the program locked out the user from the machine, and a message demanding \$378 for a software lease and additional payment of \$189 was displayed (De Groot, 2019).

The current scenario in the field of ransomware has given rise to even more dangerous variants like TeslaCrypt, CryptoWall, CryptoLocker and Locky. These variants have emerged due to a new industry called ransomware-as-a-service, where cybercriminals are creating and distributing ransomware for other customers rather than spreading the ransomware themselves. The features of the most common known ransomware variants created and distributed in the past few years are explained in the below section.

#### **- *Cryptolocker***

Even though ransomware was created in the 1990s, the modern era of ransomware attacks originated by the spread of Cryptolocker in September 2013 (Zahra and Shah, 2017). The ransomware spread to over 500,000 machines worldwide, and a combined ransom payment of over \$34,000 was made (Symantec, 2014). The method of attack for this ransomware was by using spam e-mails. Once a user clicked on the ransomware link present in the spam email, the code would start functioning in the background. The operation included scanning the network devices, altering folder name and file names, and finally encrypting the data with the use of the RSA asymmetric algorithm (Salvi and Kerkar, 2016).

A newer version named CryptoLocker 2.0 written using the C# programming language, was seen to be spreading in December 2013. The newer variant had more advanced features including a stronger 2048-bit key used for encryption, bitcoin cryptocurrency as ransom payment and usage of the TOR network for anonymous communications. (Symantec, 2014). CyptoLocker 2.0 cannot be

detected before its activation by any security software (Salvi and Kerkar, 2016). One of the weaknesses of the ransomware was the usage of symmetric keys present on the C&C servers. Even with this weakness, this ransomware was a basis for future generations of ransomware families by using its source code. CryptoWall is one of the most popular derivatives, and this ransomware is the source behind the majority of all ransomware families.

### **- *CryptoWall***

CryptoWall is considered to be one of the most widely used ransomware, even though it appeared in the year 2013 (Dunn, Macaulay and Magee, 2018). The spread ratio of the ransomware was so high that in just six months, the ransomware spread worldwide and infected more than half a million machines and the code encrypted approximately 5.25 billion files. CryptoWall included a scare tactic to increase the ransom demand after a specific period. As soon as the machine was infected, the ransom demand was \$500, which doubled to \$1000 after seven days. The ransom payments had to be done in Bitcoins (BleepingComputer, 2014). The ransomware has turned out to be a huge revenue generator for its creators, where the victims paid a total of over one million dollars as ransom. The payments demanded was \$500 with actual payments done ranging from \$200 to \$10,000 as a ransom. The bitcoin ransom payments were made using the TOR network for full anonymity. The popularity and revenue generation of this ransomware enabled many variants to be created by several cybercriminals. The properties of persistence, process injection and unique file name encryption makes this ransomware extremely dangerous and very difficult to detect.

CryptoWall 2.0 was created in January 2015 with enhancements including the use of a unique bitcoin address for each victim and safely deleting original unencrypted files. Furthermore, the C&C servers were placed in the TOR network to anonymise the location of the servers. CryptoWall 3.0 was created in March 2015 (BleepingComputer, 2014) that uses I2P for anonymity (Salvi and

Kerkar, 2016). Furthermore, newer version, CryptoWall 4.0, was released in September 2015. This version was considered to be more potent than the earlier version which included enhancements including a more powerful shadow copy deletion method and encrypted file names. Also included was a ransom note presented using HTML design and new payment mechanisms. This version of CryptoWall is considered to be very strong and remains unbroken (Liska and Gallo, 2016).

#### **- *Cerber***

The ransomware Cerber encrypted files and all encrypted files are given a filename extension of “.cerber”. The ransom notes are shown as pop-up messages using HTML codes (Zahra and Shah, 2017). The ransomware uses code obfuscation techniques to create new unique samples every fifteen seconds (Calyptix, 2015). The HTML ransom message includes an audio note apart from the message displayed to the affected user and interacts with the victim. The ransomware has originated from Russia (Liska and Gallo, 2016). The operation of the ransomware code includes displaying a fake system warning similar to Windows default messages on infected machines and prompts the user to reboot the machine before the encrypting process of files. The strength of the Cerber’s C&C servers is that they can continue to be operational even if they are offline (Adamov and Carlsson, 2017).

#### **- *Locky***

The Locky ransomware has manipulated the weaknesses of macros in Word documents. The ransomware was first seen in February 2016; the spread of the ransomware is done using spam emails. Locky is considered to be so powerful that around 100,000 computers were infected in the initial days of its appearance. Once the macro is activated, the executable file of Locky is downloaded on the victim’s system. Initially, a phishing method was used as the method of attack by the Locky ransomware (Zahra and Shah, 2017). Locky is advanced ransomware that targets not

just the data files on a computer but also restore mechanism present on the system. The Volume Shadow Copy (VSC) service is targeted and encrypted so that the user cannot restore their files and after stealing user credentials, the privileges on the system are escalated. Additionally, all files and system files are also encrypted (Dunn, Macaulay and Magee, 2018). Furthermore, Locky also looks for external hard drives and database files to encrypt them and also targets virtual machines. The preferred ransom payment method is using cryptocurrencies like bitcoins (Weckstén et al., 2016). All encrypted files are renamed with an extension of “.locky” with the file name being modified to include the victim's unique ID, appended by the file's ID (Hasherezade, 2016). Overall, Locky has an excellent technical design, and it is brilliant and ruthless.

#### **- *WannaCry***

One of the most dangerous ransomware attack and brought the name ransomware to the general public is Wanna Decryptor or WannaCry. May 12, 2017, is the date of the ransomware launch, and WannaCry exploited a vulnerability present in the Microsoft's Server Message Block (SMB) protocol which was exploited by a code called EternalBlue developed by the US National Security Agency (NSA) (Dunn, Macaulay and Magee, 2018). The ransomware targeted unpatched Windows machines containing this vulnerability and quickly spread to about 150 countries and infected more than 300,000 computers. The most significant impact happened in countries like India and Russia, which had a large number of unpatched Windows XP OS based machines. The ransom note was unique and displayed the ransom message in 28 different languages, and the demand was for an amount of \$300. The lack of applying a patch released by Microsoft allowed WannaCry to spread very quickly over the Internet (Anjana, 2017). The primary attack vectors for WannaCry included running attachments received through e-mail before checking, accessing the target systems directly, and accidental download of malware from web pages (Delaney, 2017). Features of WannaCry include an encryption methodology of RSA-2048, using the TOR network

to anonymise communications with the C&C server and using VSS to delete VSCs of the files. (Adamov and Carlsson, 2017).

#### **- *Petya***

Ransomware properties and its actions have been taken a step further with the creation of Petya. The Petya ransomware incorporated encryption of hard disks in addition to files and prevents the infected system from turning on. The infected machine is also autonomous, where once infected, Petya can work without an active Internet connection, and contact with the C&C server is not needed (Adamov and Carlsson, 2017).

Petya's operation is targeted at the Master Boot Record (MBR) located on the first sector of the hard disk where the boot loader is present. The ransomware code produces a blue screen of death crash by overwriting the MBR of the infected system. Another addition in the Petya code includes a process that deletes the unique key that is used to encrypt the Master File Table (MFT) (Zahra and Shah, 2017) and uses the "EternalBlue" exploit, which targets SMB v1 to reach its victims (Symantec Security Response, 2017). After infection, the system displays a ransom note that is shown on the screen.

#### **- *Bad rabbit***

The SMB service vulnerability is also used by Bad Rabbit to spread itself to external and internal networks via SMB service similar to Petya and WannaCry. The code in the ransomware retrieves all data obtained from the infected computer along with user credentials defined within itself. An attack vector used by Bad Rabbit is by presenting itself as an Adobe Flash Player update patch and was seen in September 2017. Once activated, the process of the ransomware includes adding the ".encrypted" extension after encrypting the file (Bi Zone, 2017).

The demanded ransom for decrypted files infected by Bad Rabbit is about \$280 (Dunn, Macaulay and Magee, 2018). There is no assured method to protect a system and do not potentially fall victim to a Bad Rabbit attack. Some security vendors claim that their applications can be used for protection against Bad Rabbit. According to Kaspersky Lab, users can prevent the execution of file “C:\Windows\cscc.dat and c:\windows\infpub.dat” to prevent infection (Palmer, 2017).

A variety of research has been conducted in the field of ransomware, and it is unknown if it is possible to decrypt files locked by most ransomware without paying the amount demanded. Academics emphasise that those who get infected by ransomware should not pay the amount demanded, as this can cause more growth of ransomware variants.

Table 2.1 shows other known variants and ransomware families (TrendMicro, 2019).

**Table 2.1: List of other ransomware families**

Family Name	Aliases	Description
ACCFDIFISA	Anti Cyber Crime Department of Federal Internet Security Agency Ransom	First spotted early 2012; Encrypts files into a password protected; Cybercriminals behind this ransomware asks payment thru <i>MoneyPak</i> , <i>Paysafe</i> , or <i>Ukash</i> to restore the files and unlock the screen; Known as a multi-component malware packaged as a Self-extracting (SFX) archive; May come bundled with third-party applications such as <i>Sdelete</i> and <i>WinRAR</i>
CRYPURA	PayCrypt	Encrypts files and appends the corresponding email address contact for file decryption; PayCrypt version

		appends .id-{victim ID}-paycrypt@aol.com to files it encrypts.
CRYPCTB	Critroni, CTB Locker, Curve-Tor-Bitcoin Locker	Encrypts data files; Ensures there is no recovery of encrypted files by deleting its shadow copies; Arrives via spam mail that contains an attachment, actually a downloader of this ransomware; Uses social engineering to lure users into opening the attachment; Uses TOR project to mask its C&C communications
KOLLAH		One of the first ransomware that encrypts files using specific extension names; Target files include Microsoft Office documents, PDF files, and other files deemed information-rich and relevant to most users; Adds the string <i>GLAMOUR</i> to files it encrypts.
REVETON	Police Ransom	Locks screen using a bogus display that warns the user that they have violated federal law; Message further declares the user's IP address has been identified by the Federal Bureau of Investigation (FBI) as visiting websites that feature illegal content.
KRYPTOVOR	Kriptovor	Part of a multi-component infection; aside from its crypto-ransomware component, it has an information-stealing component that steals specific files, processes list, and captures



		desktop screenshot; uses an open-source Delphi library called <i>LockBox 3</i> to encrypt files
CRYP SALAM	Salam	Encrypts files and drops a ransom note formatted as {month}-{day}-{year}-INFECTION.TXT; Asks the users to contact the ransomware creator via email to decrypt the files
JIGSAW	Jigsaw	Deletes files and increases ransom amount each hour; Some variants have live chat support for its victims; Some use porn-related ransom messages
MIRCOP	Mircop	Arrives as spam with an attached macro-enabled document abusing Windows PowerShell to execute; Ransom note displays hooded figure wearing a Guy Fawkes mask
APOCALYPSE	Apocalypse	Appends the .encrypted extension to encrypted files; Requires the victim to email the hacker for ransom instructions

## 2.4 Command-and-Control

By definition, a Command and Control (C&C) is a computer that is under the control of a cybercriminal or malicious person who uses the server to send code instructions to infected machines that aid the action of the malware that is present on the infected machines. Additionally, the C&C can be used to collate and collect information from all infected machines. Establishing a

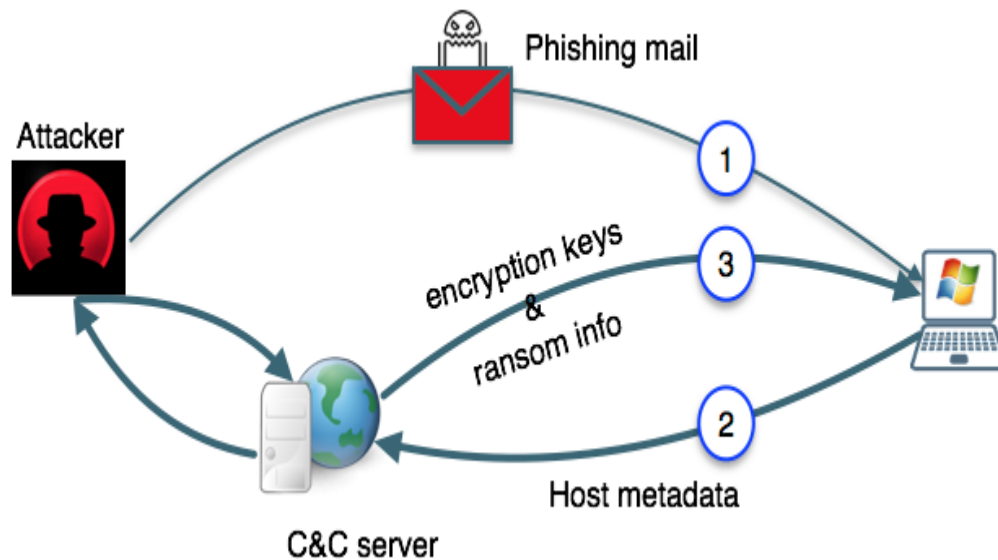
C&C communications network is an essential step for the cybercriminals to move into a target network.

C&C servers are used to set up a botnet, and the server positions itself as the headquarters for the compromised machines. The server can then pass several actions to more potential target using the nodes present in the botnet. The potential actions can include disrupting webs services, steal data, spread malware and other nefarious activities, among others. There are three basic models of C&C systems that use botnets which are, the centralised model, the random model, and the Peer-to-Peer (P2P) model.

The regular operation of ransomware is such that upon infecting a target machine, the code present in the ransomware will trigger the infected machine to contact the C&C server looking for additional instructions for its actions. Additional operations of the ransomware are triggered by these instructions which may be in the form of encryption methodology to used, files to be targeted for encryption, delay in executing the encryption, propagating itself to other machines, among others. The instructions also act as a medium to collect data about the infected machine to get more information about the target. Information is collected about the installed operating system, browsers and applications, and other identification information like IP address, domain name, and computer name to detect if the infected machine is a high-value machine for the ransomware creator. The identification information can be used to determine if the infected machine can be used for other purposes like information stealing or launching botnet attacks, rather than demand a ransom (Gallo and Liska, 2017).

A malicious weblink or an email attachment is usually the first point from which a user machine can get infected with ransomware and is considered to be the start of a ransomware attack. Once the ransomware has infected the machine, the action of ransomware begins and depending on the

variant; the action could vary. The action of encryption before spread also depends on the variant of the ransomware. Usually, the ransomware needs an encryption key to encrypt the data present on the infected machine. This key can either be derived locally from within the infected machine itself or obtained from the C&C server. The local keys generated from the infected machine is considered to be weak, and encryption is done using the same key by the ransomware. A user can manage to recover from such an attack by reverse-engineering the binary files of the ransomware to find the key and hence decrypt all encrypted files. However, when the ransomware uses a key provided by the C&C server, the encryption is usually much stronger, and it is challenging for the end-user to obtain the decryption keys. Almost all successful infection of ransomware like Locky, CryptoLocker, WannaCry, Cerber and TeslaCrypt use the C&C server to store and retrieve the decryption keys for their operation.



**Figure 2.1: Process of ransomware attack occurring using command & control and the use of encryption key management in tandem with the C&C server.**

The different stages of a ransomware attack are shown in Figure 2.1. The figure shows the steps of how ransomware communicates with a C&C server and is using encryption key management (Singh, 2018). Different symmetric and asymmetric encryption methods can be used by ransomware

for its successful operation. Algorithms RSA and AES can be used for generating such encryption keys, or a combination of symmetric and asymmetric can be used in tandem. In this hybrid method, an AES key is attached to the code of the ransomware and is used to encrypt the infected machine files. The key is then further encrypted using an RSA public key, and the corresponding private key is stored on the C&C server. Successful decryption would first need the private key to derive the symmetric key of AES, which will then be used to decrypt the data file.

## **2.5 Feature Selection**

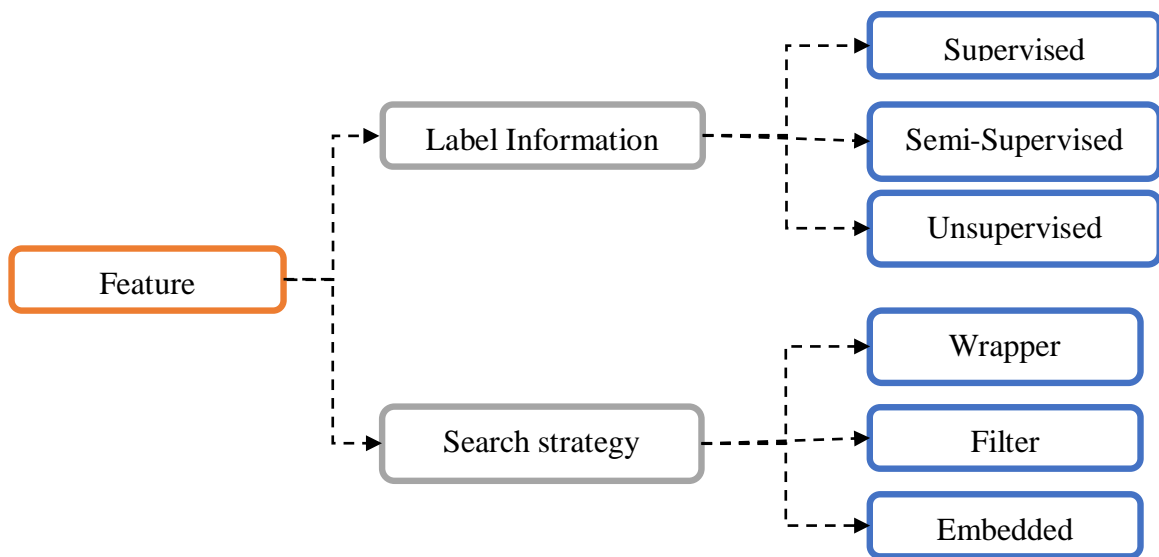
In the current scenario of computing in the world, more and more data is being generated by all sorts of devices, be it sensors, transaction systems or business systems. Almost all applications that deals with any business or day-to-day activities produce a tremendous amount of data. Examples of such vast volumes of data can be found in industries like malware, text manipulation, image processing and retrieval, and transaction systems. All these data produced have vast numbers of attributes associated with the data and are stored in massive databases.

The data and its attributes are so large that humans cannot manually process and derive meaningful information from them in an efficient manner. In such situations, technologies related to data mining come in where algorithms are used by machines to conduct machine learning to process this massive amount of data and find patterns. Even with the use of machine learning algorithms and processes, retrieving relevant and non-redundant data is a cumbersome task due to a large number of attributes associated with the data. Additionally, the pace of data being generated far outpaces the task of processing the data (Chormunge and Jena, 2018). Usually, during data collection, all relevant features associated with the data are also collected. While processing the data using machine learning, any irrelevant features do not affect the learning process but may influence the results of the process by diminishing the accuracy of machine learning algorithms.

Furthermore, the computational cost of the learning process is also adversely affected by the presence of irrelevant features.

The issue to tackle is the dimensionality problem. Due to this issue, many machine learning algorithms fail while they are handling an enormous amount of data and the presence of noisy features in data also affects and reduces the functionality of machine learning algorithms.

To resolve the dimensionality problems in machine learning and to differentiate and select between relevant and non-redundant features, a technique called feature selection is applied. The selection of appropriate attributes using feature selection methodology can contribute to machine learning algorithms becoming more accurate, scalable, and reliable. Figure 2.2 shows the feature selection methods (Miao and Niu, 2016).



**Figure 2.2: Different feature selection categories that are available to select appropriate attributes. The selection of appropriate attributes using feature selection methodology can contribute to machine learning algorithms becoming more accurate, scalable, and reliable**

Feature selection algorithms and methods are designed to comprehend the relation between feature and target variables, along with reducing the computational cost required to solve issues. Feature selection should also perform effective dimensionality reduction when dealing with high dimensional datasets where the number of observations is less than the number of features. Overall, a useful feature selection algorithm should deliver advantages such as identification of irrelevant features, produce a better classifier model, better insight into data, and enhance generalisation along with improving the predictor performance.

Concerning ransomware and other malicious software, the features and attributes found in the file define the behaviour of the software. Several features of the files, including API (Application Programming Interface), instruction sequences and text strings, are extracted for use by the detection system. Modern ransomware and malware have a large number of such characteristics, and these substantial number of features overloads the detection system during the process of malware detection. This, in effect, reduces the pace at which how the system responds to new infections (Jiang, Zhao, and Huang, 2011).

Moreover, malicious software creators are now innovating by obfuscating the primary malicious code by including a large amount of redundant, irrelevant code to escape detection. During the training process of the detection model, it is often seen that samples of malware are generally lesser than benign software and the dataset is unbalanced. This imbalance requires a malware detection system to remove irrelevant or redundant features and select the most important features to improve malware detection efficiency and accuracy (Jiang, Zhao, and Huang, 2011).

This research work has used two efficient feature selection algorithms, GWO (Grey Wolf Optimization) and Cuckoo Search for selecting the best features from the dataset. GWO is a new meta-heuristic algorithm developed by Mirjalili, Mirjalili, and Lewis (2014). The proposed

algorithm is inspired by the leadership behaviour and distinct method of hunting by grey wolves. The algorithm is population-based meta-heuristic and can avoid local optima stagnation to some extent. The algorithm has good convergence ability towards the optima (Kohli and Arora, 2018).

Researchers Yang and Deb (2009) have proposed the Cuckoo Search (CS) algorithm as a heuristic search algorithm. The basis of the algorithm is the strategy of reproduction strategy adopted by cuckoos in the wild. Usually, cuckoos do not build their nests and lay their eggs in the nests of host birds such as crows or other similar species of birds which have their eggs. If the host bird discovers the alien egg, they will take one of two actions, either destroy the alien egg or abandon the nest completely to build a new one. To apply this as an optimisation tool, Yang and Deb (2009) proposed three rules:

- “Every cuckoo lays one egg, which represents a set of solution co-ordinates, at one time and the egg is laid in a random nest.”
- “A fraction of the nests containing the best eggs, or solutions, will carry over to the next generation.”
- “The number of nests is fixed, and there is a probability that a host can discover an alien egg. If this happens, the host can discard the egg or the nest.”

The two algorithms are individually used and selects the features from the dataset. These selected features are intersected to find a better feature set.

## **2.6 DNA Sequences**

DeoxyriboNucleic Acid (DNA) sequencing refers to methods used to decide the order of the nucleotide bases in a molecule of DNA. Usually, the four nucleotide bases used in human genome sequencing are adenine (A), guanine (G), cytosine (C), and thymine (T). Knowledge of DNA sequences and its manipulation has turned out to be essential for fundamental biological research.

The discovery of DNA sequencing methodologies have significantly increased biological research and discovery, and its applications range to several fields like biological forensics, genetic diagnosis, biotechnology, and biological systems.

DNA is the information store consisting of a very long chain made up of alternate sugar and phosphate group that eventually dictates the structure of every gene product and is used to differentiate organisms (Watson and Crick, 1953). Complete set of instructions are present in the order of these bases along DNA that constitute the genetic inheritance properties. The human genome project was proposed to sequence the human genome, and modern DNA sequencing technologies have aided this proposal by sequencing the genomes quickly. (Kumar, 2012).

Forensic sciences and research related to genomes benefit significantly from DNA sequencing methodologies. Reliability and accuracy is the most significant objective of DNA sequence generation method. Below are a few common automated issues related to DNA sequencing, as highlighted by Kumar (2012).

- “Failure of the DNA sequence response.”
- “Mixed-signal in the trace (various peaks)”
- “Short read lengths and low-quality data.”
- “Excessive free dye peaks ‘dye blobs’ in the trace.”
- “Primer dimer arrangement in a sequence reaction.”
- “DNA polymerase slippage on the template mononucleotide regions.”

Agriculture is a field in which a significant role is played by DNA sequencing. The field of agriculture benefits immensely from human genome sequencing. Agriculturists have sequenced and mapped the whole genome of microorganisms to increase and have different yields of food plants. Crop yields have been improved by increasing the resistance of food plant against insect















and pests. The resistance has been increased by the manipulation of specific genes of bacteria (Bisht and Panda, 2014).

Researchers Bisht and Panda (2014) have further illustrated that human genome sequencing can be used to identify genes related to hereditary or transmitted diseases. The impact of human genome sequencing is highly significant in the field of medical research. Crime scene investigations concerning the field of forensic sciences also benefit from human genome sequencing. Criminals can be traced and litigated using human genome evidence derived from objects found at the crime scene like hair, nail, skin, or blood samples. Paternity tests and gene manipulation methodologies also significantly benefit from human genome sequencing. Polypeptide sequences can be discovered using DNA sequences that use human genome sequencing from data banks (Bisht and Panda, 2014). DNA sequencing is also used in the field of malware detection.




A proposal is recommended to use Digital DNA technologies to identify malicious software proactively. The malware can be detected even if they are residing in the memory of Windows servers and workstations and cannot be detected using traditional security systems already in place. There is a claim in the proposal that the system can detect zero-day malware using Digital DNA using automated behavioural analysis and system memory analysis. The proposed system also identifies various low-level behaviours in every running program or binary code (HB Gary, 2018).

Figure 2.3 shows suspicious software modules and compromised computers with colour-coded alerts and threat severity score. Also, the behavioural traits of the samples are shown in Figure 2.4.

In the proposed system by HB Gary (2018), an image of physical memory is created by Digital DNA. Additionally, the system recreates all digital objects of the operating system and running applications after which the kernel and the operating system are examined to detect any malicious code.

Digital DNA Sequence	Module	Process	Severity	Weight
 0B 8A C2 05 0F 51 03 0F 6...	iimo.sys	System		92.7
 0B 8A C2 02 21 3D 00 08 63	ipfltdrv.sys	System		13.0
 0B 8A C2	intelppm.sys	System		11.0
 05 19 34 2F 57 42 00 7E 1...	ks.sys	System		-10.0
 02 21 3D 2F 1C FD 00 08 63	ipnat.sys	System		-13.0
 2F 7B ED	ipsec.sys	System		-15.0

**Figure 2.3: Product developed by HB Gary that indicates Ranking Software Modules by Severity using Digital DNA Sequencing (HB Gary, 2018). The image highlights suspicious software modules and compromised computers with colour-coded alerts and threat severity score.**

Trait		
	<b>Trait:</b> 8A C2	<b>Description:</b> The driver may be a rootkit or anti-rootkit tool. It should be examined in more detail.
	<b>Trait:</b> 3F 2E	<b>Description:</b> This driver may have hooking capabilities. Hooks are not always bad, but they are also a non-standard method that is common to hacking programs and rootkits.
	<b>Trait:</b> 9F E7	<b>Description:</b> The driver has a potential hook point onto the windows TCP stack. This is common to desktop firewalls and also a known rootkit technique.

**Figure 2.4: Product developed by HB Gary that indicates Behavioral Traits (HB Gary, 2018).**

## 2.7 Ransomware Detection Methods

This section explains some highly conventional and current ransomware detection methodologies.

### 2.7.1 Signature-based Detection

Pattern matching is the backbone of signature-based detection. Using a unique feature or a developed fingerprint is common in traditional antivirus products. The process involves capturing a sample of the malware, developing its signature then distributing that pattern to antivirus software. Antivirus software then scans files on the device looking for the extracted signature, and if found, the file is red-flagged and quarantined. The main issue with this method is that zero-day attacks or brand new malware cannot be detected. The delay until the malware's signature updates

are generated by antivirus firms to protect systems from infection can leave the door open for many victims to be infected. As a consequence, more specific detection techniques have been suggested. A massive repository of malicious code signatures is the foundation for signature-based detection. The repository has to remain current and has to be frequently updated. The update process is a time-consuming task, and in such cases, corporations which release commercial antivirus software usually have teams of cybersecurity investigators who are continually discovering, investigating and extracting malicious signatures. A significant disadvantage of this process is the presence of zero-day malware which does not yet have a signature. Malware is only detectable after it has been detected and marked as malicious and appended to the malware signature repository. Static-based detection efficiency is affected by three situations, as highlighted below (Nieuwenhuizen, 2017):

***Ineffective against code obfuscation***– Malware developers use code obfuscation methods to alter the malware to bypass signature-based detection so that each enhancement appears dissimilar to the previous one. Significantly, intended malicious behaviour is not affected by this code obfuscation, and it only impacts how a human or signature-based detection system statically perceives it.

***Ineffective against high variant output***– Malware types that have quick development cycles and different output are not influenced by signature-based analysis. New ransomware is being created much quicker than original signatures can be created, tested and added to the malware signature repositories. This rapid pace is challenging for signature-based detection systems.

***Ineffective against targeted attacks***– Ransomware is a field in which targeted attacks are happening more regularly. Developers of ransomware choose to first unleash new ransomware at selected target companies instead of mass distributing the same.

### 2.7.2 Honeypot

The use of a honeypot in a computer system refers to monitoring for unwarranted use of a resource by a program. For example, User Behavior Analytics (UBA) can be applied to identify abnormal behaviour of ransomware. Explicitly, recognising when a user modifies a large number of files in a relatively short amount of time can indicate the execution of ransomware (Jackson, 2016).

Researchers and technical experts have been using monitored honeypots to identify malicious code and its operations (Moore, 2016). Honeypots work by creating a system that includes information and data files that is of no meaningful use for the other applications or users. The initial implementations of honeypot were to attract malware and hackers and to deceive them by acting as baits and create confusion. A tracking mechanism is present in the honeypot to alert security administrators about an infiltration. Ransomware can also be detected using honeypots. As the operation of ransomware is to encrypt files, after infection the ransomware will also encrypt files existing in the honeypot, which can generate an alert for the security administrator of an attack in progress.

EventSentry is a tool available to monitor and create logs of real-time events on a system. The tool can further evaluate and monitor Windows Security logs along with the generated logs to raise alerts once suspicious and malicious actions cross a specific threshold. EventSentry can further create and monitor a folder made wholly of honeypots to detect unauthorised attempts to retrieve data from the folder. The usage of a single folder reduces possibilities of false positives as users are aware of the presence of the honeypot folder and know that its contents are fake, which does not need to be interfered with. Therefore, only malicious programs or hackers will target the said folder and alter its contents.

A detection approach that identifies different sorts of attempts to access the honeypot files exists in tandem with the observing process. The detection approach consists of a system of layered responses which lead to mixed reactions, associated with the methodology that is used to detect various attempts to access the honeypot folders. The layered responses include sending alert to the security administrator, indicating an attempt to access the honeypot folder. Then the IP address of the system that has triggered the attack to the honeypot, if existing locally, is blocked. Furthermore, network service can be disabled to avoid additional tasks related to the infection. Finally, a shutdown of the server is done to protect additional files of the server from being encrypted. If a user accidentally interacts with the honeypot folder, the existence of the layered response guarantees the least amount of issues for the user and the spread of the malicious activity is contained.

#### ***- Implementation of Honeypot***

For successful implementation of a honeypot system, a system that can observe modifications of identified files in a folder is required. A tool that can serve this purpose is filemon. A string is predefined and included in files of the honeypot and filemon actively observes these files. Filemon monitors changes made to any file that contains the predefined string, irrespective of the name or file type or the action being done to the file, which could be re-creation, deletion, altering or changing the name of the file. Furthermore, the tool can detect changes in the size of a file or its properties and act on the same.

Filemon is implemented with several thresholds to reduce false positives. The implemented system has been set up with a threshold that an alert will be generated only if two files are altered or impacted within a minute. The reason for this threshold to be chosen is that even if a user accidentally interacts with the honeypot folder, the user may not alter more than a single file. The

only reason for multiple files being altered with a minute is usually a malicious attack. Another point to consider is that ransomware creators may overcome this mechanism by creating slow-acting ransomware that encrypts only one file within a minute, but such ransomware is not effective in the real world as the goal of ransomware is to encrypt maximum files within the shortest period possible (Christensen and Beuscha, 2017).

### **2.7.3 Hashing**

A hash function is typically an algorithm that takes data input and produces an output value as a direct result of the input (Learn Cryptography, 2019). In some cases, these functions have a specified output length. For example, the text file of a book could be put through a hash function, and only a 20 character output is provided. The data throughout the input directly determine each output. If the same book was modified slightly, the 20 character output is different. The advantage of these functions is that they can be used to determine if data has been modified based on a change in the hashing functions output. In research by Scaife et al. (2016), hashing is used to monitor files and identify when they have been modified.

Additionally, through the use of similarity, preserving hash functions (Kornblum, 2006), an altered file can be analysed and inferred if encryption has occurred. Since encryption results in plain text that is unrelated to the original document, these hash functions can indicate whether a file was just altered slightly or entirely encrypted. The presence of ransomware can be indicated by using similarity preserving hash functions to monitor recently modified files; however, using hashing to find out when files have been altered incurs significant overhead in CPUs time for continuously calculating the hashes of files. Additionally, for large directories, it takes a substantial amount of time to calculate hashes of the stored data. As a consequence, the delay for recognising a modified file is significant and gives the ransomware to encrypt files which can lead to data files being encrypted before the ransomware is detected and stopped.

#### **2.7.4 Shannon's Entropy**

In the context data, Shannon's entropy is used to describe the amount of randomness, or disorder that is contained in a given sample of data (Paul, 2017). Specifically, this technique presents a value to represent the amount of information and randomness in a given data sample. For example, if a string contains ten characters, the entropy may suggest that the same amount of information can be expressed in eight characters. The reason files may have low entropy values is due to the repetition of the same data, and as a result that data can be expressed with fewer data.

An example is the string "thethe" which can be represented in fewer characters, perhaps with "3xthe," depicting three times that string. A low value of entropy suggests that the information in that file can be expressed using a smaller number of bits. An enormous value of entropy suggests that the raw data cannot be expressed with a smaller number of bits.

#### ***- Implementation of Entropy***

Shannon entropy value needs to be calculated for all folders and files on a system, and these values need to be stored for the Shannon entropy detection method to be efficiently implemented. Furthermore, tracking of when the file has been changed, created, renamed or deleted is required for Shannon entropy to detect tampering attempts. If a honeypot system is already monitoring files using filemon, additional protection can be enabled by using Shannon entropy to observe every single file present in the system. Similar to filemon, false positives can be avoided by implementing thresholds. Shannon entropy detection methods of various versions have different thresholds, and the thresholds are set such that each doubtful action is addressed, and a reaction is triggered. If a new file created has an enormous entropy, then the same is counted for the threshold. Similarly, the threshold is reached if a changed file's entropy is much higher than the original file's entropy.

A process of data analysis needs to be conducted to figure out how much entropy can increase before being marked as questionable. Entropies of all files in the folder are saved. Once the ransomware begins working to encrypt the data files, the entropy of the file are calculated. The entropies are separated into various categories depending on their sizes. The average of each category is then measured to detect the change in average entropy once the files have been encrypted.

A low Shannon entropy can be attained using a weak encryption method which can be broken easily. Ransomware creators can also use encryption methods that maintain a lower difference in the entropy values after encrypting the target files (Christensen and Beuscha, 2017).

In research by Shaukat and Ribeiro (2018), a process for monitoring the entropy of the data buffer is proposed as a way to identify encryption operations from ransomware. Specifically, this research looks for increases in entropy in the data buffer because encryption methods increase the randomness and disorder of a file. This method for detecting encryption depends on the entropy of a file increasing for detection, and as a result, will fail to detect encryption of a file already having a more significant amount of entropy. Files that are stored or compressed by the user have larger entropy values before the execution of ransomware do not yield detection with this method.

### **2.7.5 Machine Learning**

Past research has used machine learning to detect malware executables. The work by Kolter and Maloof (2006) tackled the issue as a text classification problem. With Decision Trees, Naïve Bayes and support vector machines, the researchers demonstrated that they could detect malicious executables. In another study, machine learning algorithms were applied to detect cryptographic algorithms from program binaries. A large number of encryption programs were generated using known encryption libraries and compiled into binary files. Using decision tree models, one can see



that an algorithm was presented to detect ransomware in work by Sgandurra et al. (2016). These methods have large overhead requirements in terms of computational power, which causes latency between infection of a device with ransomware and its detection. Besides, these methods have trouble with malware that uses compressed or encrypted payloads.

A machine learning method that can identify compiling of cryptographic algorithms has been proposed by researcher Hosfelt (2015). DES, SHA1, AES and MD5 are different algorithms that can be used. The proposed methodology can detect Crypto-ransomware infections and encryptions. Intel's Pin Dynamic Binary Instrumentation (DBI) framework is used to identify and separate features and is included in the proposed methodology. Code is injected by the proposed process into the malicious program to analyse the behaviour of the code during runtime. The malware can avoid detection if the code injection process is identified, thus disabling the execution of the malware program. The proposed process only checks for malware written using programming languages C++ and C, but future enhancements of the proposed system can be trained to classify and detect other programming languages.

In a document submitted by Kharraz et al. (2016), the researchers have analysed many functionalities of multiple ransomware variants and the interaction methodologies employed by ransomware in a Windows system. The system proposed by the researchers monitors API calls related to defragmentation and encryption libraries, among others. The main issue with the proposed system is that false positives are generated, as a valid standard program could use the same API calls and processes. To address this issue, the researchers have suggested training classifiers that can learn to differentiate amongst malicious ransomware and good programs. Additionally, the researchers have also suggested keeping track of changes occurring to the Master File Table (MFT) using machine learning algorithms, which can detect malicious alterations to the MFT (Kharraz et al., 2016).

## **2.8 Survey on Feature Selection Methods**

Feature selection is the process of grouping the most significant set of features which reduces dimensionality and generates the most analytical results. Choosing appropriate attributes are a significant issue for data reduction and competent classifiers. This section discusses some related work on feature selection methods for malware detection and also some other relevant applications.

A model has been proposed by researchers Ahmadi et al. (2016). The proposed methodology is used to classify malware into families and the features derived can be grouped based on various characteristics. Feature selection and extraction techniques have been opted over methods for classification in the proposed methodology. Structure and content of the malicious code is the base for extracting features. Due to this feature, obfuscated and packed ransomware variants can also be analysed using the proposed method.

A machine learning-based solution is proposed by researchers Kumar, Kuppusamy and Aghila (2017) that can classify a malicious code as malware or safe using low computer processing and high accuracy. Portable executable header fields' derived values and raw value have been combined to create an integrated feature set. Furthermore, in the proposed system, classification of malware was done using a combination of Naive Bayes, Logistic Regression, Decision Tree and Random Forest machine learning algorithms.

Two methodologies, Enhanced Semi-Random Subspace selection (ESRS) and incremental bagging (iBagging), have been proposed by researchers Al-rimy, Maarof and Shaid (2019). An ensemble-based detection system is proposed by the combined use of the two methodologies. Incremental subsets are created based on crypto-ransomware behaviour during its different attack phases in the iBagging method phase. A pool of classifiers was trained after constructing noise-free, optimal and

diverse features subspaces in ESRS. Lastly, the best combination of base classifiers was selected using a grid search, and the final decisions are made by majority voting.

Multi-Strategy Ensemble GWO (MEGWO) has been proposed by researchers Tu, Chen and Liu (2019) and solutions have been updated by using three different strategies for search. In the first strategy, the local search capability of GWO is enriched using the enhanced global best lead strategy. This method is attained by fully utilising the search space around the current optimal solution. In the second strategy, the one-dimensional update operation is input into the framework of GWO with an adaptable cooperative method to encourage global searchability and provide greater population diversity. Furthermore, finally, the disperse foraging strategy compels a part of search agents to discover a promising area depending on a self-adjusting parameter.

A new Grey Wolf Optimizer algorithm is proposed by Abdel-Basset et al. (2020) that can solve the feature selection for classification problem which depends on the wrapper methods. The solution proposed is combined with a Two-phase Mutation algorithm. The continuous search space is transformed into a binary format to address the feature selection issue, using the sigmoid function. The two-phase mutation enhances the development capacity of the procedure. The first phase deals with preserving high classification accuracy while reducing the number of selected features. The second phase increases classification accuracy by adding more useful features.

Gray Wolf Optimization and a swarm-based optimisation method were combined in research conducted by Emary et al. (2015). Data description with minor redundancy and maintaining classification performance were achieved using the proposed methodology by searching the space of features to find an optimal feature subset.

A hybrid self-adaptive cuckoo search algorithm Mlakar has been proposed by Fister Jr and Fister (2016). Three features have been added to the original cuckoo search to create the proposed

algorithm. The three enhancements are linear population reduction, a self-adaptation of cuckoo search control parameters, and a balancing of the exploration search strategies within the cuckoo search algorithm.

A correlation-based filter model was proposed by researchers Yamany et al. (2016) as a system for attribute reduction. In the proposal, the attribute space with minimum correlation among selected attributes is searched for by using the Cuckoo Search (CS) algorithm. Furthermore, the selected results are contenders for additional improvement towards the classification accuracy fitness function.

Two-archive Multi-objective Artificial Bee Colony Algorithm (TMABC-FS) is a multi-objective feature selection method proposed by Zhang et al. (2019). A collection of non-dominated feature subsets with proper convergence and distribution are attained using the operators; diversity-guiding search for onlooker bees and a convergence-guiding search for working bees. Moreover, the searching ability of various kinds of bees is improved using the external archive and the leader archive.

Researchers Arora and Anand (2019) have proposed binary variants of the Butterfly Optimization Algorithm (BOA). The proposed system, in a wrapper-mode, chooses the ideal feature subset for classification purposes. BOA is a newly proposed procedure that is still to be methodically used to solve problems related to feature selection. Optimal feature combination is selected by applying the two binary variants of BOA that minimises the number of selected features while maximising classification accuracy.

A hybrid feature selection approach has been proposed by researchers Jain and Singh (2018) to lessen the dimensionality of features expressively. The proposed method is appropriate for both

microarray and text datasets, which defines the best value of the threshold for the collection of non-redundant and relevant features.

A Hybrid Particle Swarm Optimisation with a Spiral-Shaped Mechanism (HPSO-SSM) has been proposed by researchers Chen, Zhou and Yuan (2019) for choosing the best feature subset for classification by employing a wrapper based method. Three improvements are made In HPSO-SSM. Firstly, the diversity in the search process is enhanced using a logistic map sequence. Secondly, the position quality of the next generation is successfully enhanced by introducing two new parameters into the original position update formula. Thirdly, a spiral-shaped process is implemented as a local search operator around the known optimal solution region.

A combination of modified Binary Ant System (BAS) and clustering has been used to create a hybrid filter-based feature selection algorithm, called FSCBAS, and is proposed by researchers Manbari, AkhlaghianTab and Salavati, (2019). The proposed algorithm can overcome high-dimensional data processing and search space challenges effectively. This model supports local and global search capabilities within and between clusters.

Tawhid and Dsouza (2018) presented a new Hybrid Binary version of a Bat and Enhanced Particle Swarm Optimisation algorithm (HBBEPSO) to solve feature selection problems. It combines the bat algorithm with its capacity for echolocation and can help to explore the feature space and enhanced version of the particle swarm optimisation with its ability to converge to the best global solution in the search space.

Researchers Zhang et al. (2018) propose a modification of the Firefly Algorithm (FA) for discriminative feature selection in regression and classification models. The proposed system uses data-based learning approaches to support decision making processes using. Furthermore, the

proposed system uses global and local promising solutions enhanced using Simulated Annealing (SA) to avoid the premature convergence issue present in the original FA algorithm.

## **2.9 Survey on DNA Sequencing**

DNA computing is a computational model that uses bimolecular data as information storage materials and biological laboratory experiments as information processing operators. The information, in DNA computing, is encoded by DNA sequences and the design of DNA sequences is crucial to successful DNA computation. For a set of DNA sequences to be useful in DNA computing, they must fulfil several combinatorial and thermodynamic constraints, which is challenging to be solved by the traditional optimisation methods. This section explains some DNA sequencing related work.

Arita et al. (2000) introduced a genetic algorithm into the DNA sequence design system and proposed a random generate and test algorithm (GA). The GA generator is used to estimate how critical each constraint is for the planned sequence design. The random generator is used to design sequences which satisfy the user's preference, such as presetting restriction sites or terminal sequences.

A method named template method is proposed by researchers Arita and Kobayashi (2002). The method steadily produces a set of sequences of length "l" such that any of its members will have approximately "l/3" disparities with other sequences, the overlaps of their concatenations and their complements.

A support method for sequence design in DNA computing has been established by researchers Tanaka et al. (2001). The technique minimises the evaluation function computed as the linear sum of the plural evaluation terms. The method offers a contribution ratio of each evaluation term to the evaluation function and searches for good sequences. The evaluation function present in the

method can decrease the number of combination of evaluation terms. The precise conditions for sequence design in DNA computing can be located using the proposed system.

A stochastic local search algorithm has been proposed by Tulpan, Hoos and Condon (2002) for the modelling of DNA codes. The method encompasses sets of equal length words over the nucleotides alphabet A, C, G and T, which fulfil definite combinatorial constraints. Another researcher Asahiro (2005), has discovered a set of good DNA sequences by applying established greedy methods.

A constrained multi-objective evolutionary algorithm has been proposed by Shin et al. (2005) to address DNA sequences optimisation for reliable DNA computing. The method is implemented into NACST/Seq, a DNA sequence design system, to help pick the best answers among many options. Khalid et al. (2008) used continuous particle swarm optimisation to solve the DNA sequence design. Guangzhao et al. (2007) proposed a DNA encoding algorithm based on PSO optimisation.

Exploratory research done by Wei et al. (2007), improved the Genetic Algorithm (GA) / Simulated Annealing (SA) algorithm and offered to solve the multi-objective optimise problem, using which the DNA sequence design system was developed. Qiu et al. (2007) designed a hardware microprocessor to discover the DNA code under thermodynamic restrictions. The work by Qiu et al. (2007), deals with the design, implementation and testing of a hybrid design that contains a hardware accelerator and a general-purpose microprocessor. The system can be used for hastening the detection of DNA under thermodynamic restrictions.

A taboo search algorithm was adopted and improved by researchers Zhang et al. (2008) for the systematic design of equal length DNA sequences. The proposed system can satisfy certain thermodynamic and combinatorial restrictions. The method helps to avoid trapping into local

optimisation using a taboo search algorithm and can detect a set of good DNA sequences that satisfy essential limitations.

Kawashimo et al. (2007), present a local search-based algorithm designing DNA short-sequence sets satisfying thermodynamic constraints about minimum free energy criteria. The researcher also proposes a dynamic neighbourhood search algorithm where an efficient search is achieved by dynamically controlling the neighbourhood structures for the sequence design under hamming distance-based constraints.

An approach based on the Invasive Weed Optimisation (IWO) algorithm has been proposed in work by Zhang et al. (2009) and has been developed to optimise encoding sequences. In the first stage, the mathematical simulations of constrained objective optimisation design for encoding problems based on the thermodynamic measures are set up. Afterwards, an improved IWO technique is created by defining the colonising behaviour of weeds to beat the problems of the original IWO algorithm.

A Quantum Chaotic Swarm Evolutionary Algorithm (QCSEA) has been proposed by Xiao et al. (2009) to address the DNA sequence optimisation issue. The chaotic search and the particle swarm optimisation are merged to propose a hybrid algorithm. The method proposed keeps the rapid convergence rate and evades the drawback of rapidly getting to the optimal local solution.

In the research by Zhang, Wang and Wei (2011), DNA sequences sets are generated and use the Minimum Free Energy (MFE) algorithm to assess the various combinatorial constraints. The practical method to influence the generation of the unexpected secondary structure of DNA sequences can cause mistakes. In work by Zhang and Wang (2011), the improved lower bounds of DNA codeword are founded for the capacity of DNA to encode information using a combinatorial model of DNA homology given by the so-called H-distance. Xiao, Zhang and Xu (2012) developed



a membrane evolutionary procedure based on crossover and mutation rules to optimise DNA sequences design.

A Dynamic Membrane Evolutionary Algorithm (DMEA) is proposed by Xiao et al. (2013) to solve issues related to DNA sequences design. ADE/PSO search strategy and the division and fusion rules of P systems with active membranes are combined in the method. Vector Evaluated Particle Swarm Optimisation (VEPSO) is employed by Ibrahim et al. (2011) to solve the design problem in DNA sequence design by minimising four objective functions, which are H-measure, similarity, hairpin and continuity while being exposed to two restrictions: GC content and melting temperature.

Researchers Xu et al. (2008), articulated the DNA sequence design as a multi-objective optimisation issue and answered it by Genetic Algorithm (GA)/ Particle Swarm Optimization (PSO). The solution incorporates PSO with the attribute of rapid convergence and GA with the attribute of convergence. A Modified Particle Swarm Optimization/Genetic Algorithm (MPSO/GA) is proposed by researchers Cui and Li (2010) to address the multi-objective optimisation issue. Suitable constrained conditions that DNA sequence need to fulfil are chosen and then the evaluation formulas of each DNA individual, matching the chosen limited terms are offered to find a better DNA coding sequence.

A model for Particle Swarm Optimisation (PSO) implementation to solve the DNA sequence design problem is proposed by Khalid et al. (2008). Social behaviour is simulated by PSO where associates of a group have a habit of following the leader of the group. The DNA sequences used to get results needs to be vitally designed to decrease errors that may come up during the computation process. A Population-based Ant Colony Optimisation (P-ACO) method has been

proposed by Kurniawan et al. (2009) to design DNA sequences set for Direct-Proportional Length-Based DNA Computing (DPLB-DNAC),

Ant Colony System (ACS) is proposed to address the DNA sequence design problem by researchers Kurniawan et al. (2008). The solution is based on Ant Colony Optimization (ACO) and is an enhancement of the Ant System (AS) that uses agents to get answers based on the pheromone present.

Researchers Wang et al. (2018) have suggested INSGA-II based on the standard NSGA-II for designing DNA codes. A constraint in the process of non-dominated sorting has been introduced in the algorithm. A combination of Bat and PSO algorithm is proposed by Liu et al. (2019) to design reliable DNA codes. A rank for codes has been allocated using fast non-dominated sorting. The algorithm is named BPSON.

## **2.10 Survey on Online/Active Learning Algorithm**

This section explains some related work on online/active learning algorithms. A methodology encompassing scalable and efficient machine learning algorithms that are used in large scale and wide range of industries and applications is defined as online learning algorithms. The applications of online learning are wide and varied, from machine learning, statistics to artificial intelligence due to their strengths of being fast, simple and lack of statistical assumptions.

Online learning algorithms are generalised classified into two categories; First-order based online learning and Second-order based online learning. The first category involves exploiting only first-order feature information, whereas the second category involves exploiting the first-order feature information as well as the covariance matrix of the feature information, which is considered to be the second-order information (Zhao, Hoi and Zhuang, 2013).

LIBOL put forward by Hoi, Wang and Zhao (2014), consists of state of the art online learning algorithm which is efficient and scalable in its applications that can be used to address substantial online classification tasks. LIBOL supports large scale online learning and is also an open-source library available for all for free.

Ruvolo and Eaton (2014) enhanced and developed an online algorithm based on the K-SVD (Aharon, Elad and Bruckstein, 2006) algorithm for sparse dictionary optimisation. The enhancement includes learning multiple consecutive tasks. The algorithm starts with a batch multi-task learning method that enhances K-SVD. The batch method is then extended to train models online in a lifelong learning setting. The algorithm proposed has lower computational complexity than other similar lifelong learning algorithms without compromising on performance.

The approach proposed by researchers Hayakawa et al. (2015) is used to design online mechanisms for multi-dimensional valuations associated with fewer costs. Cost and allocation function is not a significant criterion with this approach as long as a predefined set of conditions are met. Minor power consumption and multi-dimensional performance are considered to be a significant incentive to use this approach.

Predicting the value of a continuous variable can be done by using an online learning technique proposed by researcher Jahedpari (2015) by fulfilling the below tasks:

- (i) “integrate a set of data streams from diverse sources with time-varying configurations”,
- (ii) “integrating the results of several analysis algorithms for each data source when the most suitable algorithm for a given data source is not known from before”,
- (iii) “dynamically weighting the prediction of each analysis algorithm and data source on the system prediction based on their varying quality”.

Wan et al. (2015) proposed a new framework of learning to rank based on Content-Based Image Retrieval (CBIR). The purpose of the new framework is to learn from large scale training data in CBIR and achieve an ideal combination of different retrieval schemes. The framework consists of a group of online learning that can rank algorithms that are considerably more capable and scalable than traditional batch algorithms for comprehensive online CBIR.

Two novel online boosting algorithms are proposed by Wang and Pineau (2015), and they are intended to influence the knowledge of instances in other domains. The two algorithms have specific properties where one is used for transfer learning, and others are used for multitask learning.

Perceptron algorithm (Mohri and Rostamizadeh, 2013) is a popular algorithm in first-order based online learning algorithms. This algorithm updates the learner by subtracting or adding the misclassified instance with a constant weight to the existing set of support vectors.

Researchers Crammer et al. (2006) have proposed a simple online algorithm called Passive-Aggressive (PA) for online binary classification. The algorithm checks for the present instance, and any misclassification causes updates to the current model. It is generally seen that large margin algorithms generally outperform the Perceptron algorithm by examining the observed performance of first-order based online learning algorithms. However, one can observe the restricted performance of these large margin algorithms, as only the first-order information is adopted.

Second-order based online learning algorithms are designed to overcome the restrictions of first-order based algorithms. Additionally, parameter confidence information, also known as second-order information, has been explored to improve the performance of second-order based algorithms. To achieve this, researchers Cesa-Bianchi, Conconi and Gentile (2005) have enhanced

the classical Perceptron algorithm to propose second-order Perceptron. The proposed algorithm is analysed for its performance by putting it through the mistake bound model of online learning.

Researchers Dredze, Crammer and Pereira (2008) introduce a proposal based on confidence-weighted linear classifiers. In this proposal, parameter confidence information is added to linear classifiers. Furthermore, classifier parameters and the estimate of their confidence are updated in this proposed setting. The proposal further uses the covariance of the parameters to direct the update of each parameter and keeps a Gaussian distribution over the model parameters. An aggressive hard margin update strategy in noisy data is considered a disadvantage for the proposed methodology.

Crammer, Kulesza and Dredze (2009) have proposed AROW (Adaptive Regularization of Weights), an online learning algorithm that combines several useful properties including confidence weighting, large margin training, and the ability to handle non-separable data. The algorithm performs immaculately even if label noise is present due to the ability to perform adaptive regularisation of the prediction function upon seeing each new instance.

A Soft Confidence-Weighted (SCW) online learning methodology has been proposed by Wang, Zhao and Hoi (2012). The proposed methodology enables the conventional confidence-weighted learning method to handle non-separable cases. SCW online learning methodology has the following characteristics:

- (i) “considerable margin training,”
- (ii) “confidence weighting,”
- (iii) “capability to handle non-separable data, and”
- (iv) “adaptive margin.”

A unique characteristic of the second-order algorithms is that they can regularly perform better and converge faster than the first-order based algorithms during implementation. Unlabeled instances are sequentially selected by active learning to label them to reduce the effort needed to learn a good classifier.

The widely prevalent Passive-Aggressive algorithms have been adapted and enhanced by researchers Lu, Zhao and Hoi (2016) to propose a new generation of algorithms for active online learning called Passive-Aggressive Active (PAA) learning algorithms in an online dynamic learning setting. The strength of the PAA learning algorithms is that it not only uses the misclassified cases, similar to Perceptron-based algorithms, to update the classifier but also exploit correctly classified examples with low prediction confidence.

A new algorithm that implements pool-based active learning employing Support Vector Machines (SVMs) has been proposed by Tong and Koller (2001). When SVMs are used with Active learning, the need for labelled training instances is considerably reduced. Researchers Guo and Schuurmans (2008), have proposed a differential batch mode dynamic learning method that articulates the instance selection work as a continuous optimisation issue over auxiliary instance selection variables.

Sheng, Provost and Ipeirotis (2008) tries to solve the issue of repeated acquirement of labels for data items when the labelling is flawed. The research highlights repeated labelling approaches of increasing complexity and concluded with results as below:

- (i) “Repeated labelling can improve label quality and model quality, but not always.”
- (ii) “When labels are noisy, repeated labelling can be preferable to single labelling even in the traditional setting where labels are not particularly cheap.”

- (iii) “As soon as the cost of processing the unlabeled data is not free, even the simple strategy of labelling everything multiple times can give a considerable advantage.”
- (iv) “Repeatedly labelling a carefully chosen set of points is generally preferable.”

## **2.11 Ransomware Detection based on Machine Learning**

Over the years, various methods for ransomware detection has been proposed and conducted by using several machine learning techniques and frameworks. This section explains some machine learning techniques used for detection of ransomware.

Gangwar, Mohanty and Mohapatra (2017) proposed a system to detect ransomware by analysing patterns such as a dropped file, file paths, network activity and ransomware file properties. For the purpose of detection, different metrics were collected using exploit kits. Consequently, the payload of the application was analysed to check for goodware and malware codes. Finally, several algorithms, including Naive Bayes, Random Forest and J48 Decision Tree, were used in the classification of these samples.

A dynamic analysis tool has been proposed by Kardile (2017) to detect ransomware present in the user data by analysing the file system. Process Monitor software is used in the proposed methodology to monitor user activity and saves a registry with the file access rates. The analysis process includes parameters such as registries, network and process activity, among others. Furthermore, the tool captures execution screenshots and file system input/output accesses.

A tool named RanDroid has been developed by Alzahrani et al. (2018). The tool is to be primarily used on the Android platform and is a lightweight automated method for detecting ransomware variants. The tool operates by comparing and measuring the structural and operational comparison between a predefined threat data collected from known ransomware variants and a set of collected data which belongs to the application that needs to be analysed.

Researchers Sgandurra et al. (2016) have proposed a tool named EldeRan. This tool works on the principle of ransomware detection and classification dynamically by analysing operations such as registry key operations, calls from Windows APIs, and folder and file system operations, among others. Logical Regression classifier algorithm, a machine learning algorithm, is used by the tool for classifying each user application, and the tool has an additional function to identify and create signatures for unknown ransomware.

Additionally, suspicious behaviour is detected using an anomaly detection method in a proposal presented by Alam et al. (2018). The system proposed is named as RAPPER, and it gathers data from High-Performance Statistics (HPC). The tool takes into account Fast Fourier Transformation (FFT), and Artificial Neural Network approaches.

Most of the research conducted in the field of ransomware detects and analyses the behaviour of API calls. As part of this research, Lu et al. (2017) have used the V-detector negative selection algorithm in conjunction with a mutation optimiser to detect ransomware. The proposed methodology analyses API function calls, operations related to registers, networks and memory usage. Similarly, researchers Hampton, Baig and Zeadally (2018) worked on analysing API calls for Windows machines to detect and analyse fourteen ransomware families. The proposed methodology benchmarks healthy behaviours of these compared to ransomware behaviour, which is then used to identify specific API actions that are related to a ransomware infection.

A binary sequence of API calls is classified by researchers Maniath et al. (2017) in a proposal involving Long Short-Term Memory (LSTM). Cuckoo sandbox is used in the methodology to conduct feature extraction on the captured file operations, API calls, and values in the registry.

A proposal by researchers Zhang et al. (2019) detects ransomware threats and code by using a supervised machine learning approach. Other researchers Takeuchi, Sakai and Fukumoto (2018),



have proposed a system in which each Window API call is kept track of, and then the SVM algorithm is used to signify the vector model.

Additionally, researchers Chen et al. (2017) have proposed a system to analyse API calls with the help of data mining algorithms like Random Forest (RF) and Support Vector Machine (SVM), and others. The process of the proposed system includes phases as below:

- (i) “raw feature extraction,”
- (ii) “preprocessing,”
- (iii) “selection, and”
- (iv) “analysis through learning algorithms.”

Other varied proposals related to ransomware detection have used the honeypot concept to organise a trap. The honeypot mechanism was extensively used by the research conducted by Moore (2016) to detect ransomware and involved monitoring a honeypot folder which was set up to monitor Windows Events using a role named File Server Resource Manager (FSRM) File Screen. EventSentry monitoring tool was an additional tool used in this system to catch logs.

Another tool named UNVEIL is proposed by Kharaz et al. (2016) which analyses the end-user machines to detect a ransomware attack. The tool was tested in a lab environment and monitored by the researchers. During the testing, specific files on the test machines were monitored during a ransomware attack. Records were taken as screenshots to track the activities at three stages including before, during and after the ransomware attack. Records took into consideration also include timestamps of the files, pointers and changes in the input/output operations.

In another research, researchers Gómez-Hernández, Álvarez-González and García-Teodoro (2018) have suggested a honeypot set up as a mechanism to detect, analyse and respond to crypto-ransomware attacks. Advanced countermeasure like file locks is also in place against these attacks

in this system. Several other researchers have proposed to pay attention to the analysis or enhancement of current encryption technologies being used. In this aspect, researchers, Zavarsky and Lindskog (2016) used the Anubis and Cuckoo Sandbox to conduct a simple experiment to detect ransomware detection. In this setup, the experiment covers MD5 check, monitoring network communication, file and register activities, among others.

MD5 encryption technologies are used to check the antivirus engine that includes File Fingerprinting technique. Paybreak is a tool proposed by researchers Kolodenker et al. (2017), which includes enhanced file encryption. PayBreak uses an asymmetric key pair that is stored in a secure key vault. The system works in such a way that the public key is configured with the user instructions and the private key is stored in a safe place. The vault is accessed with the private key if the device is attacked by ransomware,

Researchers are also looking at other methods to detect ransomware attacks. One of these methods is to monitor the communication between the attacked machine and the C&C server that control the ransomware.

In this aspect, a tool named NetConverse (Alhawi, Baldwin and Dehghantanha, 2018) incorporates a machine learning methodology to detect ransomware attacks. The proposed tool from the researchers monitors network traffic including IP address, protocols, packets, ports, among others, using a tool TShark, a network analyser tool. Similarly, researchers Cabaj, Gregorczyk and Mazurczyk (2018) presented a ransomware detection methodology that is based on the concept of Software Defined Networks (SDN). The concept presented by the researchers addresses communication from the ransomware using the HTTP protocol. The first ransomware which uses HTTP traffic is Locky and CryptoWall. SDN controller monitors the traffic between the potential

internal devices which could be infected and the C&C server and takes appropriate action in case of any suspicious activity.

Another researcher, Pillai et al. (2018), has applied the SDN concept to detect ransomware. Asymmetric encryption technologies are taken into consideration by these researchers to track the behaviour of crypto locker ransomware. SDN takes into account the encryption time and performs an appropriate action to address the threat from the ransomware.

HelDroid is a detection system for the android platform ransomware proposed by researchers Andronio, Zanero and Maggi (2015). The system uses code characteristics, including application manifests and call functions to identify ransomware and its classification and uses Natural Language Processing (NLP). Common messages that appear in ransomware code is the basis for this system and training of the tool is done on these messages. However, the text classifier represents the main weakness of HelDroid.

A practical method is proposed by researchers Song, Kim and Lee (2016) to detect and prevent modified ransomware from attacking the Android platform. The proposed system has a very high and fast detection rate as the tool is designed in such a way to be implemented in the android source code rather than as an external mobile application. The proposed system is powerful as it can detect ransomware even if the pattern is not available and additionally detect varied patterns of ransomware. The backbone of the system is the methodology used by the code where the processor, memory usage and I/O rates are monitored to detect abnormal behaviours. In case of any discrepancy being detected, the system takes immediate action by stopping the process and takes an additional action of deleting the program associated with the process.

Another tool proposed by researchers is named EldeRan (Sgandurra et al., 2016). The tool analyses installation activities and processes to dynamically detect ransomware using machine learning.

EldeRan checks characteristic signs of ransomware by observing a set of actions occurring during the initial phases of installation. Mercaldo et al. (2016), have proposed a parser. The process includes a three-step process to automatically identify related ransomware instructions by analysing sample code and detecting the associated Android ransomware family. Code Metamorphism is a technique used to evade detection that is similar to other static malware detection approaches.

Another researcher used deep learning to detect ransomware (Tseng et al., 2016). A deep neural network was constructed and used to train the perceptron with critical payloads selected from packets which were extracted from real network traffic. This model is also a feasible and practical approach to zero-day variants and categories of ransomware which can complement existing antivirus technologies.

NetConverse is a tool put forward by researchers Alhawi, Baldwin and Dehghantanha (2018). The tool is used on Windows machine systems and uses machine learning analysis to accomplish a tremendous, consistent detection rate. NetConverse uses various machine learning classifiers and samples of features extracted from network traffic conversation for ransomware detection.

Researchers Kharaz et al. (2016) and Scaife et al. (2016) have proposed two different ransomware detection methods which are called UNVEIL and CryptoDrop, respectively. Both proposed systems look at the file system to detect ransomware activity. However, they both lack recovery from any ransomware attack and do not have a process to identify cryptographic primitives. Text analysis techniques, similar to the work done by Andronio, Zanero and Maggi (2015), is included in UNVEIL. The method looks at the ransom demanding notes and messages on the locked screen. Whereas, the Hash function is used by CryptoDrop to detect the difference between the encrypted and original file.

A similar approach was suggested by researchers Baldwin and Dehghantanha (2018), in which circumstantial opcodes were analysed to detect the execution of crypto-ransomware. The method proposed by the researchers consists of building a unique model which included benign and malicious binaries using SVMs and static analysis. The method contributed by building a model of 443 unique opcodes and displayed high accuracy in the discrimination of the opcodes. The process was successful in proving that efficient discrimination can prevent specific instructions with similar densities. However, a drawback of the system was that this distinction required dominant manual interpretation.

A unique proposal from researchers Zhang et al. (2019) has approached the issue of ransomware by defining a static analysis method to detect and classify ransomware. The methodology proposed consists of several steps where initially N-Gram sequences are created from opcode sequences of ransomware samples. Consequently, Term Frequency-Inverse Document Frequency (TF-IDF) is calculated for each N-gram. The TF-IDF is a base value used to select feature N-grams so that these N-grams display improved discrimination amid families. In the final step, feature vectors are derived by treating the vectors composed of the TF values of the feature N-grams. Afterwards, the output is provided to five machine-learning approaches that complete the ransomware classification.

Another set of researchers Hodayoun et al. (2019) have used the Softmax algorithm for classification, Convolution Neural Network (CNN) and Long Short-Term Memory (LSTM), and a combination of two deep learning techniques which contribute to creating Deep Ransomware Threat Hunting and Intelligence System (DRTHIS). The DRTHIS system can be used to differentiate ransomware from goodware and identify ransomware families.

A tool named RansomWall has been proposed by researchers Shaukat and Ribeiro (2018). The tool is considered to be a layered system available to protect against crypto-ransomware. The system has been developed after analysing a vast dataset of ransomware families. Dynamic and static analysis methodologies have been combined in this proposed system to produce a unique compact set that highlights the ransomware behaviour. The system has been advocated to detect zero-day ransomware and a layer named Strong Trap Layer proposed in the system aids in the detection of ransomware at early stages. The system works by classifies any suspicious activity detecting by the initial layers present, and if any files are found to be modified by the rogue process, the files are backed up for protecting the user data until the rogue process is classified as dangerous or not.

In the current scenario, there is a variety of anti-ransomware methods or tools available to address the threat from ransomware. The tools need to have characteristics to detect suspicious behaviour, prevent a ransomware attack before it happens and mechanism to recover from a ransomware attack. Unfortunately, not all tools are successful in efficiently accomplishing their tasks against new threats. Another vital point to note is that most of the anti-ransomware tools can detect and block ransomware only if the ransomware code has executed at least once in the target system. The detection process is done through behavioural and forensic analysis (Gonzalez and Hayajneh, 2017). There is no solution available that can detect ransomware before the code executes on a target machine.

## **2.12 Ransomware Detection mechanism from web**

This section explains some detection mechanism or tips from the web for ransomware detection.

Comodo (2019) gives four tips to detect ransomware:

- A technique used by ransomware creators would be to use phishing methodologies. An end-user needs to be careful scrutinise the “from” address of any received email address

as a person with malicious intent would try to disguise the email address made to look like a valid known email address. Techniques used could include replacing the letter “l” with a numeral “1” to trick the receiver.

- The other way to detect ransomware attacks is to scrutinise the email content carefully. In earlier days, malware creator would not spend much time to create professional-looking email content. Nevertheless, that is not the case with most attacks now. The malicious user tries to duplicate professional emails that look as if coming from a real entity and tries to trick the user into clicking the link present on the email. These are reasons why careful scrutiny of email and if in doubt a call to check if the sender actually sent the email is absolutely necessary before clicking on any link present in the email.
- The next thing to look for to prevent ransomware attack is to look at the link that needs to be clicked. A malicious person will send a link that may look like an authentic link. Carefully analyse the provided link to check for spelling errors and mistakes in the domain name. If one has any doubts, do not click on the link as ransomware may be automatically downloaded on the user's system.
- Caution needs to be taken while downloading attachments. Malicious files can be hidden inside compressed files like ‘zip’ or ‘rar’. Also, malicious code can be present in ‘pdf’ files or as macros in office files. Ensure that the downloaded files are not downloaded onto a machine that is mission-critical if the user has any doubts.

Murphy (2018), suggests six security tips for ransomware detection

- **Update the Softwares and Applications:** It is of paramount importance to update all application and software for known bugs as soon as the software creator releases n update. The update process applies to all devices, be it a mobile device or traditional

PC used by the end-user. Most of the recent ransomware attacks have occurred by exploiting the known vulnerabilities present in software which have not been patched including the recent WannaCry attack which used a vulnerability in the SMB protocol found in Windows systems.

- **Conduct Frequent Security Scans:** The user needs to have a comprehensive antivirus and malware protection tool installed on their system. Also, the user should conduct periodic scans of his system using the installed software to detect any potential malware hidden and lying dormant in the system. Just having a protection system is not enough as new variants may not be detected if the signature files are not updated regularly.
- **Install Active Monitoring Solutions in the network:** The presence of a host-based protection system is not enough from an organisation perspective. The organisation needs to install a system-wide protection and security monitoring solution to protect all end devices in the organisation. The network protection system would also actively monitor network traffic to identify potential malicious network transfer. The advantages of having such a system include protecting open wifi networks, and inhouse security professionals can tap into the network to monitor the network traffic.
- **Create Fake Servers or Honeypots:** Honeypot is an efficient method to implement security for a production network. Honeypots are set up to attract potential hackers and distract them away from the production network. The honeypots consist of fake file and folders that are designed to attract and mislead malicious intent users. While the malicious person is busy interacting with the honeypot files, alarms can be generated to intimate the system administrator about a potential attack. Honeypots are also efficient in protection against ransomware attacks where the effects of a ransomware attack can



be seen on the fake files. The encryption process of the ransomware can be observed to look of a possible solution in case of an attack on the production servers.

- **Use Awareness and Education:** Almost all success attacks worldwide occur due to user negligence. User awareness and education is an essential cog in the security wheel. Any security measure at an organisation is only successful if a proper users education plan is put in place. Users need to be trained about the reading and understanding non-authentic emails, concepts of social engineering, password protection, among others. Almost all security and ransomware attacks can be prevented to an extent by educating the users on proper security procedures. The users have to know the importance of their data and measurements have to be taken by them to protect the same. Additionally, the management of an organisation should have a top-down approach to security awareness training, where users have to attend these training as mandatory.
- **Procure Insurance against Cyber Attacks:** Cyber-attacks and exclusively ransomware attacks are primarily targeted to extract money from the end-users. Alternatively, malware attacks can bring down the systems of an organisation and may cause data loss and system downtime. The attack can lead to a loss in revenue for the organisation. Insurance is available against these types of attacks to protect end-users and organisations. These insurances cover the organisation's liability and reduce the impact of the ransomware attack on organisations. The insurance typically covers data loss of personal identities, repairing of damaged systems, notifying customers and legal fees. It is advised to invest in insurance which will protect the organisations against cyber-attacks and especially ransomware attacks.

Raman (2018) suggests seven ways to protect against and detect ransomware beyond antivirus.

- **Network Shares/folders need to be secured:** Security policies have to be put in place to ensure permissions at folder level are not compromised. The most basic operation any malware is to spread itself through the organisation. The spreading happens only when devices and folders are found that can be accessed by the malware. By default, no folder should have full access rights for the “Everyone” group. Close monitoring is needed on the network to ensure that folders or systems do not have open-ended permissions. There are several tools available in the market that can audit, detect and inform the system administrator in case of any discrepancies.
- **Audit Service Usage:** Unused services in any system need to disable or stopped. By default, several background services are running on a system which may have vulnerabilities and could be used by malicious programs to penetrate the system. Security audit tools are available in the market that can detect such services that are running on a system and recommend actions on them. Disabling services on a machine to protect is called as device hardening.
- **Detect Unknown C&C Accounts:** Mostly, the first activity of any malware after penetrating a system to create a backdoor for persistence. Backdoors are user accounts with elevated system privileges. Any action from an authorised user in the system goes undetected by security software present on the system as these are authorised actions. The security audit tool is required to periodically check for such account and take appropriate actions on them.
- **Detect Rogue Browser Plugins:** Most of the ransomware enter into a system through a browser which provides a gateway to the Internet. Many times malicious plugin are present on the browsers that disable security measure put in place on the system. User browsing on the Internet may be lead to fake websites by these bowser plugins and

prompts the user to install dangerous malware unknown to the user. Even though security measure is in place on most modern browsers, external tools can help to scan the end-user system to detect malicious browser plugins and centralised security policy can implement removal of such plugins.

- **Monitoring Network Traffic:** There is much traffic that flows between devices within the organisation and outside the organisation. Several hardware and software devices are available including Firewall, IPS, WAF, NetFlow and Proxy, which can monitor the traffic coming in and leaving an organisation. The devices use Threat Intelligence databases to cross-verify network traffic content. The combination of network security devices and Threat Intelligence helps to detect and prevent malicious network traffic.
- **Usage of Indicators of Compromise:** Indicators of Compromise (IOC) are features used to identify malicious activities of programs or codes. These properties can be potent in situations where anti-malware and antivirus signature are outdated. Usually, when a zero-day, malware or virus is released, the end-users are at risk of getting infected as the anti-malware or antivirus software installed on their machines may not have the latest signatures. The delay is because companies producing such software will have a delay in creating the required signatures. Consequently, for full protection, every end-user system should have an IOC based software for conducting IOC scans in addition to signature-based scans.
- **System to Drive-by Download detection:** Network security devices such as a proxy server, DNS logs and NetFlow have built-in measures to detect drive-by downloads. Drive-by downloads occur when malicious code is downloaded automatically onto a machine when a user visits a website. Additional tools are available in the market that can help to detect malicious code in websites that can disable drive-by download.

### **2.13 Summary**

This chapter reviews the related work of ransomware detection. It outlines the topics of ransomware and their families. It explains a similar survey for the problem of feature selection, DNA sequence and active online learning. It also describes and presents a detailed study of various ransomware detection techniques using machine learning.

## **Chapter 3:**

### **Preprocessing and Feature Selection**

#### **3.1 Introduction**

Preprocessing is a process conducted on data during the data mining techniques to convert raw data into a comprehensible user format. Data from real-world usually has missing values, consists of noisy values, and is generally incomplete, inconsistent and include outlier information. Hence, raw data need to go through a preprocessing step before being mined for useful information and going through this process enhances and ensures data efficiency. Data preprocessing is an essential phase of data mining that involves data preparation and transformation of the dataset and pursues to make knowledge discovery more efficient. Several techniques like cleaning, integration, transformation, selection and reduction are included in Preprocessing.

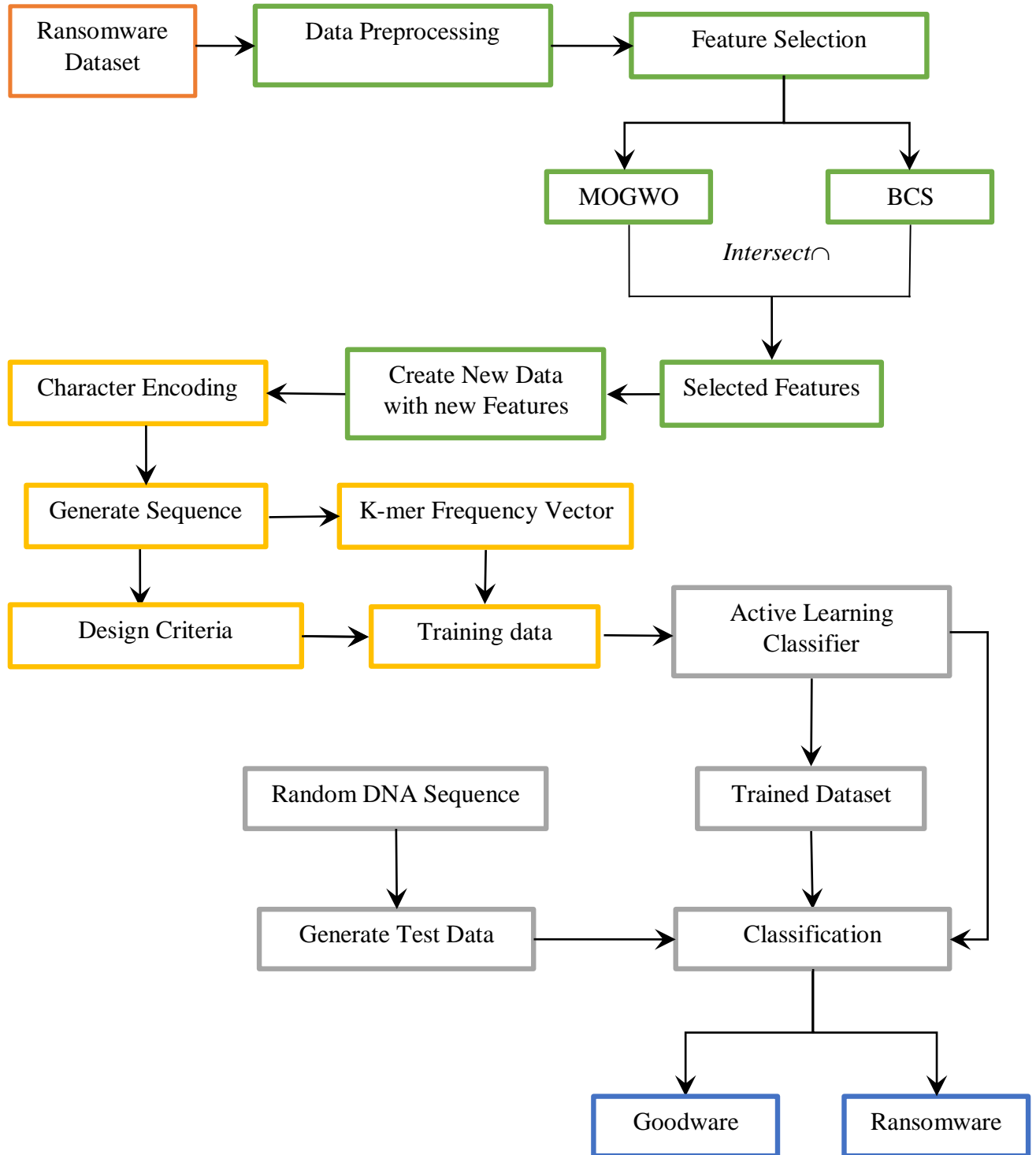
Feature selection is a process used to build a much clearer and understandable model. Feature selection also contributes to improving data mining performance and generates clean, comprehensible data. Feature selection is considered to be a data preprocessing strategy that is used in different machine learning and data mining issues and is proven to be effective and efficient in preparing data, especially concerning high-dimensional information. Feature selection is also used to identify a subset of the most beneficial features which outputs comparable results as the original entire set of features.

This chapter explains the proposed feature selection algorithm for ransomware detection. Before applying the feature selection, data preprocessing technique is applied to the dataset. The preprocessing of data includes remove missing value records and remove columns that are having the value as zero. The feature selection uses Grey Wolf optimisation and a Binary Cuckoo search algorithm for selecting the best features from the dataset.

This chapter presents in detail a preprocessing and feature selection method for ransomware detection. The overall system architecture is presented in section 3.2. Section 3.3 explains different data preprocessing and its methods. Section 3.4 explains feature selection and related work on feature selection. The proposed feature selection for ransomware detection is described in section 3.5, and it includes Grey Wolf Optimisation and binary search feature selection algorithm. Section 3.6 provides screenshots and sample code used in the developed product, and finally, section 3.7 gives a summary of this chapter.

## **3.2 Overall System Architecture**

Figure 3.1 shows the overall system architecture for ransomware detection.



**Figure 3.1: An outline of the Process Diagram highlighting the overall System Architecture. At the beginning of the process, the ransomware dataset is taken and pre-processed. The pre-processed dataset is further used for feature selection and then consequently a new dataset is created. The dataset is used to create training data and an active learning classifier is used to conduct the final classification to either ransomware or goodware.**

### 3.3 Data Preprocessing

Data preprocessing is one of the essential data mining tasks, in which data is transformed and prepared into an appropriate form for the mining procedure. Data preprocessing aims to reduce the data size, find the relations between data, normalise data, remove outliers and extract features for data. It improves the quality of data and, consequently, of the mining results. Raw data is preprocessed to enhance the efficiency and ease of the mining process. Figure 3.2 shows the data preprocessing tasks (Alasadi and Bhaya, 2017).

Data preprocessing methods are divided into four categories:

- Data Cleaning
- Data Integration
- Data Transformation
- Data Reduction

#### 3.3.1 Data Cleaning

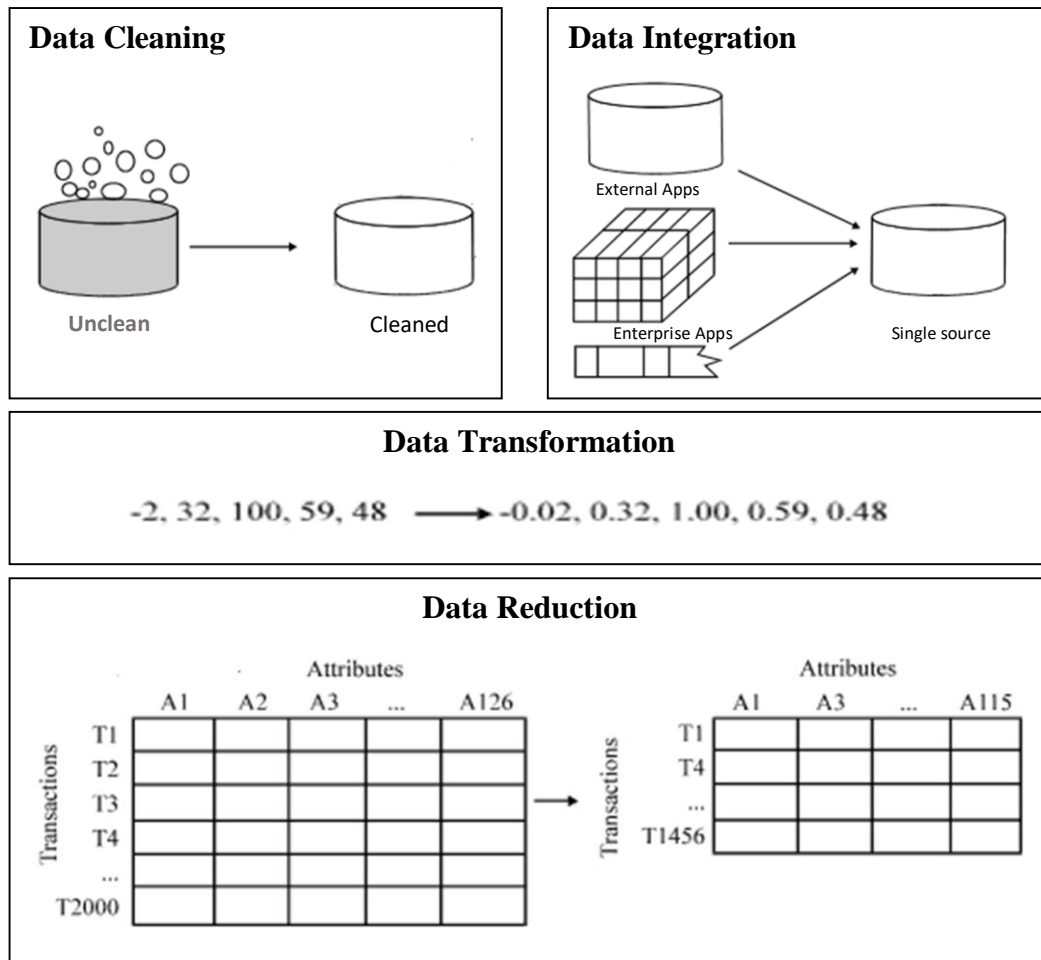
Raw data may have incomplete records, noise values, outliers and inconsistent data. Data cleaning is the first step in data preprocessing techniques which are used to find the missing values, smoothen noisy data, recognise outliers and correct inconsistent data. The data without preprocessing can lead to an unreliable and inadequate output. Therefore some data cleaning routines need to be used.

**Missing Values:** If there are records with missing values, then these values have to be filled using the following ways:

- *Ignore the tuple:* This method is selected when the value of the class label does not exist. This method is not practical, but it is used when the tuple has several attributes with empty values.



- *Fill the missing value manually:* This approach, in general, requires human effort and is time-consuming. It cannot be used with datasets of large sizes.



**Figure 3.2: Various process and steps involved in Data Preprocessing**

- *Use a global constant to fill the missing value:* The approach works by changing missing values of an attribute by a particular constant which is the same for all records, for example, “unknown” is used as a label. The issue with this method is when the missing values are replaced by a specific term “unknown” as an example, the mining programs may believe that they form an essential concept since they have a standard value.

- *Use the attribute mean to fill the missing value:* The approach works by replacing the missing value for a particular quality using the attribute's average value.
- *Use the attribute mean for all samples belonging to the same classes as the given tuple:* This method can be explained using an example. Classify user depending on credit risk; the missing value can be replaced by the average value of income for the users which belong to the similar credit risk class for a given tuple.
- *Use the most probable value to fill the missing value:* The method is used with a technique such as inference based regression using a decision tree induction or Bayesian formalism.

**Noise Data:** One of the most significant problems which affect the missing process is noise. A random error or change in a measured variable is called as noise. The meaning of noise data is that there is an error in data or outliers which deviates from the normal. The issue can be corrected using the following methods

- *Binning:* This approach works on smoothing stored data based on its neighbourhood data, which are the data values around it. Since these approaches depend on neighbour's data, they perform local smoothing. The sorted values are divided into several buckets or bins.

The lowest and most significant data values in each bin are indicated as bin boundaries during the process of smoothing by bin boundaries. Furthermore, each value is substituted by the nearest boundary value. Generally, whenever the width of the bucket is larger, the effect of smoothing is more significant. Consequently, binning is used as a discretisation technique in case of equal width bucket wherein the interval range of values in each bucket is the same,

- *Regression:* This approach smoothen's data by fitting it to a function. Linear regression, as for an example, includes determining the best line to fit two attributes such that each attribute can be used to predict the other. Using regression to fit data by finding a mathematical equation may be used to smooth the noise data. Linear regression is further expanded to create Multi Linear regression. It involves two or more variables, and hence, data is fitted to a multidimensional domain.
- *Clustering:* This approach can detect outliers as the method group similar points into a cluster while points that are outside the clusters are referred to as outlier points. Grouping a set of points into collections according to a distance measure is called clustering. The result of clustering is set if each cluster will have a set of points with a small distance from one another and with a considerable range from other clusters.

### 3.3.2 Data Integration

This technique works by combining data from multi and various resources into one consistent data store, like in a data warehouse. These resources can have multi-database, files or data cubes. In data integration, there are several issues for consideration, like schema integration, object matching and redundancy, which are an essential aspect. Each attribute, like annual revenue, is considered to be redundant if the same is derived using a different attribute or a set of attributes. Inconsistency in the attribute is another form of redundancies. Redundancies can be detected using correlation analysis. The correlation between two variables can measure how the characteristics can imply one the other sharply. The relationship between (X, Y) attributes can be evaluated by finding the correlation coefficients (Dharmarajan and Vijayasanthi, 2015).

### 3.3.3 Data Transformation

Data Transformation process includes transforming the data to forms suitable for the mining process. It involves the following methods (Baskar, Arockiam and Charles, 2013):

**Smoothing:** It removes noise from data. It includes techniques such as clustering, regression and binning.

**Aggregation:** It is the process of applying statistical metrics like means, median and variance, which are necessary to summarise the data. The resultant aggregated data are used in data mining algorithms. For example, apply aggregation on daily sales to compute monthly and annual sales.

**Generalisation:** It includes replacing the lower-level data by higher-level using hierarchical concepts. An example, street which is a type of categorical attributes may be replaced to city or country which is high-level terms. Another pattern, age, which is a type of numeric concepts, can be mapped to senior, younger and youth, which are essential.

**Normalisation:** This method works by adjusting the data values into specific ranges, such as between 0 to 1 and -1 to 1. This method is useful for mining techniques like classification, artificial neural networks and clustering algorithms. Using the normalisation to scale the data attributes in the training phase for the backpropagation neural network algorithm can be used to speed the learning stage. Minimum-Maximum, Z-score and decimal scaling are popular forms of normalisation.

### 3.3.4 Dimensionality Reduction

These techniques can be used to reduce the representation of dataset in smaller volume while maintaining the integrity of the original dataset. Thus a better data results can be obtained from applying mining techniques to reduce data. The following section explains data reduction methodologies (García et al., 2016).

**Data cube aggregation:** This approach constructs a data cube by applying operations of aggregation on data without losing information necessary for data analysis.

**Attribute subset selection:** It reduces the dataset size by removing redundant features or dimensions and irrelevant attributes.

**Dimensionality reduction:** It reduces the size of the dataset. Principal Component Analysis (PCA) is an example of dimensionality reduction.

**Numerosity reduction:** It replaces or maps data to an alternative or smaller representation of data. It consists of parametric and non-parametric models. The first model needs to store only the parametric of the model. It includes techniques such as sampling, histogram and clustering.

**Data discretisation and concept hierarchy generation:** These techniques can be used to replace the attributes data values by a high level of conceptual or interval ranges. It is a type of numerosity reduction, which is very useful for the generation of hierarchal data automatically. One of the essential tools of data mining is hierarchical and discretisation, which performs the extraction in multi abstraction levels. Data discretisation can be classified based on how it performs supervised or unsupervised discretisation if it uses class label; otherwise, it is top-down or bottom-up discretisation.

### 3.4 Feature Selection

The process of eliminating the redundant and irrelevant features from a dataset to improve performance of classification or clustering algorithm is called as feature selection. Feature selection methods can be classified based on two criteria: the search strategy and the evaluation criterion.

Many application areas that relate to expert and intelligent systems use the feature selection methodology. Examples of such application areas consist of image processing, data mining and

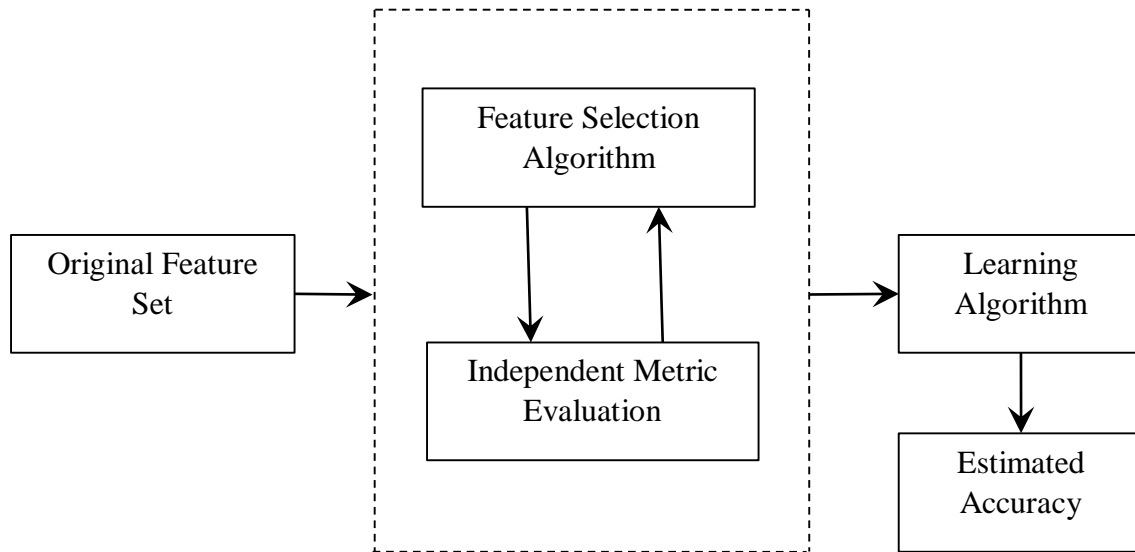
machine learning, natural language processing, anomaly detection and bioinformatics. Feature selection methodologies are applied at the pre-processing data stage before a classifier is trained and directly selects a subset of applicable features for model construction. Variable selection, feature reduction or variable subset selection are terminologies used in tandem with the term feature selection (Bennasar, Hicks and Setchi, 2015). Real-world data like tweets and blogs, contain a lot of irrelevant, redundant and noisy information along with relevant data. Feature selection is used to removing irrelevant features and reduces storage and computational cost. The benefit of feature selection is that a significant loss of information can be avoided and the degradation of learning performance is not encountered. Feature selection methods are divided into three categories based on the evaluation strategy. The categories are explained in the following section:

**Filter methods:** Filters are generally much faster and function with dependency on the induction algorithm. The selected features derived using the filter methods can be passed to any modelling algorithm. Filtering measures such as statistical measures, information, dependence, distance, similarity and consistency are some of the measures that can be used to classify filtering measures roughly.

Figure 3.3 depicts the functionality of the filter approach in the feature selection. It encompasses the feature selection and its evaluation in one component. In the filter approach, the individual features are analysed and evaluated. Based on this evaluation, the features are selected for the data mining task such as classification.

The concepts of Fisher Score, mutual information and ReliefF has been used by researchers Hancer, Xue and Zhang (2018) to propose a new filter criterion. The proposed criterion chooses the highest-ranked features determined by ReliefF and Fisher score while providing the mutual relevance between the class labels and features, rather than using mutual redundancy. The proposed criterion

further develops two new Differential Evolution (DE) based filter approaches. One approach uses this criterion in a multi-objective design, and the other approach considers this criterion as a single objective problem in a weighted manner.



**Figure 3.3: Functionality of the Filter Approach in Feature Selection. The feature selection and its evaluation are encompassed in one component.**

A mutual information-based algorithm has been proposed by researchers Ambusaidi et al. (2016). The proposed algorithm classifies by analytically selecting the most optimal feature. Linearly and nonlinearly dependent data features are handled by the proposed feature selection algorithm. Additionally, researchers by Zhao et al. (2018) have proposed a new mutual information algorithm of the Redundant Penalty between Features (RPFMI) algorithm that can select optimal features. The proposed algorithm considers the following factors; the relationship between candidate features and classes, the impact between selected features and classes and the redundancy between features.

Another method is proposed by researchers Osanaiye et al. (2016). The proposal is an ensemble-based multi-filter feature selection method that achieves an optimum selection by combining the

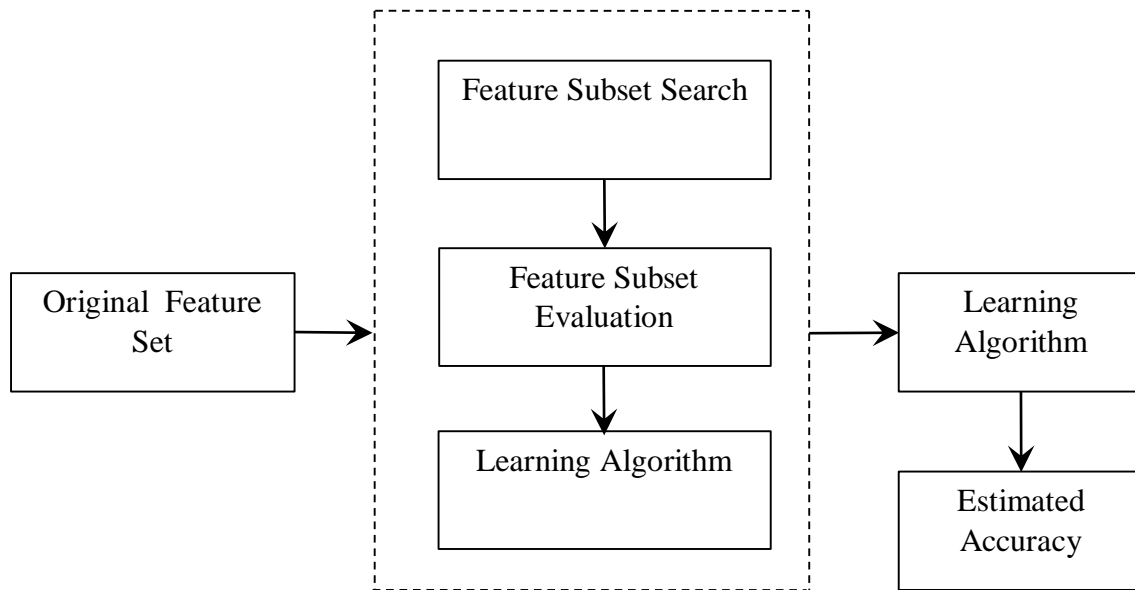
output of ReliefF, Information Gain, chi-squared and gain ratio. Researchers Labani et al. (2018) have further proposed another filtering method for feature selection, named Multivariate Relative Discrimination Criterion (MRDC). The methodology uses maximal-relevancy and minimal-redundancy concepts to focus on the reduction of redundant features. Additionally, the methodology accounts for a correlation metric to accommodate redundancy, and maximum relevancy features are also selected.

**Wrapper Methods:** a predictive model using a candidate feature subset is trained by stand-alone modelling algorithm. The measurement of the feature set is derived by testing the performance of a hold-out set. Alternatively, estimated feature importance scores can be applied to select a feature subset in a modelling algorithm like the random forest. Atypical of any wrapper method, a new model must be trained to test any subsequent feature subsets. As the wrapper methods are naturally iterative and computationally intensive, they can identify the best performing features set for that particular modelling algorithm. Consequently, the feature subset is generated based on the selected search strategy at each iteration of the wrapper.

Figure 3.4 shows the functionality of the wrapper feature selection process that encompasses three in one single component. Feature subset search, evaluation and learning algorithm are performed under a single unit.

Researchers Mafarja and Mirjalili (2018) have used the Whale Optimization Algorithm (WOA) and proposed a new wrapper feature selection approach. Optimal feature subsets for classification purposes are explored for by using two binary variants of the WOA. The first variant studies the influence of using the Tournament and Roulette Wheel selection mechanisms rather than using a random operator during the search process. In the other variant, mutation and crossover operators are used to improve the usage of the WOA algorithm.



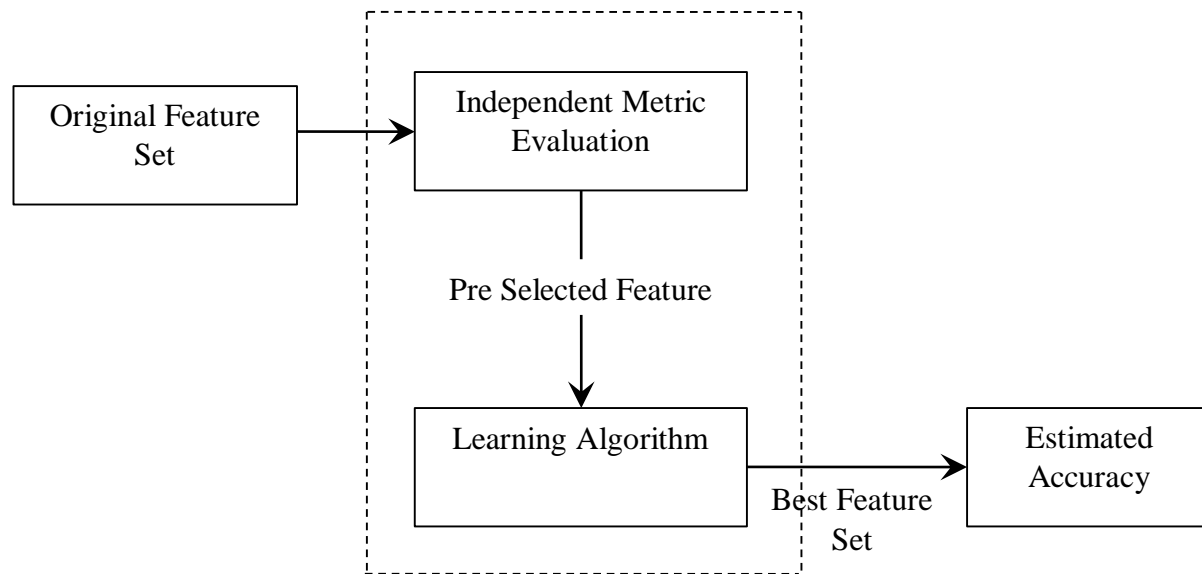


**Figure 3.4: Functionality of the Wrapper Approach in Feature Selection where Feature subset search, evaluation and learning algorithm are performed under a single unit.**

Information Gain Directed Feature Selection algorithm (IGDFS) is a unique approach to feature selection proposed by researchers Jadhav, He and Jenkins (2018). The proposed approach is used in credit scoring applications. The proposed algorithm additionally propagates the top  $m$  features through the GA Wrapper (GAW) algorithm using three classical machine learning algorithms of KNN, the ranking of features based on information gain, Naïve Bayes and Support Vector Machine (SVM) for credit scoring.

A conventional algorithm is used as a basis for research conducted by Khammassi and Krichen (2017). The researchers have applied a wrapper approach as a search strategy and logistic regression. The proposed learning algorithm can be used to select the best subset of features from network intrusion detection systems. Another set of researchers Chen and Chen (2015) have introduced another wrapper method which is named Cosine Similarity Measure Support Vector Machines (CSMSVM). The method adds the cosine distance into Support Vector Machines (SVM) to eliminate redundant or irrelevant features during classifier construction.

**Embedded methods:** these methods perform feature selection as part of the modelling algorithm's implementation. The embedded methods integrate feature selection with modelling and thus are more computationally efficient than wrappers. The process can be done by optimising a two-part objective function with a goodness-of-fit term and a penalty for a more significant number of features.



**Figure 3.5: Functionality of the Hybrid Approach in Feature Selection. Encompasses two different evaluation methods like a filter-type evaluation and a wrapper-type evaluation**

When two different evaluation methods like a filter-type evaluation and a wrapper-type evaluation are used, the result is a hybrid evaluation method. In this method, the feature set is assessed using a data mining algorithm and an independent measure. The best subset for a given cardinality is achieved by using a separate standard, and the finest subset among the best subsets across different cardinalities is selected using a data mining algorithm. The hybrid approach can be seen in figure 3.5.

Large Margin hybrid algorithm for Feature Selection (LMFS) is a new feature selection method proposed by researchers Zhang, Xiong and Min (2019). The proposed method uses a distance-

based evaluation function, in which preferably samples from the same class are near each other. Besides, samples from other classes are far apart. Furthermore, a weighted bootstrapping search strategy is used to find a set of candidate feature subsets. Finally, the method uses cross-validation and a specific classifier to select the final feature subset from among the candidate feature subsets.

A hybrid method consisting of filter-wrapper methods for clustering has been proposed by researchers Solorio-Fernández, Carrasco-Ochoa and Martínez-Trinidad (2016). The proposed method combines a modified Calinski–Harabasz index and the spectral feature selection framework using the Laplacian Score ranking. The methodology sorts the features according to their relevance in the filter stage. Additionally, in the wrapper stage, the features are evaluated by considering them to be a subset.

Researchers Chen, Zhou and Yuan (2019) have proposed Hybrid Particle Swarm Optimisation with a Spiral-Shaped Mechanism (HPSO-SSM). The methodology uses a wrapper based approach to select the most optimal feature subset for classification. Three improvements have been suggested in HPSO-SSM. a) The diversity in the search process can be enhanced by using a logistic map sequence. b) The position quality of the next generation can be enhanced by introducing two new parameters into the original position update formula. c) A spiral-shaped mechanism is adopted as a local search operator around the known optimal solution region.

Another hybrid filter-based system has been proposed by researchers Manbari, AkhlaghianTab and Salavati (2019). The proposed feature selection algorithm is based on a combination of clustering and modified Binary Ant System (BAS), called FSCBAS. The researcher proposes that the new algorithm can overcome the search space and high-dimensional data processing challenges proficiently.

Forward sequential selection, Hill Climbing and Backward sequential selection are three different feature selection approaches used by diverse feature selection methods (Wan, 2019).

**- *Forward Sequential Selection (FSS)***

The primary aim of FSS is to produce an optimal feature subset by ignoring insignificant and immaterial features. The process of FSS is such that the process explores for the top feature in every iteration and adds to an empty set of optimal features. The search for the best feature stops and returns the current optimal set of essential features, if there is no improvement after adding any further feature or if all features are already added.

**- *Backward Sequential Selection (BSS)***

The primary objective of BSS is to study the contribution of all features at the initial stage and then eliminate the most redundant and immaterial features resulting in a smaller optimal feature subset. BSS then explores for the feature to be eliminated in every iteration from the full dataset. Validation procedures are used to evaluate the subsequent set, and if the evaluation rate of new feature subset is superior to the previous subset, the current best feature subset is replaced. The process continues until every feature is eliminated from the dataset and reaches an empty set. BSS is comparatively better than FSS in terms of computational operation.

**- *Hill Climbing (HC)***

A feature is either added or removed from the dataset at a time in HC. The method searches for the best features from a random set of features and then swaps the current status of each feature in the subset. The stopping criteria define the number of iteration for the selection of the optimal set. After reaching the limit of the last iteration, the method returns the last optimal set of features.

### 3.5 Proposed Feature Selection Methodology

Feature selection is a methodology that is used to reduce the number of features to build simpler machine learning algorithms that have shorter the prediction and training duration. This section explains Grey Wolf Optimisation, binary search and the proposed feature selection algorithm.

#### 3.5.1 Grey Wolf Optimization (GWO)

Researchers Mirjalili, Mirjalili and Lewis (2014) have proposed GWO to be a swarm intelligence based algorithm. Grey wolves inspire the GWO algorithm that searches for the most optimal method to hunt for preys. The proposed GWO algorithm works on the same procedure as found in the wild, and defines the pack hierarchy for organising the different roles in the wolves pack. The pack's members are divided into four groups named alpha, beta, delta and omega based on the role of the wolf while the hunt is on. Generally, the best solution for hunting is represented by alpha.

The GWO search process is based on the Swarm Intelligence (SI) algorithms and starts with creating a random population of grey wolves. Further, in this algorithm, four groups of wolves and their positions are designed, after which the distance to the prey is computed. A potential solution is represented by each wolf and further updated using the searching process. Powerful operations controlled in GWO applies two parameters to maintain the exploitation and exploration to avoid the local optima stagnation.

In GWO,  $\alpha$ ,  $\beta$  and  $\delta$  guides the hunting process and  $\omega$  wolves follows them. The calculation of the encircling behaviour of GWO is done, as shown below:

$$\vec{X}(t+1) = \vec{X}_p(t) + \vec{A} \cdot \vec{D} \quad (3.1)$$

$$\vec{D} = |\vec{C} \cdot \vec{X}_p(t) - \vec{X}(t)| \quad (3.2)$$

Where  $t$  indicates the current iteration,  $\vec{A}$  and  $\vec{C}$  are coefficient vectors,  $\vec{X}_p$  is the position vector of the prey, and  $\vec{X}$  indicates the position vector of a grey wolf.

$\vec{A}$  and  $\vec{C}$  vectors are calculated as follows:

$$\vec{A} = 2\vec{a} \cdot \vec{r}_1 - \vec{a} \quad (3.3)$$

$$\vec{C} = 2 \cdot \vec{r}_2 \quad (3.4)$$

Where,  $\vec{r}_1$ ,  $\vec{r}_2$  are vectors randomly in  $[0, 1]$ .  $\vec{a}$  a set vector linearly decreases from 2 to 0 over iterations.

Alpha is considered to the most optimal hunter to target the prey while considering the strategy used during the hunting process of grey wolves. Furthermore, beta and delta are aware of the prey's possible position during the hunt. This way, the three best solutions that have been found are kept valid until a particular iteration. The algorithm further forces others like omega, to modify their positions in the decision space consistent with the best place. The mechanism to update positions in GWO can be calculated as follows:

$$\vec{X}(t+1) = \frac{\vec{x}_1 + \vec{x}_2 + \vec{x}_3}{3} \quad (3.5)$$

Where  $x_1$ ,  $x_2$ ,  $x_3$  are defined and calculated as follows:

$$\begin{aligned} \vec{x}_1 &= \vec{x}_\alpha - A_1 \cdot (\vec{D}_\alpha), \\ \vec{x}_2 &= \vec{x}_\beta - A_2 \cdot (\vec{D}_\beta), \\ \vec{x}_3 &= \vec{x}_\delta - A_3 \cdot (\vec{D}_\delta) \end{aligned} \quad (3.6)$$

Where  $\vec{x}_1$ ,  $\vec{x}_2$ ,  $\vec{x}_3$  are the three best wolves (solutions) in the swarm at a given iteration  $t$ .

Where  $A_1$ ,  $A_2$ ,  $A_3$  are calculated as in Eq. (3.3).  $\vec{D}_\alpha$ ,  $\vec{D}_\beta$ ,  $\vec{D}_\delta$  are calculated as in Eq. (3.7)

$$\vec{D}_\alpha = |\vec{C}_1 \cdot \vec{X}_\alpha - \vec{X}|,$$

$$\vec{D}_\beta = |\vec{C}_2 \cdot \vec{X}_\beta - \vec{X}|,$$

$$\vec{D}_\delta = |\vec{C}_3 \cdot \vec{X}_\delta - \vec{X}|, \quad (3.7)$$

Where  $\vec{C}_1$ ,  $\vec{C}_2$ ,  $\vec{C}_3$  are calculated based on Eq. (3.4)

Vector  $\vec{a}$  is the main component that is used in GWO to tune exploration and exploitation. A suggestion has been made to decrease the vector for each dimension that is linearly proportional to the number of iterations, from 2 to 0. The formula used for the same is shown below:

$$\vec{a} = 2 - t \cdot \frac{2}{\max ter} \quad (3.8)$$

where  $t$  is the iteration number,  $ter$  is the optimisation total iterations number.

The pseudocode of the GWO algorithm is presented below (Gherboudj, Layeb and Chikhi, 2012):

1. Initialize the grey wolf population  $X_i = (i=1, 2, \dots, n)$
2. Initialise  $a$ ,  $A$  and  $C$
3. Calculate the fitness of each search agent
4.  $X_\alpha$  = the best search agent
5.  $X_\beta$  = the second-best search agent
6.  $X_\delta$  = the third best search agent
7. while ( $t < \text{Max number of iterations}$ )
  8.     for each search agent
  9.         Update the position of the current search agent by Eq. (3.5)
  10.     end for
  11.     Update  $a$ ,  $A$  and  $C$
  12.     Calculate the fitness of all search agents
  13.     Update  $X_\alpha$ ,  $X_\beta$ ,  $X_\delta$
  14.      $t=t+1$
  15. end while

16. Get Optimal Solution $X_{\alpha}$
---------------------------------------

The GWO is applied to select the optimal features. Some related works of GWO based feature selection methods are explained here. A wrapper approach was developed by researchers Emary et al. (2015) whereas a binary version of GWO is combined with the k-Nearest Neighbour (k-NN) as a fitness function to evaluate the candidate subsets of features. A faster convergence speed and higher classification accuracy were observed by the researcher's GWO based feature selection method.

Researchers Yamany, Emary and Hassanien (2016) have proposed a feature reduction approach based on GWO for maximising a rough set-based classification fitness function to search the feature space for a subset of features. A binary version of GWO algorithm is put forward by researchers Emary, Zawbaa and Hassanien (2016) to be used for feature selection, which is one of the crucial and essential modifications of the GWO algorithm.

Another set of researchers Tawhid and Ali (2017) have combined GA with GWO for minimising potential energy operations. Researchers Gaidhane and Nigam (2018) have proposed a combination of the Artificial Bee Colony (ABC) and GWO algorithms as a hybrid method to improve the performance of complex systems. GWOSCA methodology has been put forward by researchers Singh and Singh (2017) that uses using GWO and Sine Cosine Algorithm (SCA) as a hybrid methodology. All researchers and their work highlighted in the above section shows that hybrid methods have performed much better than other global or local search methods.

### **3.5.2 Cuckoo Search**

Yang and Deb (2009) have proposed and developed Cuckoo Search (CS) as an optimisation technique based on the nature of the cuckoo bird. The cuckoo bird has a natural habit of laying its



eggs in the nests of other species of birds and does not provide any care for its offspring. This behaviour of cuckoos classifies them as brood parasites.

Brood parasitism is classified into two types, obligate and non-obligate. Species of birds which have lost their ability to build a nest and incubate eggs are called as obligate brood parasites. These species of birds usually lay their eggs in the nests of birds of other species. Whereas, species of birds that lay their eggs in the nests of the same species are called non-obligate brood parasites.

The parasitic behaviour of cuckoos further increases the chance of survival of the cuckoo's gene. The cuckoo can have more offspring as the bird can devote its time to breed and lay eggs rather than build nests and take care of its young. This behaviour of the cuckoo is based on the selfish gene theory. The cuckoos depend on the other birds to incubate its eggs and feed the young chicks. The parasite cuckoo is thus freed from the nest-building duties, incubation of eggs and caring for the young (Payne and Sorensen, 2005).

Cuckoos have been successful as brood parasites as they carefully select the nests of other birds to lay their eggs. The cuckoo looks for similar colour and pattern of eggs that match its egg to avoid detection by the host bird. Sometimes the host bird may recognise a cuckoo egg that is present in their nest and may take appropriate action. The action may be to throw the alien egg out of the nest or abandon the nest completely to build a new nest. To evade its egg from being destroyed, the cuckoo continuously tries to find nests with similar eggs as its own, whereas the host would always try to find alien eggs and destroy them. This struggle between the cuckoo and host is similar to the continuing competitive attempt, where each bird is trying to out-survive the other. The least egg rejection shown by the hosts is directly proportional to the cuckoos that can find nests which match its eggs. Each egg already present in a nest represents a solution, whereas every cuckoo egg added to the nest represents a new solution. A not so good egg in the nest is replaced with a better new

solution by the cuckoo. Each nest has one egg in the purest form, and a new solution is generated by Lévy flight.

The rules for CS are expressed as follows (Senthilkumar et al.,2016):

- “Each cuckoo lays one egg at a time and dumps it in a randomly chosen nest.”
- “The best nests with high-quality eggs will carry over to the next generation.”
- “The number of available host nests is fixed, and a host can discover a foreign egg with a probability  $p_a \in [0, 1]$ . In this case, the host bird can either throw the egg away or abandon the nest to build an entirely new nest in a new location.”

When looking at the solution algorithmically, each host nest  $n$  is defined as an agent who can contain a simple egg  $x$  or when the issue relates to multiple dimensions, more than one. The CS algorithm starts by placing the nest population randomly in the search space. The nests are updated using random walk via Lévy flights after each algorithm iteration (Rodrigues et al., 2013):

$$x_i^j(t) = x_i^j(t-1) + \alpha \oplus Levy(\lambda) \quad (3.9)$$

$$Levy \cong s^{-\lambda}, (1 < \lambda \leq 3) \quad (3.10)$$

Where  $s$  is step size, and  $\alpha > 0$  is the step size scaling factor/parameter. Here the entry wise product  $\oplus$  is similar to those used in PSO, and  $x_i^j$  stands for the  $j^{th}$  egg at nest  $i$  (solution)  $i=1, 2, \dots, m$  and  $j=1, 2, \dots, d$ . The Lévy flights employ a random step length which is drawn from a Lévy distribution. Consequently, the CS algorithm's step length is much longer in the long run and is more efficient in exploring the search space. Finally, the nests which have eggs with the lowest quality, are replaced to new ones according to a probability  $p_a \in [0, 1]$  (Rodrigues et al., 2013).

The pseudocode of the CS algorithm is presented (Al-Obaidi, 2013):

1. Generate an initial population of  $n$  host nests;
2. While ( $t < \text{MaxGeneration}$ ) or (stop criterion)
  3. Get a cuckoo randomly (say,  $i$ ) and replace its solution by performing Lévy flights;
  4. Evaluate its fitness  $F_i$ ;
  5. Choose a nest among  $n$  (say,  $j$ ) randomly;
  6. If ( $F_i < F_j$ )
    7. Replace  $j$  by the new solution;
  8. End If
9. A fraction ( $p_a$ ) of the worse nests are abandoned, and new ones are built;
10. Keep the best solutions/nests;
11. Rank the solutions/nests and find the current best;
12. Pass the current best to the next generation;
13. End While.

### 3.5.3 Proposed Feature Selection

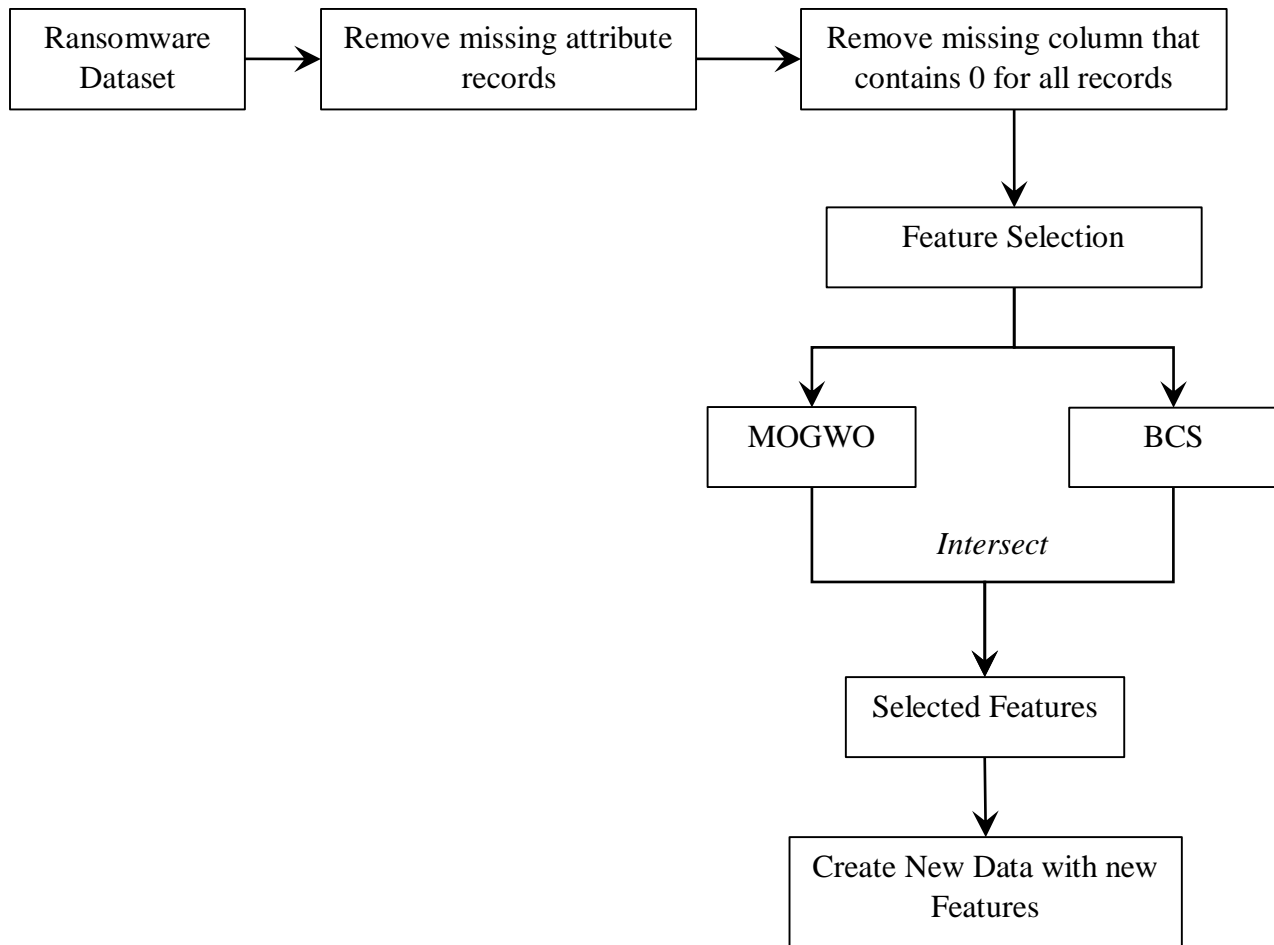
Selecting the most representative features and eliminating redundant noisy and irrelevant features, along with reducing the number of features, are reasons to implement feature selection. The issue of searching for the best set of features is considered a complex and challenging problem due to the vast search space when the number of features is extensive. This section explains the proposed feature selection algorithm that uses GWO and BCS. Figure 3.6 shows the architecture for preprocessing and feature selection.

The pseudocode of the proposed feature selection follows:

1. Read the Dataset  $D$
2.  $D_1 = \text{Remove missing value records in } D$
3.  $D_2 = \text{Remove features (columns) that contains zero value for all records in } D_1$
4.  $F_1 = \text{Apply MOGWO}$
5.  $F_2 = \text{Apply BCS}$

6.  $SF = F_1 \cap F_2$

7. Create new Data set  $D'$  using SF (Selected Features)



**Figure 3.6: Subsection of the proposed process and indicates the Preprocessing and Feature Selection Architecture**

#### **- Multi-Objective Grey Wolf Optimization (MOGWO)**

This section explains the MOGWO algorithm for feature selection. MOGWO consists of two main components, a grid and an archive. The responsibility of the grid component is to keep the archive solutions as varied as possible. In MOGWO, the objective space is divided into several regions named grids. All grid locations need to be recalculated if a newly obtained solution lies outside the

grid to cover the new solution. Moreover, a new solution is directed to the portion of the grid with the lowest number of particles, if the solution lies within the grid (Mirjalili et al., 2016).

The other component, archive, is deciding as to whether a solution should be added to the archive or ignored. A new solution needs to be discarded promptly if the solution is dominated by one of the archive members. Alternatively, if the archive members do not dominate the new solution, the solution should be added to the archive. If a new solution dominates a member of the archive, it has to be replaced by the new solution. To conclude, the adaptive grid mechanism is triggered if the archive is full (Mirjalili et al., 2016).

The pseudocode of the MOGWO is presented as follows (Mirjalili et al., 2016):

1. Initialize the grey wolf population  $X_i(i = 1, 2, \dots, n)$
2. Initialise  $a$ ,  $A$ , and  $C$
3. Calculate the objective values for each search agent
4. Find the non-dominated solutions and initialised the archive with them
5.  $X_\alpha$ =Select Feature (archive)
6. Exclude alpha from the archive temporarily to avoid selecting the same feature
7.  $X_\beta$ =Select Feature (archive)
8. Exclude beta from the archive temporarily to avoid selecting the same feature
9.  $X_\delta$ =Select Feature (archive)
10. Add back alpha and beta to the archive
11.  $t=1$
12. While ( $t < \text{Max number of iterations}$ )
13.     For each search agent
14.         Update the position of the current search agent by equations (3.5), (3.6), (3.7)
15.     End For
16.     Update  $a$ ,  $A$ , and  $C$
17.     Calculate the objective values of all search agents
18.     Find the non-dominated solutions

19. Update the archive concerning the obtained non-dominated solutions
20. If the archive is full
  21. Run the grid mechanism to omit one of the current archive members
  22. Add the new solution to the archive
23. End if
24. If any of the new added solutions to the archive is located outside the hypercubes
  25. Update the grids to cover the new solution(s)
26. End if
27.  $X_{\alpha}$ =Select Feature (archive)
28. Exclude alpha from the archive temporarily to avoid selecting the same feature
29.  $X_{\beta}$ =Select Feature (archive)
30. Exclude beta from the archive temporarily to avoid selecting the same feature
31.  $X_{\delta}$ =Select Feature (archive)
32. Add back alpha and beta to the archive
33.  $t=t+1$
34. End while

### - *Binary Cuckoo Search (BCS)*

A feature selection model based on a binary version of the Cuckoo Search (BCS) which consists of a search space modelled as a d-cube. In this model, d refers to the number of features. A set of binary coordinates is connected with each nest that denotes whether a feature will belong to the final set of features. Additionally, the supervised classifier's accuracy determines the function to be maximised (Rodrigues et al., 2013).

Solutions are updated in the search space towards continuous-valued positions in the traditional CS algorithm. Disparately, in the BCS for feature selection, "the search space is modelled as an n-dimensional Boolean lattice, in which the solutions are updated across the corners of a hypercube." Additionally, a solution binary vector is employed to solve the issue of selecting or not selecting a

given feature. In BCS, the number “0” indicates if a feature will not be chosen to compose the new dataset and number “1” indicates a selection (Rodrigues et al., 2013). The binary vector is built by using Equation 3.12, which can provide only binary values in the Boolean lattice restricting the new solutions to only binary values:

$$S\left(x_i^j(t)\right)=\frac{1}{1+e^{-x_i^j(t)}} \quad (3.11)$$

$$x_i^j(t+1)=\begin{cases} 1 & \text{if } S\left(x_i^j(t)\right) > \sigma \\ 0 & \text{otherwise} \end{cases} \quad (3.12)$$

In which  $\sigma \sim U(0,1)$  and  $x_i^j(t)$  denotes the value of the new eggs at time step  $t$ .

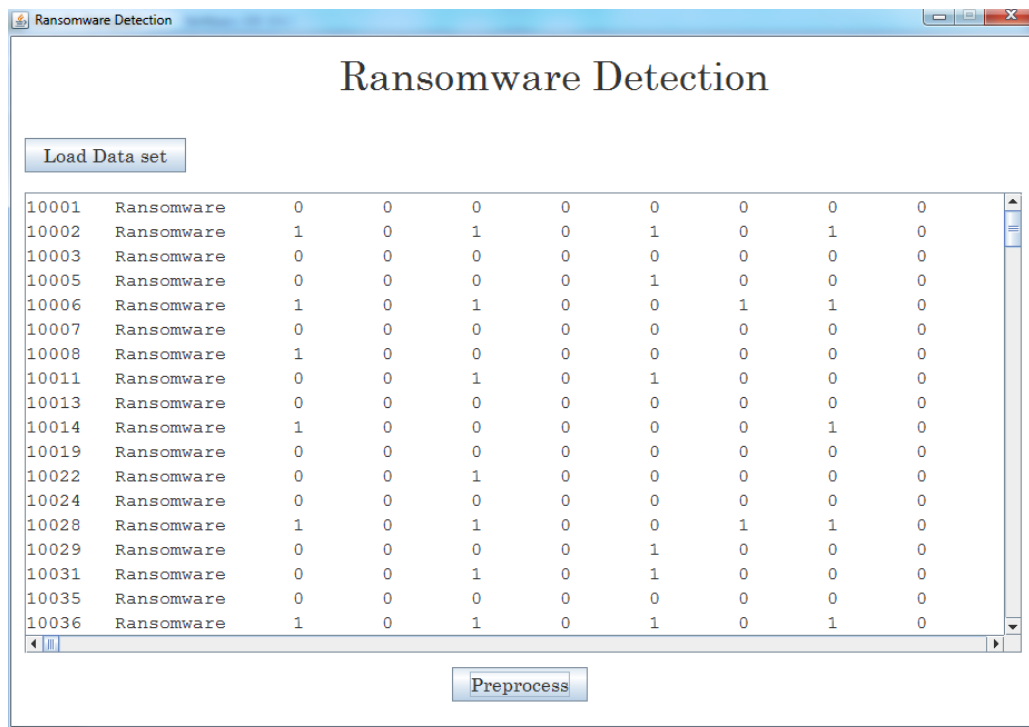
The pseudocode of the BCS for feature selection is presented as follows:

1. For each nest
2.      $x_i^j(0) = \text{Random}\{0,1\}$
3.      $f_i = -\infty$
4. End For
5. Global\_Fitness =  $-\infty$
6. For each iteration  $t$
7.     For each nest
8.         Create new training ( $TS_1$ ) and evaluating set ( $ES_1$ ) from  $TS$  and  $ES$  (original)
9.         Compute classification accuracy  $acc$ .
10.        If ( $acc > f_i$ )
11.            $f_i = acc$
12.        End If
13.     End For
14.      $maxFit = \max(f)$
15.     If ( $maxFit > globalFit$ ) then
16.         Global\_Fitness =  $maxFit$
17.     End If

18. Select the worst nests and replace them for new solutions
19. Update the nest using Levy flights
20. End For

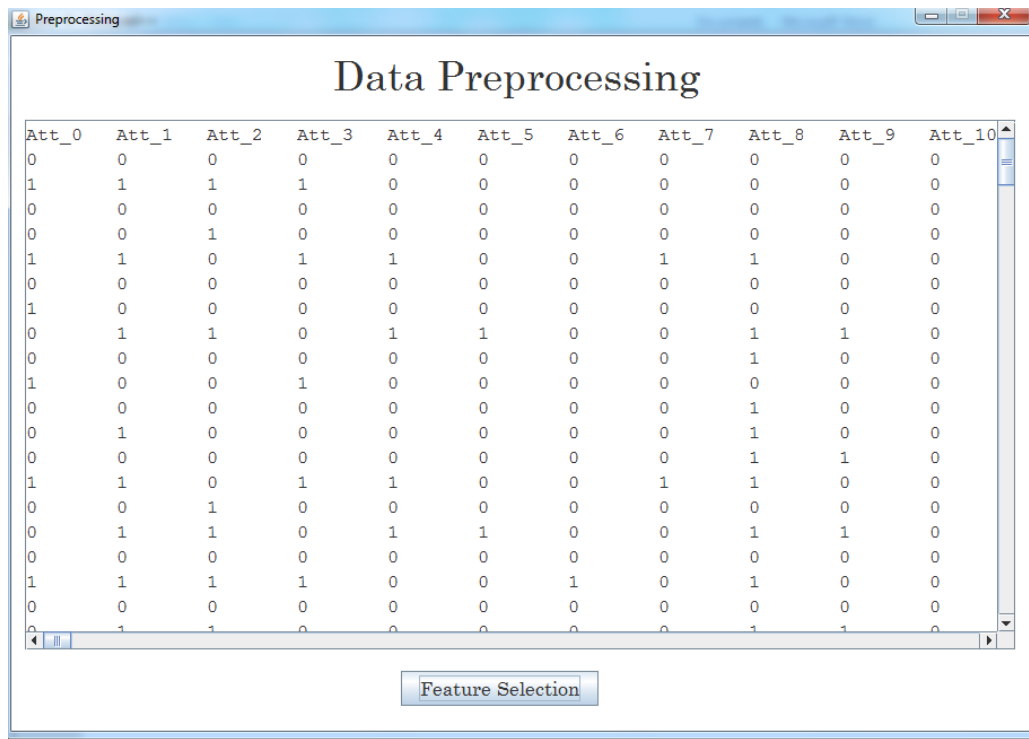
### 3.6 Product: Pre-processing and Feature Selection

The following section shows some sample screenshots of preprocessing and feature selection. Figure 3.7 shows the screen after the input dataset has been loaded into the program. This dataset will be preprocessed remove invalid records and attributes. This action is shown in figure 3.8. The screen shows that the number of attributes as being lesser after the initial preprocessing. The features are generated by using GWO and Binary Search algorithms, and these algorithms avoid selecting the same features many times over. So the order of features is not needed as we require only the best feature to be selected in the search space. The proposed algorithm selects the most representative features and eliminates redundant noisy and irrelevant features.



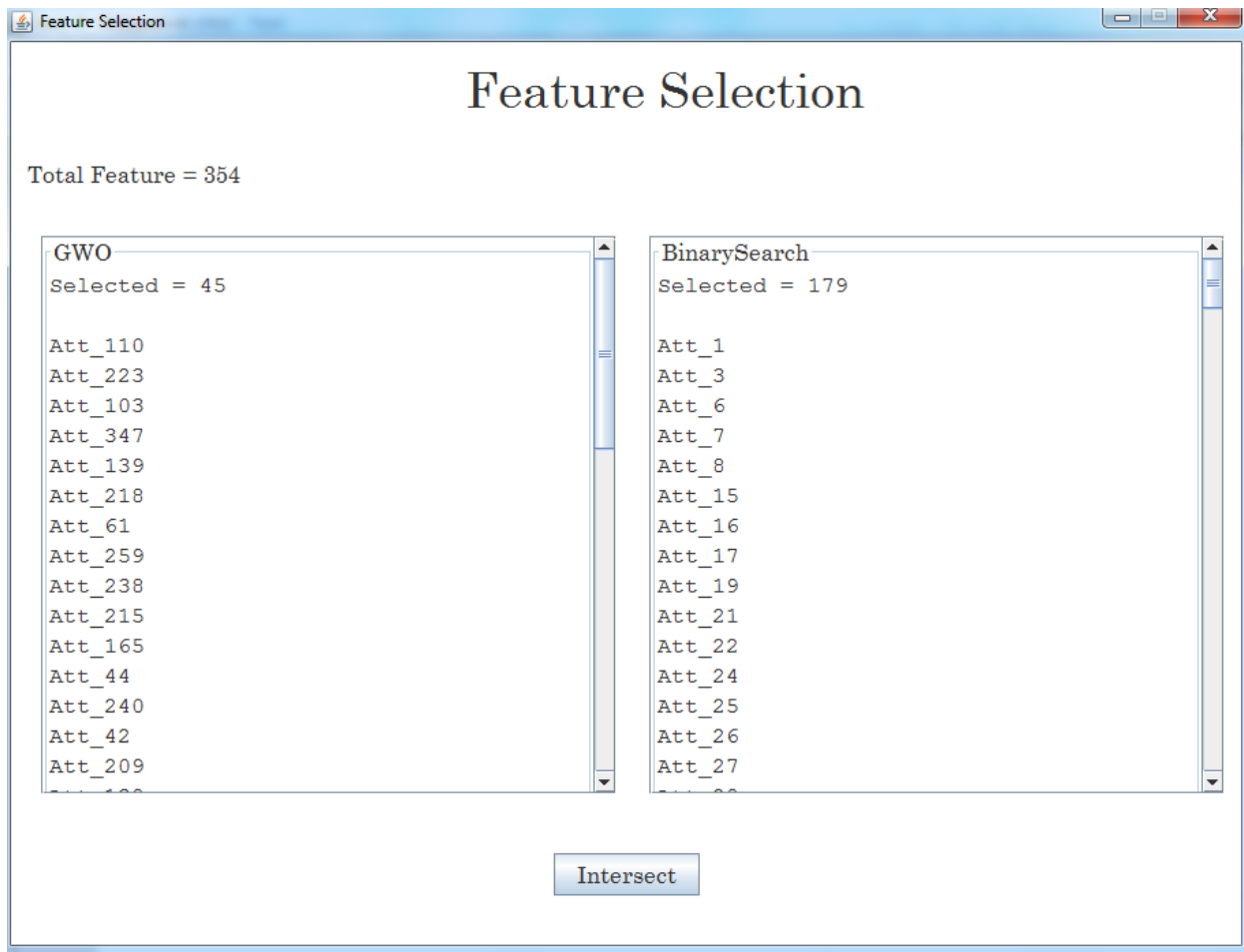
**Figure 3.7: Screenshot showing the ransomware dataset loaded to the proposed system.**



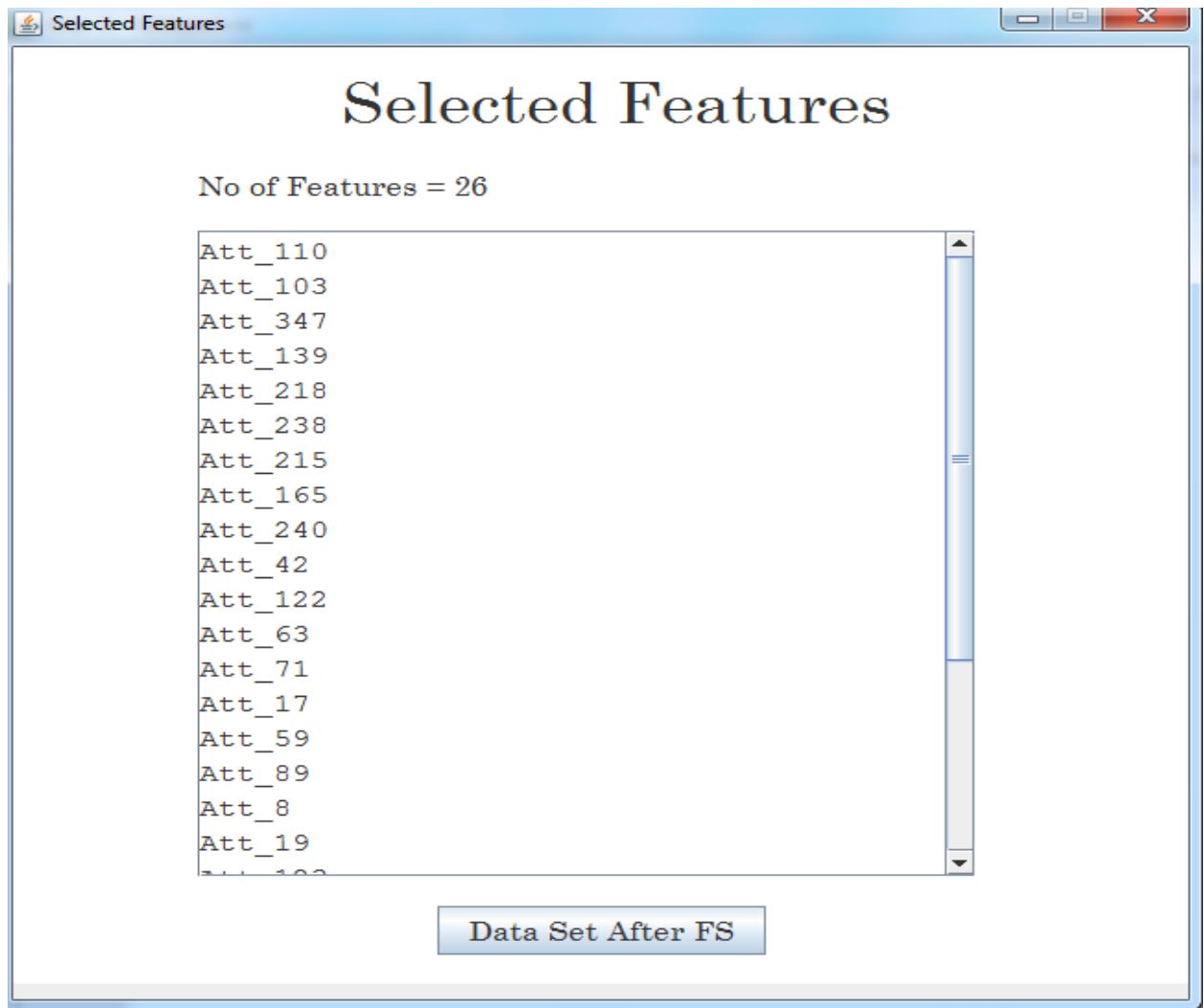


**Figure 3.8: Screenshot indicating the number of attributes as being reduced after the initial preprocessing**

Consequently, figure 3.9 indicates feature selection using the GWO and Binary Search algorithms, which results in figure 3.10 that indicates the selected features after the two algorithms have been applied.

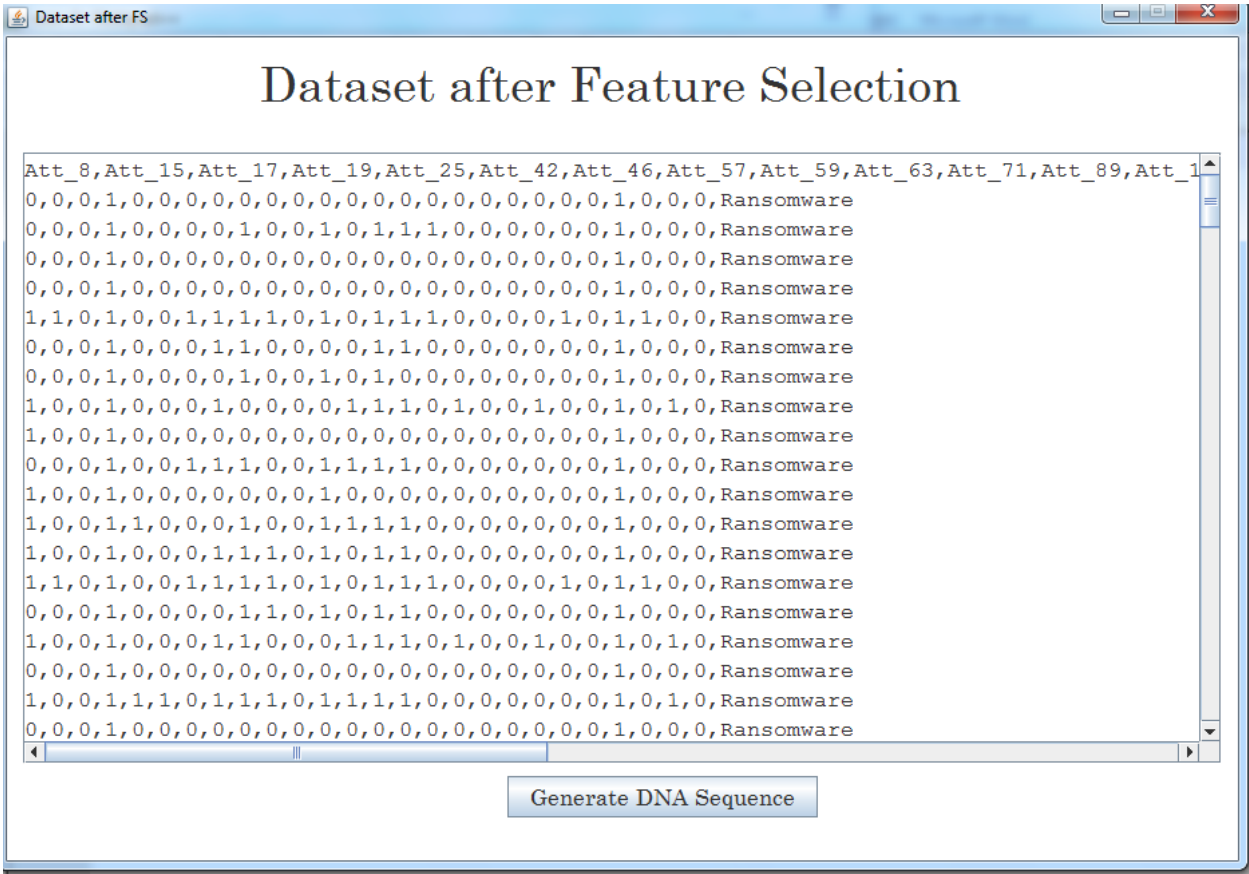


**Figure 3.9:** Screenshot indicating the number of features selected after applying the GWO and Binary Search algorithms



**Figure 3.10:** Screenshot indicates the selected features after the two algorithms have been applied.

Finally, figure 3.11 shows the new dataset that will be used based on the selected features.



**Figure 3.11: Screenshot indicating the resultant Dataset after Feature Selection**

### 3.7 Summary

This chapter explains the preprocessing and feature selection algorithm for ransomware detection. Ransomware data is very high dimensional data. The single feature selection method is not enough for efficient features, hence this chapter highlights two feature selection algorithm namely, Grey Wolf Optimisation and cuckoo search algorithm as an efficient feature.

## **Chapter 4:**

### **Digital DNA Sequence Generation**

#### **4.1 Introduction**

The term Human Genome sequencing refers to methods that are used to determine the order of the nucleotides bases adenine (A), guanine (G), cytosine (C) and thymine (T) in a molecule of human DNA. The structure of every gene product is defined and dictated by DNA, and it is the information store that delineates every part of the organisms. Genetic inheritance is the complete set of instructions that makes up the human genome and is derived from the nucleotide order. Advanced research that involves studying biological processes includes the knowledge of human DNA sequences of genes and other parts of the genome of organisms. Genome study is also applied in the field that is relevant and related to forensic research. This research has extended the DNA sequencing methodologies from biology to the field of computer science for detecting ransomware.

This chapter uses the design constraints of the human genome sequence and k-mer frequency vector. K-mer frequency is a unique subsequence of a particular sequence of length k. The measure is also used within the context of computational genomics, in which k-mers are comprised of nucleotides bases of DNA, k-mers are taking advantage of genomics to assemble sequence and enhance heterologous gene expression. In this research, k-mer frequency is used to generate a new dataset for ransomware detection.

The proposed Digital DNA sequence generation for ransomware detection is explained in this chapter. A newly generated dataset after feature selection is used to generate the DNA sequence. The design constraint of Digital DNA is computed, and k-mer frequency vector is generated for the DNA sequence. Based on these computations and vector, a new dataset is generated for ransomware detection training phase.

This chapter presents in detail Digital DNA sequences for ransomware detection. Section 4.2 explains the basis of Human DNA Sequence, design constraints and its applications. Section 4.3 explains different methods used for generating the human genome sequence. The proposed Digital DNA Sequence generation for ransomware detection is described in section 4.4, and it includes design constraints and k-mer frequency vector. Section 4.5 provides screenshots and sample code of the product developed and finally, section 4.6 gives a summary of this chapter.

## **4.2 Basis of Human Genome Sequence**

Human DNA consists of nucleotides, which are the chemical building blocks and referred to as the blueprint of life. The building blocks primarily consists of phosphate and sugar group. The third element is one of the four types of nitrogen bases which are either Adenine (A), Thymine (T), Guanine (G), or Cytosine (C) (Watson and Crick, 1953). The base sequence of a DNA fragment can be identified using human genome sequencing. Genetic properties of humans, for example, blue eyes are due to a DNA sequence of ATCGTT and brown eyes are a result of the sequence of ATCGCT. Gene is defined as instructions to make a protein that are within a DNA (Bisht and Panda, 2014). Forensic science related to crime scenes and advanced research is influenced dramatically from human genome sequencing methodologies. The primary objective of human genome sequence generation methodologies is to analyse DNA and derive results with very high reliability and accuracy (Kumar, 2012).

### **4.2.1 Human DNA Sequence Design**

In DNA computing, good human DNA sequences should have excellent chemical properties and can avoid non-cross hybridised properties with others. The goal of the DNA sequence design optimisation is to select a set of DNA sequences with equal-length  $n$ , and each sequence can satisfy specific combinatorial and thermodynamic constraints.

There are various kinds of criteria to constraint the set of sequences, which are Free energy, Continuity, Similarity, H-measure, GC ratio and Melting temperature, among others. These constraints are classified into four categories:

***- Preventing undesired reactions:***

Duplexes between a given DNA sequence and its complement are formed by using this criterion to force an action using the set of sequences. H-measure, Similarity and Reverse complement hamming distance are included in this category. H-measure tests the possibility of unintended DNA base pairing that is based on the Hamming distance. An inverse Hamming distance between two given DNA sequences is defined as Similarity. The Hamming distance using one DNA sequence and the reverse complement of another sequence is checked under Reverse complement Hamming distance.

***- Controlling secondary structures:***

The interaction of single-stranded DNA usually forms secondary structures. A bulge loop, an internal loop and a hairpin loop are included in the secondary structure. The secondary structure can be predicted based on thermodynamic parameters, and many algorithms have been proposed to achieve this task (Kawashimo et al., 2007), or, the Hamming distance of given sequences can be calculated by folding the sequences to hybridise with itself. The repeated run of identical bases are verified using the continuity test, and consequently, an unusual secondary structure can be formed if one base is repeated. Even though the secondary structure of DNA strands is usually forbidden, they can be used to implement DNA computing in several ways (Lee et al., 2002).

***- Controlling the chemical characteristics of Human DNA sequences:***

The field of human genome desires to control DNA sequences to have comparable chemical characteristics. Free energy, melting temperature, and GC ratio are the usual measures used towards

this criterion. Free energy is the required energy to create a duplex, or it can also be the energy required to break a duplex. The temperature at which 50% of the oligonucleotide and its perfect complement are in the duplex is referred to as Melting temperature. Moreover, finally, the percentage of guanine or cytosine in a whole DNA sequence is the GC ratio. Concerning fixed protocols, an accurate and reliable measure for the relative stability of a DNA duplex is its free energy. The free energy of a DNA duplex is closely related to Melting temperature, and GC content is a much less accurate measure of stability.

**- *Restricting one of the Human DNA symbols in DNA sequences:***

The composition, either the DNA base or the subsequence of a DNA sequence, is regulated by this criterion. Restriction enzyme site also referred to as special DNA subsequence, should be controlled to achieve proper reactions.

#### **4.2.2 Criteria for Human DNA Sequence Design**

This section explains seven criteria for Human DNA sequence design constraints. The criteria are H-measure, Continuity, Similarity, Hairpin, GC content, Minimum Free Energy and melting temperature.

The uniqueness of each DNA sequence is estimated by choosing Similarity and H-measure. The secondary structure of DNA sequence is avoided using Continuity and Hairpin. Melting temperature and GC content used to maintain uniform chemical characteristics; minimum free energy to constrain the thermodynamic stability of DNA sequence.

**- *Continuity***

The number of same bases in a single-stranded DNA is calculated using Continuity. If the same bases are occurring continuously in a sequence, it will weaken the stability of the strand. The formulations are shown as follows:

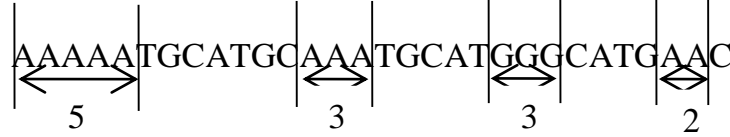


$$F_{Con}(\Sigma) = \sum_{i=1}^m \sum_{j=1}^{n-t+1} \sum_{\alpha \in \{A,T,C,G\}} T(C_{\alpha}(x_i, j), t)^2$$

$$C_{\alpha}(x_i, j) = \begin{cases} c, & \text{if there is } c \text{ such that } x_i \neq \alpha, x_i^{j+k} = \alpha, \\ & \text{for } 1 \leq k \leq c, x_i^{j+k+1} \neq \alpha \\ 0, & \text{otherwise} \end{cases}$$

Where  $x_i$  ( $1 \leq i \leq m$ ) denotes the DNA sequences with length  $n$ ,  $m$  is the cardinality of a set of DNA sequences;  $x_i^j$  is the  $j$ th base in DNA sequence  $x_i$ , and  $t$  is the target. If  $i > j$ ,  $T(i, j)$  is 1, otherwise is 0.

The sequence can display unexpected structures if the same bases happen continuously in a sequence.  $F_{Con}(x)$  calculates the degree of successive occurrence of the same base, as shown in Figure 4.1.



**Figure 4.1: Continuity Measure that disallows the repeated runs of the same base over the given threshold**

Continuity measure disallows the repeated runs of the same base over the given threshold as can be seen from figure 4.1. The first run violates the continuity, and other runs do not if the threshold is four.

#### - Hairpin

The probability of a single-stranded DNA that is used to form a secondary structure is calculated as Hairpin. The hairpin formulation is defined as follows:

$$F_{Hairpin}(\Sigma) = \sum_{i=1}^m \sum_r^{(n-2*pinlen)} \sum_{c=pinlen+[r/2]}^{(n-pinlen-[r/2])} Hairpin(x_i, c)$$

Where  $pinlen$  denotes the minimum length of the stem, and  $r$  is the minimum length to form a hairpin ring. A hairpin structure is formed at position  $c$  for the sequence  $x_i$ .  $Hairpin(x_i, c)$  is 1 when a reverse-complement distance of two sequences which sequence  $ix$  is folded around the  $c$ -th base is more than  $pinlen/2$ , otherwise is 0.

### - Similarity

Similarity is used to measures the similarity of two given sequences in the same direction to keep each sequence as unique as possible. The similarity measure can be calculated as follows:

$$F_{Similarity}(\Sigma) = \sum_{i=1}^{i=m} \sum_{\substack{1 \leq j \leq m \\ j \neq i}} \max_{0 \leq g \leq n} \max_{0 \leq k \leq n+g-1} S(x_i(-)^g x_i, \sigma^k(x_j))$$

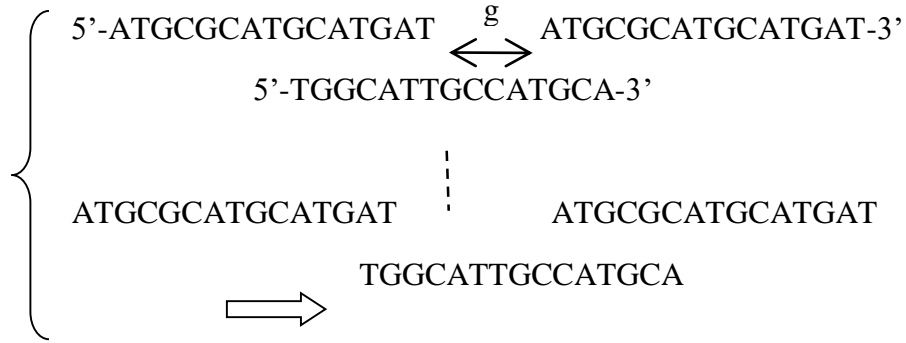
Where  $(-)^g$  denotes  $g$  gabs,  $\sigma^k(x_j)$  denotes the  $k$  position right shift for DNA sequence  $x_j$ ,  $S(*, *)$  is the number of corresponding places where two characters are the same.

Figure 4.2 shows the example for similarity measure, the similarity between 5'-ATGCGCATGCATGAT-3' and 5'-TGGCATTGCCATGCA-3' is calculated. Sequence 5'-ATGCGCATGCATGAT-3' is extended by adding its sequence to the 3', ending with gaps.

Given two strands, the similarity measure checks how many positions are the same. An example of the similarity measure can be seen in figure 4.2. In this example, two DNA strands are selected, and one DNA strand is elongated with gap  $g$ . Furthermore, comparison in each position is conducted using position shifts between the regular strand and the elongated strand. Once all the shifts have been checked, gap  $g$  is increased by one, and the comparison process continues.

For every pair of sequences in a set

For g = 0 to 1



**Figure 4.2: An example of a Similarity Measure to check how many positions are the same. In this example, two DNA strands are selected, and one DNA strand is elongated with gap g.**

#### - H-measure

H-measure values are used to prevent cross-hybridisation of two sequences, including position shift. The calculation is based on the number of complementary nucleotides and the formula for the same is defined as follows.

$$F_{H-measure}(\Sigma) = \sum_{i=1}^{i=m} \sum_{\substack{1 \leq j \leq m \\ j \neq i}} \max_{0 \leq g \leq n} \max_{0 \leq k \leq n+g-1} C(x_i(-)^g x_i, \sigma^k(x_j^R))$$

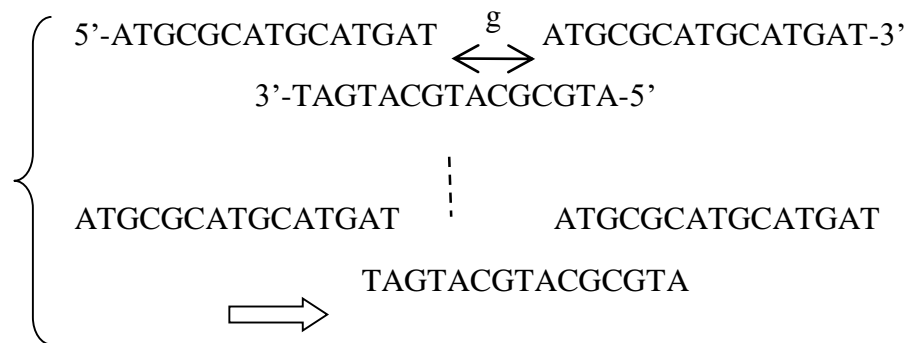
Where  $x_j^R$  is the reverse sequences of sequence  $x_i$ ,  $\sigma^k(x_j^R)$  is the number of corresponding places where two nucleotides are the same.

An example of H-measure is shown in figure 4.3. Cross-hybridisation of two sequences can be prevented by computing H-measure that accounts for how many nucleotides are complementary

between the given sequences. H-measure also uses the elongated sequence for the similarity measure. The significant difference between similarity measure and H-measure is that similarity measure checks and compares two strands with the same direction whereas H-measure checks the given two strands with opposite direction, 5'-3' and 3'- 5'. H-measure also computes the unintended DNA base pairing rate.

For every pair of sequences in a set

For  $g = 0$  to 1



**Figure 4.3: Example of H-measure. Cross-hybridisation of two sequences can be prevented by computing H-measure that accounts for how many nucleotides are complementary between the given sequences.**

### - *Melting Temperatures*

The temperature at which half of a double-stranded DNA starts to break into its single-stranded form is referred to as Melting temperature. Equations such as GC% method (Wetmur, 1991), the nearest neighbour mode (SantaLucia, 1998), and others can be used to calculate the melting temperature. This section explains the nearest neighbour mode for calculating the melting temperature, and the formula is defined as

$$F_{Tm}(\Sigma) = \sum_{i=1}^m \left( \frac{\Delta H^o(x_i)}{\Delta S^o(x_i) + R \ln(C_T/4)} - 273.15 \right)$$

Where  $\Delta H^o(x_i)$  is the enthalpy of the generated sequence  $x_i$ ,  $\Delta S^o(x_i)$  is the entropy of the generated sequence  $x_i$ ,  $R$  is the gas constant;  $C_T$  is salt concentration.

#### **- GC Content**

The percentage of G base and C base in a DNA sequence is referred to as the GC content. It is an essential criterion for keeping the uniform chemical properties of DNA sequences. The formulation of GC is defined as follows;

$$GC(x_i) = \frac{\#G + \#C}{|x_i|}$$

where  $\#C$ ,  $\#G$  are the amount of C and G in sequences, respectively, and  $|x_i|$  is the number of bases for DNA sequence  $x_i$

#### **- Minimum Free Energy**

Known that the secondary structure is more stable for DNA sequence with small free energy. The minimum value among free energies of all possible secondary structures of a sequence is referred to as the minimum free energy (Garzon, 2008). The nearest neighbour model is used to compute the minimum free energy. The formulation is defined as follows:

$$\Delta G^o(x_i) = \sum_j n_j \Delta G^o(j) + \Delta G^o \left( \text{init} \frac{w}{\text{term}} G.C \right) + \Delta G^o \left( \text{init} \frac{w}{\text{term}} A.T \right) + \Delta G^o(\text{sym})$$

Where  $\Delta G^o$  is the standard free energy change for the ten possible Watson-Crick nearest-neighbours.  $n_j$  is the number occurrences of each nearest neighbour  $j$ , and  $\Delta G^o(\text{sym})$  is 0.43 kcal/mol if the duplex is self-complementary; otherwise, it is zero.

### 4.2.3 DNA Sequence Application

DNA Sequence application in several industries are listed below (Bisht and Panda, 2014):

- The field of agriculture benefits immensely from human genome sequencing. Agriculturists have conducted sequencing and mapping of the whole genome of microorganisms to increase and have different yields of food plants. Crop yields have been improved by increasing the resistance of food plant against insect and pests. The resistance has been increased by the manipulation of specific genes of bacteria.
- Human genome sequencing can be used in the medical field to detect genes which are related to acquired or hereditary diseases.
- Human DNA and its characteristics can be used on data extracted from the crime scene in the form of blood, nail, hair and skin samples to identify the criminals using medical forensics. Additionally, a paternity test can be conducted using DNA sequencing.
- Methodologies of gene manipulation can be achieved using sequenced DNA information.
- A polypeptide sequence from the data bank can be detected using DNA sequences or DNA sequences from other organisms can be compared for phylogenetic analysis.
- A molecular evolution map can be constructed in addition to discovering variations among interspecies and intraspecies.
- In this thesis, we are proposing Digital DNA sequencing to detect malware.

### 4.3 Methods of DNA Sequence

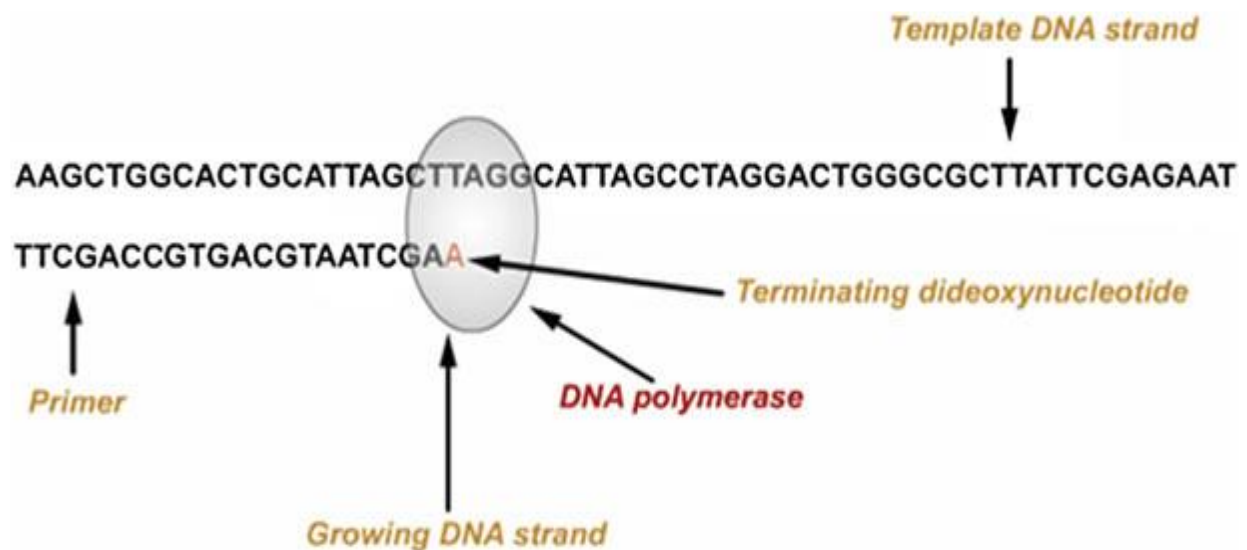
The determination of the precise sequence of nucleotides in a sample of DNA is referred to as DNA sequencing. This section explains seven essential methods used for DNA sequencing (Kumar, 2019).

### - Sanger's Method

Researchers Sanger and Coulson (1975) were among the first to propose a DNA sequencing method. The proposed method was also referred to as plus and minus sequencing that utilised *E. coli* DNA pol I and DNA polymerase from bacteriophage T4 with different limiting triphosphates. The proposed method had low efficiency.

Consequently, the researchers Sanger, Nicklen and Coulson (1977) improved and invented a novel method for DNA sequencing via enzymatic polymerisation that revolutionised DNA sequencing methodologies. The adopted method involved a third form of ribose sugar.

The method indicates that “ribose has a hydroxyl group on both the 2’ and the 3’ carbons, whereas deoxyribose has only one hydroxyl group on the 3’ carbon. Deoxyribose is a third form of ribose, in which the hydroxyl group is missing from both the 2’ and the 3’ carbons.” Whenever a dideoxynucleotide is combined into a polynucleotide, the chain permanently terminates.



**Figure 4.4: Principle of Sanger sequencing.** Sanger developed the chain termination method to generate all possible single-stranded DNA molecules complementary to a template that starts at a common 5’ base and extends up to 1 kilobase in the 3’ direction.

Sanger developed the chain termination method to generate all possible single-stranded DNA molecules complementary to a template that starts at a common 5' base and extends up to 1 kilobase in the 3' direction (Bisht and Panda,2014), as can be seen in Figure. 4.4.

The identity of the 3'-end base is allowed in each molecule by the labelling of single strands of DNA.

#### **- *Maxam and Gilbert Method***

Researchers Maxam and Gilbert (1977) have described a sequencing method based on chemical degradation at certain locations of the DNA molecule. Specific chemical agents have been used to end-labelled DNA fragments which are subjected to random cleavage at adenine, cytosine, guanine or thymine positions. PolyAcrylamide Gel Electrophoresis (PAGE) is used to separate the products of these four reactions. The sequence can be easily read from four parallel lanes in the sequencing gel, similar to Sanger's method

Additional cautions in the Maxam and Gilbert method comprises of purification and separation of DNA fragments and better analysis time. Consequently, this technology is not suitable for high throughput large-scale investigation.

#### **- *Hybridisation Method***

Specific DNA sequences using hybridisation of complementary probes are detected using researcher Ed Southern's (1990) sequencing by hybridisation technique. Computer analysis of the hybridisation pattern of the sample DNA to come up with the target sequence.

The target molecule will hybridise to all words which are related to Watson-Crick complements and occur somewhere along its sequence. The average length of a uniquely reconstructible sequence using an 8-mer array is < 200 bases, which is below the single read length on the commercial gel-lane machine. There is no way to determine the exact sequence when several



sequences have the same spectrum. The main weakness of sequencing by hybridisation is that it is an ambiguous solution.

#### **- *Pal Nyren's Method***

In 1996, Pal Nyren's group reported that efficient incorporation during a sequencing-by-synthesis protocol could be obtained by using natural nucleotide. The detection was based on the pyrophosphate (inorganic biphosphate) released during the DNA polymerase reaction, the quantitative conversion of pyrophosphate to ATP by sulfurylase and the subsequent production of visible light by firefly luciferase.

The first significant improvement was the inclusion of dATPaS in place of dATP in the polymerisation reaction, which enabled the pyrosequencing response to be performed inhomogeneous phase in real-time.

Visible light is generated that is proportional to the number of incorporated nucleotides in a cascade of the enzymatic reaction. Inorganic biphosphate (PPi) is released as a result of nucleotide incorporation by the polymerase is a result of a cascade of a nucleic acid polymerisation reaction (Stefanis et al., 2013).

The energy to luciferase to oxidise luciferin and generate light is provided by the release of PPi which subsequently converts to ATP by ATP sulfurylase. The light so generated is captured by a CCD camera and recorded in the form of peaks known as pyrogram (compared with electropherograms in Sanger's method). The sequence of the template can be determined as the added nucleotide is known.

#### **- *Capillary Gel Electrophoresis***

In these systems, slab gel electrophoresis is replaced by capillary gel electrophoresis to analyse DNA samples. In these systems, instead of scanning DNA as it migrates through 96 lanes each in a series of 96 capillary tubes, DNA fragments pass are scanned.

In the original models of the above old slab gel machines, gels must be poured and reagents frequently reloaded, interrupting the sequencing.

In capillary gel sequencing systems, the robot moves the DNA samples and reagents through the tubes continuously. The system produces a steady flow of data, each signal representing one of the four DNA bases (adenine, cytosine, guanine and thymine).

#### ***- Slab Gel Sequencing Systems***

These systems make use of ultrathin (75  $\mu\text{m}$ ) slab gels and involve running of at least 96 lanes per gel. In these systems, automation in sample loading of sequencing gels has also been achieved, by using a plexiglass block having wells that are the same distance apart as the comb teeth cut in a porous membrane that is used as a comb for drawing samples by capillary action.

Each well in plexiglass block is filed with a sample (PCR dideoxy-reaction mixture) so that when the porous membrane comb is lowered onto the sample wells in the plexiglass, the samples are drawn up automatically into the comb teeth by capillary action.

Automated loading of up to 192, 384 or 480 samples per gel has been achieved using the approach of employing porous combs. The porous comb with the samples is placed between the glass plates of the gel apparatus above the flat surface of the polymerised gel, and the samples are driven from the comb into the gel by electrophoresis.

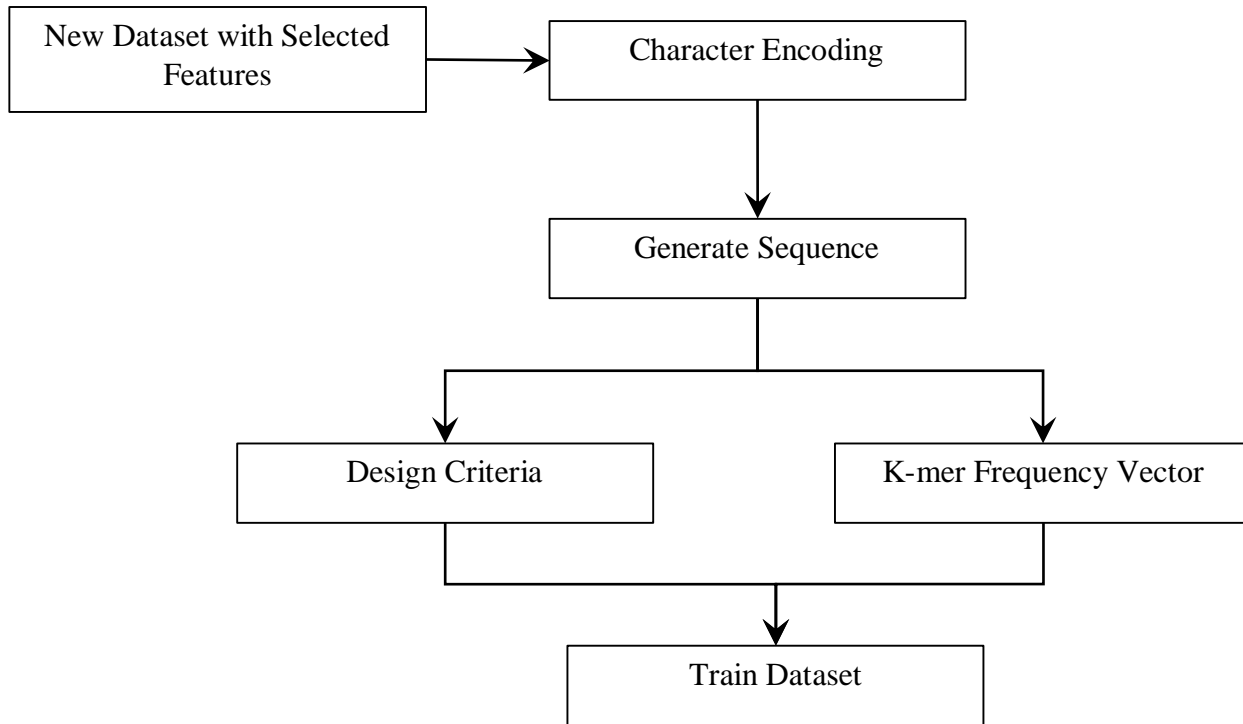
#### ***- Automatic DNA Sequencer***

A variant of the above dideoxy-method was developed, which allowed the production of automatic sequencers. In this new approach, different fluorescent dyes are tagged either to the oligonucleotide primer (dye primers) in each of the four reaction tubes (blue for A, red for C), or to each of the four ddNTPs (dye terminators) used in a single reaction tube: when four tubes are used, they are pooled.

After the PCR reaction is over, the reaction mixture is subjected to the separation of synthesised fragments through electrophoresis. Depending upon the electrophoretic system used, whether slab gel electrophoresis or capillary electrophoresis, the following two types of automatic sequencing systems have been designed.

#### **4.4 Proposed Digital DNA Sequencing Methodology**

DNA is the biological blueprint used for building proteins and other cellular components of living organisms. It is comprised of a long stretch of adenine (A), guanine (G), cytosine (C), and thymine (T) molecules, commonly referred to as “bases” due to their chemical nature. They are also referred to as nucleotides. DNA is represented computationally by character strings containing only the characters A, G, C and T (Pedersen et al., 2012). This section explains the proposed methodology of DNA generation. Figure 4.5 shows the architecture of DNA sequence generation.



**Figure 4.5: Subsection of the proposed process and indicates the DNA Sequence Generation**

#### 4.4.1 DNA Sequence Encoding

DNA sequencing is the process of determining the nucleic acid sequence, the order of nucleotides in DNA. A synthetic DNA representation of a digital artefact is a sequence of DNA characters (A, C, G and T) used to represent a digital artefact. It is considered synthetic as it represents the content of a digital artefact rather than biologic DNA. The synthetic DNA used by Pedersen et al. (2013) is created by a reversible translation of the byte sequence of a digital artefact. A DNA representation of a digital artefact is directly useable by BLAST (Pedersen et al., 2012).

A digital artefact may be considered to be a sequence of byte values, with each byte being a sequence of four two-bit pairs. Each two-bit pair has four possible values 00, 01, 10 and 11 which can be mapped to the four DNA characters A, T, G and C. This mapping procedure generates four DNA characters for each byte in the digital artefact.

The following map shown in Table 4.1 was used by Pedersen et al. (2013).

**Table 4.1: DNA Character Mapping**

Binary Bit	DNA Character
00	T
01	G
10	C
11	A

There are twenty-four possible mappings of bit values to DNA characters, each of which provides a consistent and comparable DNA representation. The mapping that was used has the property that the values for G and C and A and T are bit complements of each other. The mapping was chosen to reflect the fact that G and C and A and T are paired biologically (Watson and Crick, 1993).

For example, the character “B” has an ASCII character code of 66, which is 01000010 in binary and has the synthetic DNA representation “GTTC” based on the above mapping procedure. A file whose first two bytes are ASCII “BB” has a synthetic DNA representation starting with “GTTCGTTC”.

Synthetic DNA is fully reversible to the original artefact. A reverse lookup table can be constructed which maps “GTTC” to its associated byte value of 66, and so on. A straight forward reversal method is needed to process the synthetic DNA four characters at a time using a reverse lookup table to obtain the original byte values.

In this research work, the DNA Sequence is generated using the following mapping procedure shown in Table 4.2. The proposed mapping is suggested as an enhancement to work done by Pederson et al. and alters the mappings of the binary bits to associated DNA character and the enhancement is evaluated using tests.

**Table 4.2: Proposed DNA Character Mapping**

Binary Bit	DNA Character
00	A
01	C
10	G
11	T

For example, consider the sample dataset shown in Table 4.3. The table combines two attributes to generate a DNA character. Consider the first record in Table 4.3, sample mapping of the first record is shown in Table 4.4.

**Table 4.3: Sample Data**

A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11
1	0	0	1	0	0	1	0	1	0	0
0	0	1	1	1	0	0	1	1	0	1
0	1	0	0	0	1	0	1	0	1	0
1	0	0	1	0	1	1	0	0	1	1
0	1	1	1	1	0	0	0	1	1	0

**Table 4.4: Sample Mapping**

Attribute-1	Attribute-2	Binary Bits	DNA Character
A1	A2	10	G
A2	A3	00	A

A3	A4	01	C
A4	A5	10	G
A5	A6	00	A
A6	A7	01	C
A7	A8	10	G
A8	A9	01	C
A9	A10	10	G
A10	A11	00	A

The sample dataset contains 11 attributes; it generates a length of 10 character DNA Sequence shown in Table 4.5.

**Table 4.5: DNA representation**

DNA Sequence
GACGACGCGA
ACTTGACTGC
CGAACGCGCG
GACGCTGACT
CTTTGAACTG

#### 4.4.2 DNA Sequence Design Criteria

The aim of DNA sequence design is done by satisfying constraints to avoid such unexpected molecular reactions and is considered to be an approach of control. Good DNA sequences are

designed by using constraints such as Continuity, H-measure, GC content and Melting Temperature, among others (Xiao et al., 2009). This section explains and describes three constraints for DNA sequences design.

**- *T<sub>m</sub> Constraint***

DNA denaturing or DNA denaturation is the process of breaking down double-stranded DNA into single strands. A double-stranded DNA is broken down by heating the DNA solution to a predetermined temperature, which causes the double-stranded DNA to unwind. Furthermore, the hydrogen bonds that hold the two strands together weaken and finally break. Melting temperature ( $T_m$ ) is referred to as the temperature at which the DNA strands are breaks down into single strands (Wikibooks, 2017).

A very critical factor for DNA encoding is melting temperature. GC% method (Wetmur, 1991), the nearest neighbour mode (SantaLucia, 1998) are a few methods used to compute melting temperature. A proposed research work uses a simple method that assigns 4°C to each G-C pair and 2°C to each A-T pair.  $T_m$  is the sum of these values for all individual pairs in a DNA double-strand. The process takes into account that the A-T bond is stronger than the G-C bond. The following formula is used to compute  $T_m$ .

$$T_m = 2^{\circ}\text{C}(A + T) + 4^{\circ}\text{C}(G + C) \quad (4.1)$$

Consider the sequence ‘GACGACGCGA’,

$$2^{\circ}\text{C}(3 + 0) + 4^{\circ}\text{C}(4 + 3)$$

$$2^{\circ}\text{C}(3) + 4^{\circ}\text{C}(7)$$

$$6^{\circ}\text{C} + 28^{\circ}\text{C}$$

$$T_m = 34^{\circ}\text{C}$$



**- GC Content Constraint**

The percentage of G and C in a DNA sequence is known as the GC Content (GCC). GC content can reduce the probability of non-specific hybridisation occurring effectively and affects the thermodynamic properties of DNA.

GC content is usually calculated as a percentage value and sometimes referred to as GC-ratio or G+C ratio. The formula for calculating the GC-content percentage is shown below:

$$GCC = \frac{G+C}{A+T+G+C} \times 100\% \quad (4.2)$$

Consider the sequence 'GACGACGCGA',

$$\frac{4 + 3}{3 + 0 + 4 + 3} \times 100\%$$

$$\frac{7}{10} \times 100\%$$

$$GCC=70\%$$

**- AT\_GC Ratio Constraint**

AT\_GC ratio can be calculated as,

$$\frac{A+T}{G+C} \quad (4.3)$$

Consider the sequence 'GACGACGCGA',

$$\frac{3 + 0}{4 + 3}$$

$$\frac{3}{7}$$

$$AT\_GC \text{ ratio } = 0.429$$

Table 4.6 shows the sample DNA sequence constraints

**Table 4.6: DNA Sequence Constraints**

DNA Sequence	TM	GCC	AT_GC
GACGACGCGA	34	70	0.429
ACTTGACTGC	30	50	1
CGAACGCGCG	36	80	0.25
GACGCTGACT	32	60	0.667
CTTTGAACTG	28	40	1.5

#### 4.4.3 K-mer Frequency

K-mers are sub-sequences of length  $k$  contained within a biological sequence and is used in the field of bioinformatics. K-mers are composed of nucleotides bases and is used in the field of genome and sequence analysis to assemble DNA sequences. Typically, the term  $k$ -mer refers to all of a sequence's subsequences of length  $k$  such that the sequence AGAT would have four monomers (A, G, A, and T), three 2-mers (AG, GA, AT), two 3-mers (AGA and GAT) and one 4-mer (AGAT).

The feature vector  $F_k(s)$  for an input DNA sequences was constructed from the number of occurrences of all  $4^k$  possible  $k$ -mers (given the nucleotide alphabet  $\{A, C, G, T\}$ ), divided by the total length of  $s$ . Any ambiguous nucleotide codes (e.g., 'N' for completely ambiguous nucleotides) were removed from  $s$  before computing  $F_k(s)$ . As a concrete example, suppose  $s = \text{GACGACGCGA}$  and  $k = 2$ . Then, if use the arbitrary order [AA, AC, AG, AT, CA, CC, CG, CT, GA, GC, GG, GT, TA, TC, TG, TT] for 2-mers, the  $k$ -mer frequency vector for  $s$  is [0, 2, 0, 0, 0, 0, 3, 0, 3, 1, 0, 0, 0, 0, 0, 0] and thus

$$F_k(s) = [0, 0.2, 0, 0, 0, 0, 0.3, 0, 0.3, 0.1, 0, 0, 0, 0, 0, 0]$$

Table 4.7 shows the sample  $k$ -mer frequency vector

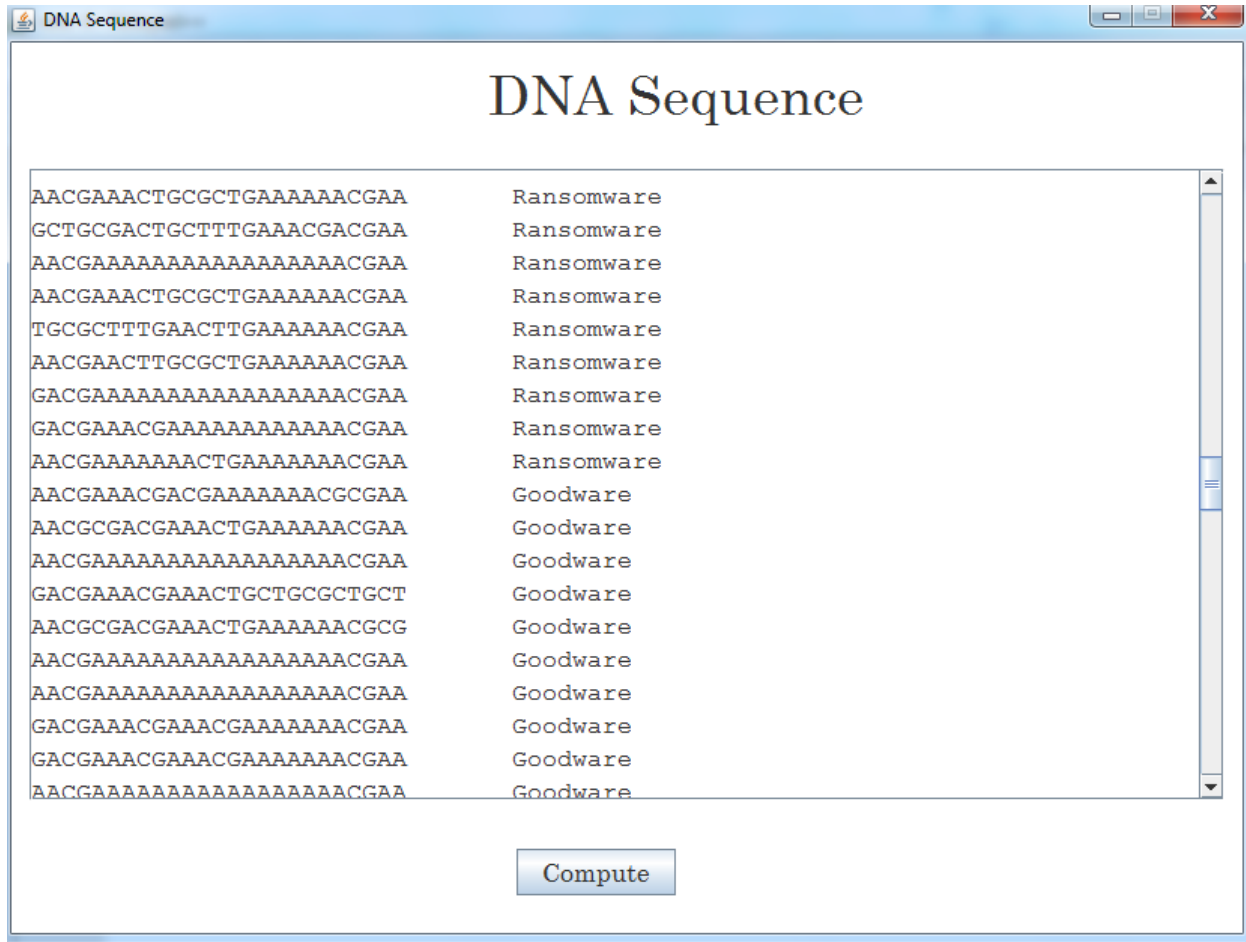
**Table 4.7: Sample K-mer Frequency**

k-mer	DNA Sequences				
	GACGA CGCGA	ACTTG ACTGC	CGAAC GCGCG	GACGCT GACT	CTTTGA ACTG
AA	0	0	0.1	0	0.1
AC	0.2	0.2	0.1	0.2	0.1
AG	0	0	0	0	0
AT	0	0	0	0	0
CA	0	0	0	0	0
CC	0	0	0	0	0
CG	0.3	0	0.4	0.1	0
CT	0	0.2	0	0.2	0.2
GA	0.3	0.1	0.1	0.2	0.1
GC	0.1	0.1	0.2	0.1	0
GG	0	0	0	0	0
GT	0	0	0	0	0
TA	0	0	0	0	0
TC	0	0	0	0	0
TG	0	0.2	0	0.1	0.2
TT	0	0.1	0	0	0.2

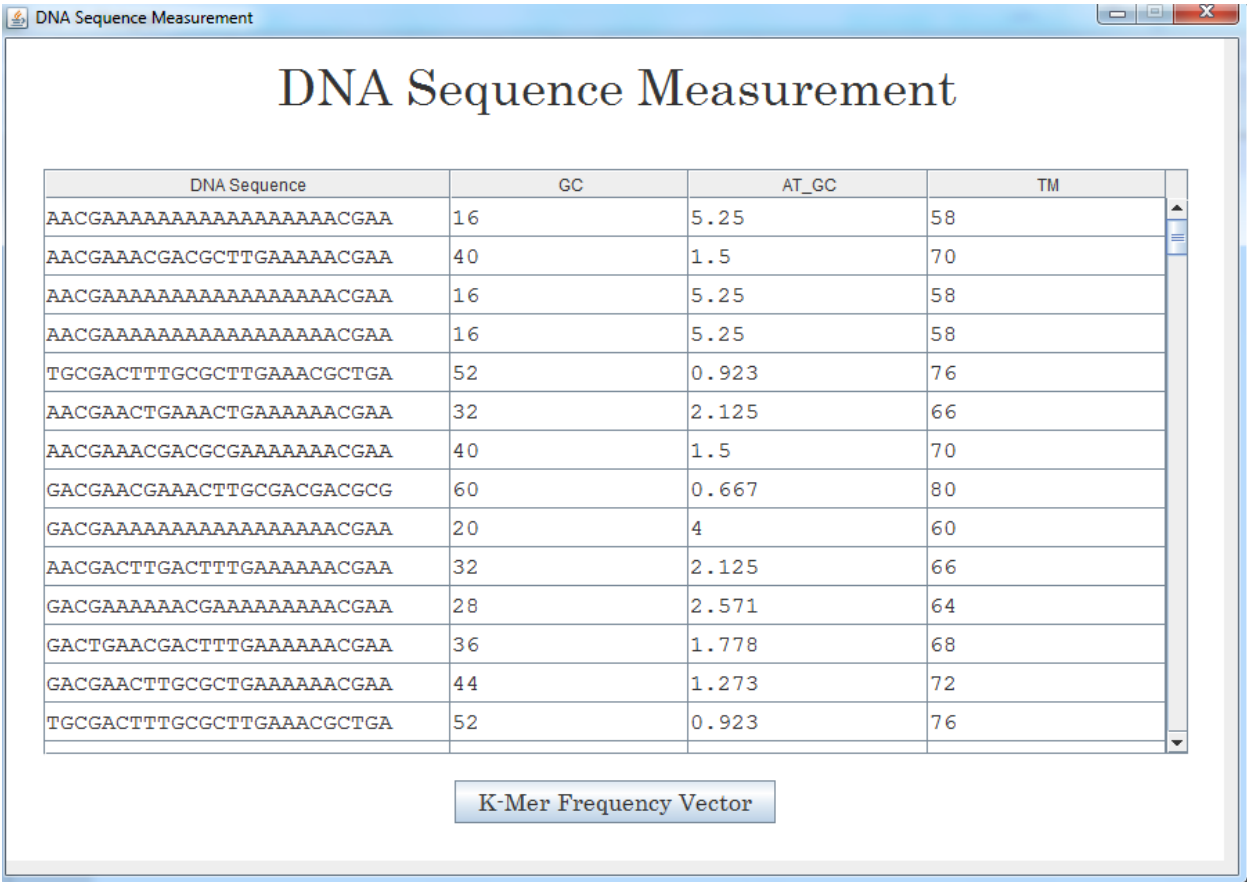
The new dataset is created based on the design criteria and frequency vector for further process.

## 4.5 Product: DNA Sequence Generation

The following section highlights Digital DNA Sequence Generation. Figure 4.6 shows a screen of DNA sequence based on the dataset derived after the preprocessing steps of the previous chapter. Furthermore, figure 4.7 highlights DNA sequence measurement, which includes GC, AT\_GC and TM computations.



**Figure 4.6:** Screenshot of DNA Sequence based on the dataset derived after the preprocessing steps



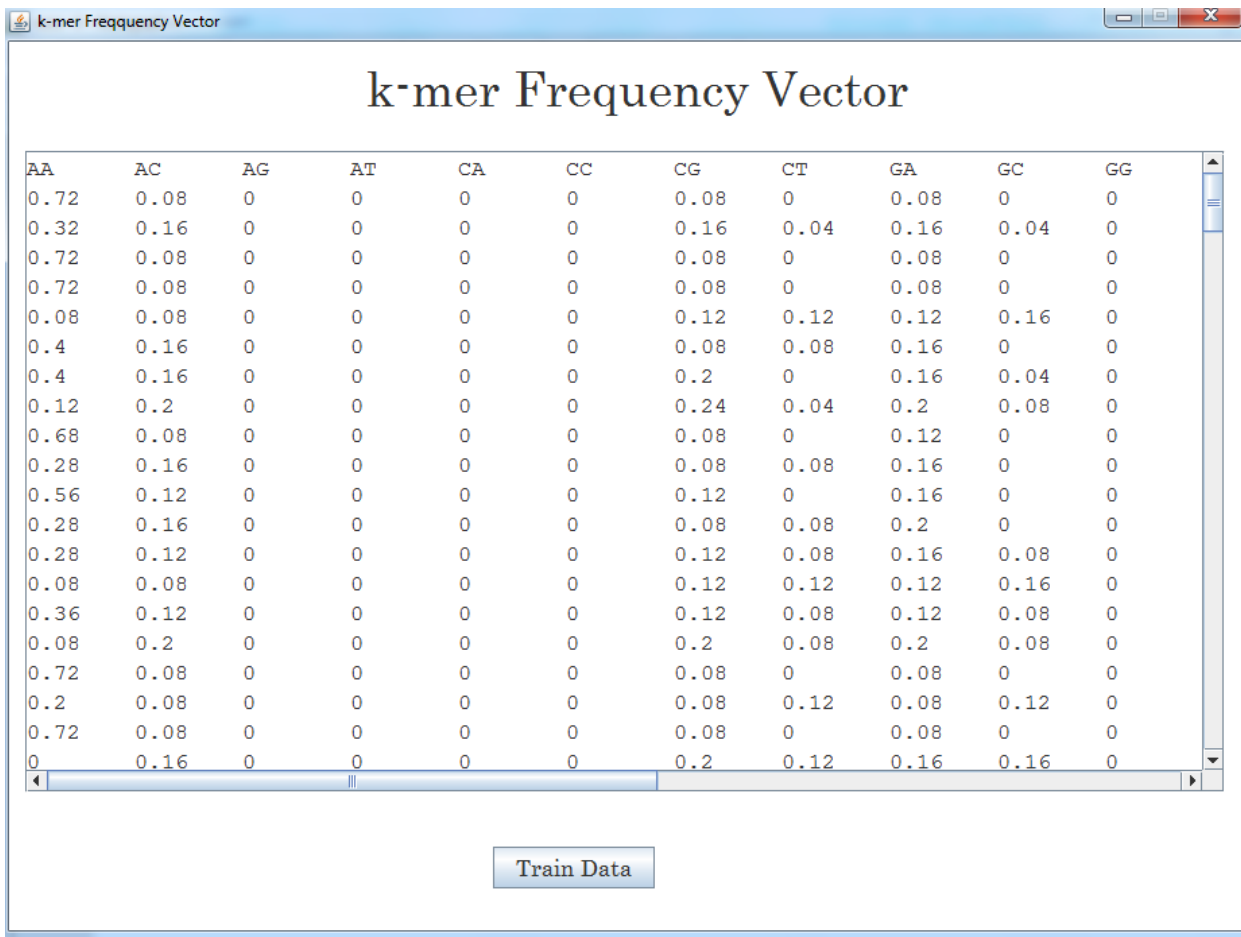
**DNA Sequence Measurement**

DNA Sequence	GC	AT_GC	TM
AACGAAAAAAAAAAAAAAAAACGAA	16	5.25	58
AACGAAACGACGCTTGAAAAACGAA	40	1.5	70
AACGAAAAAAAAAAAAAAAAACGAA	16	5.25	58
AACGAAAAAAAAAAAAAAAAACGAA	16	5.25	58
TGCGACTTTGCGCTTGAAACGCTGA	52	0.923	76
AACGAACTGAACTGAAAAACGAA	32	2.125	66
AACGAAACGACGCGAAAAACGAA	40	1.5	70
GACGAACGAACTTGCGACGACGCG	60	0.667	80
GACGAAAAAAAAAAAAAAAAACGAA	20	4	60
AACGACTTGACTTTGAAAAACGAA	32	2.125	66
GACGAAAAACGAAAAAACGAA	28	2.571	64
GACTGAACGACTTTGAAAAACGAA	36	1.778	68
GACGAACTTGCGCTGAAAAACGAA	44	1.273	72
TGCGACTTTGCGCTTGAAACGCTGA	52	0.923	76

K-Mer Frequency Vector

**Figure 4.7: Screenshot indicating DNA Sequence Measurement that includes GC, AT\_GC and TM computations.**

Figure 4.8 shows the k-mer frequency vector computation results, and consequently, figure 4.9 shows trained dataset based on Digital DNA measurement and k-mer frequency vector.



**Figure 4.8:** Screenshot indicating the k-mer Frequency Vector computation in the selected features



## **Chapter 5:**

### **Ransomware Detection**

#### **5.1 Introduction**

Ransomware is one of the most dangerous malware that affects the user's data by encrypting, modifying or deleting it or block access to the files. The main intention of the attackers is to get money from the victim. Some ransomware locks the desktop screen while other ransomware uses the cryptographic technique to encrypt the victim's files and make them inaccessible and demands a payment to decrypt the encrypted file.

The traditional ways to detect ransomware in antivirus software are through matching binary patterns and monitoring APIs. However, these methods are signature-based. If ransomware changes their behaviour or uses packers to camouflage, they are no longer being seen until the antivirus software updates. That is, the antivirus software cannot defend new and unique attacks. Besides, the former one has a limit that it cannot be detected immediately when the infection starts. This chapter describes a machine learning approach for ransomware detection.

The proposed active learning-based ransomware detection is explained in this chapter. A new dataset is formed based on DNA sequence design constraints and k-mer frequency which is used for training purpose. The new dataset is trained using active learning concept, and the test data is generated by random DNA sequences method, and subsequently, appropriate sequences are selected. The test data is classified as either ransomware or goodware using a learning algorithm.

This chapter presents in detail a ransomware detection using active learning. Section 5.2 explains machine learning methods for malware detection. Section 5.3 explains the concept of active learning. Section 5.4 analyses the ransomware. The proposed ransomware detection is described in section 5.5; it includes the learning algorithm for detection and analysing the type of ransomware



family. Section 5.6 shows screenshots and sample code of the proposed product, and finally, section 5.7 gives a summary of this chapter.

## **5.2 Machine Learning for Malware Detection**

The word malware is derived from the term malicious software which is created to meet the needs of a malicious attacker to do harmful intent. Malware is designed to do nefarious activities like compromising computers and smart devices, procure confidential data, breach into networks, and shutdown critical infrastructures, among others. Malware is generally divided into different categories based on their infection methodology and behaviours (Gandotra, Bansal and Sofat, 2014).

- Virus: A program that infects other application and spreads manually
- Worm: A program that is similar to a virus except that it can spread to other systems automatically
- Trojan: A malicious program which appears as a good program but has hidden malicious code
- Rootkit: A set of tools that evade detection and provide services on an infected machine its creator
- Spyware: A program that hides and resides on a target system to steal personal information and pass to its creator.

### **5.2.1 Malware Analysis**

The analysis of a known or unknown malicious piece of code or program to determine its operational model, functionality and purpose is called as malware analysis. Conducting malware analysis is very important to protect a computer system against future attacks and helps security

software organisation to develop programs that can identify malware based on its operation and can address detection and removal of the malware. There are two types of malware analysis.

**- *Static Analysis***

Static analysis is conducted when malware is analysed for its operation and activities without running the malware. The malware is reverse engineering in a safe environment to detect patterns that can identify malware samples. The patterns can be specific string signatures, API call statements, frequency distribution, byte-sequence n-grams, opcode frequency distribution and other system patterns. To reverse engineer any malware, the malware executable file needs to be decrypted and then unpacked to conduct static analysis.

Several tools related to these requirements are available in the market. IDA Pro (Hex-rays, 2019) is a popular disassemble or debugger tool. Similarly, OllyDbg (OllyDbg, 2019) is a tool that can convert an executable and display the contents in an assembly language format. The tools highlighted above can help to look into the malware code to understand its functionalities and identify patterns of strings to detect the origin of the malware. Other tools LordPE (Aldeid, 2019) and OllyDump (OllyDump, 2019) are memory dumper program that can detect protected code located in a system's memory and dump it to a file to be processed and analysed.

**- *Dynamic Analysis***

Dynamic analysis is when malware is executed in a controlled environment to observe the behaviours of the malicious code present. The controller environment is usually either a sandbox environment or a virtual machine. The controlled environment needs to be configured with all required software that can detect and analyse the malware behaviour before conducting a dynamic analysis. Popular tool that is available for this purpose is Process Explorer (Microsoft, 2019) and Process Hackerreplace (SourceForge, 2019) for process monitoring, Process Monitor (Microsoft,

2019) and Capture BAT (Capture BAT, 2019) for registry monitoring and file system, Regshot (SourceForge, 2019) for system change detection and Wireshark (Combs, 2019) for network monitoring. The analysis operations include information flow tracking, auto-start extensibility points, function call monitoring, instruction traces and function parameter analysis.

Dynamic analysis is considered to be much more useful than static analysis as there is no need for the malware executable to be reverse-engineered. The advantage is that a malware viewer and analyser can observe the natural behaviour of a malware which is not evident while conducting a static analysis. There are, however, some issues that users can face while conducting the dynamic analysis. Malicious users are innovating with their malware designs, and an element they are including is that they change the behaviour of malware when the same is executed inside a virtual environment. Additionally, some malware may be triggered only when a specific external factor is met.

### **5.2.2 Machine Learning Concept**

A set of methods that gives computers “the ability to learn without being explicitly programmed” is known as Machine Learning. Machine learning algorithms are used to discover and formalise the fundamental principles of data seen by the computer. A wide variety of methodologies are used by machine learning to reach an optimal solution rather than a single method. Different tasks and different capacities of the machine learning algorithm are available to befit the problem at hand. Machine learning approaches are differentiated into two as supervised and unsupervised learning (Kaspersky Labs, 2019).

The goal of the supervised learning methodology is to fit the model that will produce the correct answers for new data and objects. There are two stages of supervised machine learning. In the first stage, a model is trained, and the trained model is tested against the available training data. In the

second stage, the trained model is applied to a new sample of data and predictions are computed from the model.

The structure of the data and laws used for data generation are discovered in unsupervised learning. A significant example of unsupervised machine learning is clustering. Groups of similar objects are created by splitting the dataset in the clustering process. Another example of unsupervised machine learning is representation learning. In representation, learning objects based on the low-level description are built as an informative feature set. The cost of manual labelling by experts is huge for unlabeled datasets that are available to cybersecurity vendors. Due to the high cost of manual labelling, unsupervised learning is valuable for threat detection. Efforts for manual labelling of new samples can be optimised using clustering.

The extraction of features of a high level of abstraction from low-level data is facilitated using machine learning. This unique machine learning approach is known as Deep Learning. The learning works best when a machine is used to find high-level meaning from low-level data. The suggested methodology of deep learning is useful in various applications that help humans, including computer vision, natural language processing, speech recognition, and image processing, among others. An application of deep learning methodology is ImageNet, which is used for challenges related to image recognition. The results of the tool, ImageNet, is way better than human recognition of the same data. (Kaspersky Labs, 2019).

Machine learning is a very general and fundamentally a multidisciplinary field. The research from various other fields, as highlighted below, has contributed to machine learning (Pantic, 2005):

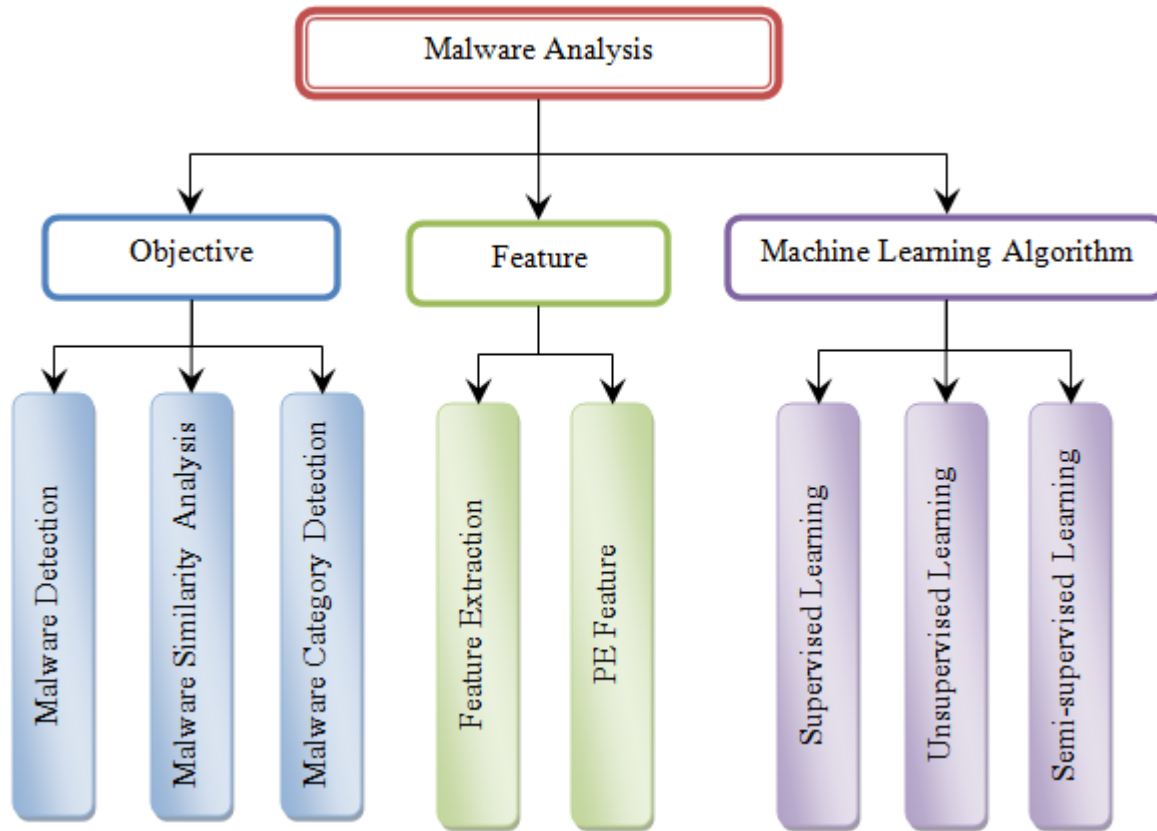
- Artificial Intelligence
- Bayesian methods.
- Computational complexity theory

- Control theory
- Information theory
- Philosophy
- Psychology
- Neurobiology

### **5.2.3 Malware Analysis using Machine learning**

This section explains the taxonomy of how machine learning is used for malware analysis. The graphical representation of the taxonomy (Ucci, Aniello and Baldoni, 2018) is shown in Figure 5.1. Machine learning contains three primary dimensions. The first dimension looks at the final objective of the analysis and is similar to the detection of malware using machine learning. The second dimension defines the features used for the machine learning process and how the features are extracted. The extraction process could be dynamic analysis, and an example of the features to be extracted could be CPU registers. Lastly, the decision to use a particular type of machine learning is covered in the third dimension. The reason being the use of supervised machine learning.

Statistical models with correct instance samples are used to gain knowledge during supervised machine learning. The knowledge gain happens during the training phase, which is the preliminary phase of machine learning. Some supervised algorithms used for malware detection are Bayesian Network, Bayes classifier, Naïve Bayes, rule-based classifier, Support Vector Machine (SVM), Multiple Kernel Learning, Prototype-based Classification, Artificial Neural Network, Decision Tree, Gradient Boosting Decision Tree Random Forest, Logistic Model Tree, k-Nearest Neighbors (k-NN) and Multilayer Perceptron Neural Network.



**Figure 5.1: Taxonomy of machine learning techniques for malware analysis.** The first dimension looks at the final objective of the analysis. The second dimension defines the features used for the machine learning process and how the features are extracted. The third dimension covers the decision to use a particular type of machine learning.

Unlabeled data are used by unsupervised approaches to learn from, rather than relying on the training phase of the machine learning algorithm. Malware detection can be conducted using multiple types of algorithms. Some of such unsupervised learning algorithms are Density-based Spatial Clustering of Applications with Noise, k-Means Clustering, Hierarchical Clustering, Clustering with locality-sensitive hashing, Prototype-based Clustering, Clustering with Distance and Similarity Metrics, Expectation-Maximization, k-Medoid, Self-Organizing Maps and Semi-supervised learning. All the algorithm mentioned can combine both unlabelled and labelled data can acquire knowledge by feeding statistical models.

### 5.3 Active Learning

The most significant issue with many machine learning applications is the effort and time required to annotate large quantities of data required for supervised learning, during the process of training a high-accuracy classifier. To solve this issue, a protocol called active learning has been proposed and designed. The active learning protocol decreases the cost by identifying highly informative data points to be annotated sequentially and be used by the learning algorithm. Pool-based active learning is another learning algorithm which is initially given access to a massive pool of unlabeled data points. These data points are considered to be inexpensive and abundant. The protocol can request the label of any unlabelled data point after selecting the same from the pool.

The algorithm works in such a way that another unlabeled data point is identified to be labelled after the initial labelling process, and this process continues. A classifier is produced once the algorithm halts after the interactive process continue for a pre-specified number of iteration (Hanneke and Yang, 2015). Passive learning is where data points are randomly chosen during the machine learning process, and this methodology contrasts with passive learning.

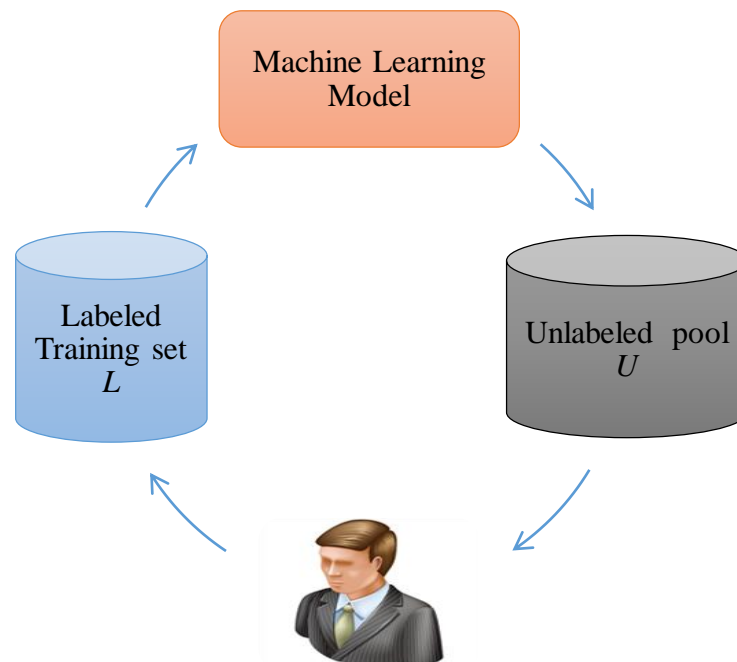
The advantage of the active learning algorithm is that all annotation effort can be directed toward the highly informative data points and can sequentially select the data points to be labelled. Additionally, active learning can reduce the total number of labels required to construct a classifier based on the information already gathered from formerly labelled data. Active learning is, therefore, capable of predicting the labels of new instances with a desired level of accuracy. The application of active learning protocol has resulted in data annotations with significant reductions in effort and time, and the protocol is currently solving various learning problems.

Active learning algorithms can be generally categorised into four groups based on usage (Hao et al., 2017):

- “Uncertainty based query strategies, where instances with the lowest prediction confidence are queried.”
- “Disagreement based query strategies, which query the instances on which the hypothesis space has the most disagreement degree on their predictions.”
- “Labelling the instances which could minimise the expected error and variance on the pool of unlabeled instances.”
- “Exploiting the structure information among the instances.”

### 5.3.1 Active Learning Cycle

An active learner may ask queries in several scenarios, and there are different query strategies to determine informative instances.



**Figure 5.2: Pool-based active learning cycle in which Labelled training set  $L$  is a small number instance. The method learns from the query results, and then the new information is taken advantage of, to choose which instances to query next.**



The pool-based active learning cycle is illustrated in figure 5.2. Labelled training set  $L$  is a small number instance with which a learner may begin and then request labels for one or more carefully selected instances. Afterwards, the method learns from the query results, and then the new information is taken advantage of, to choose which instances to query next. No additional assumptions on the part of the learning algorithm are made after the query have been made. Labelled set  $L$  is appended with the new labelled instance after which the learner continues onwards in a standard supervised way. The methodology has a few exceptions. One exception is when the learner can make other types of queries and the other when semi-supervised learning is combined with active learning.

### 5.3.2 Active Learning for Classification

Labelling instances to generate a training set is costly and time-consuming in several supervised learning tasks. Therefore, it is advantageous to find ways to reduce the number of labelled instances. A random sampling of instances is the usual training set chosen for classification. However, active learning can be employed in many cases whereby the learner can actively select the training data. A learner's need for large quantities of unlabelled data can be reduced by allowing the learner more flexibility of selecting the training data.

A large quantity of unlabeled data is readily available in many domains. The learner has access to this pool of data and can request the true class label for a specific number of instances in the pool. Finding a way to choose useful queries from the pool is considered to be a disadvantage of active learning in this setting.

Pool-based active learning can be employed in the following situations (Tong, 2001):

**Web searching:** An organisation wants to collect a list of web pages that contain the contents of people's publications. To achieve this job, the organisation assigns the task to several of its

employees to hand label the identified web pages. These labelled web pages are used to create a training set for an automatic classifier that will be used as a base set to classify and extract other web pages of similar nature from the rest of the web. Since the organisation has a limited number of employees, the organisation wishes to use the services of its employees effectively. To achieve this task, the organisation uses a computer to use active learning that requests for targeted pages that are considered to be the most informative and relevant to label rather than labelling pages at random from the web.

***Email filtering:*** An organisation's user wants to create a custom automatic junk or spam email filter that is personalised to his requirements. Machine learning is being used to achieve this task at hand. The machine learning system has access to the user's current and past emails. During the use of active learning methodologies, the system interactively displays a past email to the user and queries whether the email is spam email or not. The system then displays another email based on the previous response to the user and queries the same of the user. This process is repeated several times until an email filtering system that identifies the incoming email as good, span or junk, based on the user's personalised taste is created.

***Relevance feedback:*** A system needs to be created for a user who wishes to browse through a database or website for relevant items that which are of personal interest like articles, images and other items. This is called "I will know it when I see it" type of search. During the training of the system, the learning system displays an item from the database to the user and requests for a positive interest or negative interest response from the user. Another item is displayed to the user based on the user's response. The learning system then identifies several items in the database that it believes will be of interest to the user after a few iterations of query and responses.

Two methodologies, induction and transduction are shown in the examples above. Induction is used in the first two examples. The objective of induction is to work on unseen future instances to create a classifier. An example of transduction is shown in example three. In this example, the learning system's performance is measured against the remaining instances in the database instead of an independent test set.

### **5.3.3 Online Active Learning**

Traditional supervised online learning approach has a significant issue of the strong dependence of labelled data. This issue can be addressed by using Online Active Learning algorithms where a process of iterations are used. The algorithm works in such a way that during every iteration, one unlabeled instance is shown to the learner. The learner then has to decide whether to respond and query the label of the instance. Once the label is queried, the labelled instance updates the model by the learner and inversely if no query is done, the model is kept unchanged (Lu, Zhao and Hoi, 2016).

Label efficient learning setting and selective sampling setting are two types of settings present in the online active learning algorithm. The fixed distribution is used to derive random instances in the selective sampling setting, whereas the instances can be generated adversarially in the label efficient setting. Predictions on the instance where the label is not requested is made by the label efficient setting, whereas, generalisation error rather than the performance of the algorithm on the sequence of instances are of more concern for the selective sampling models.

### **5.3.4 Pool based active learning**

It is assumed that the learner has access to a pool of  $n$  unlabeled examples in pool-based active learning, and each labelling round can request labels for up to  $m \ll n$  examples. These  $m$  examples are selected using several methods. The  $m$  examples are requested using uncertainty sampling for

which the current hypothesis has the least confidence in classification. Another approach is to reduce the size of the version space by requesting labels for the same examples. Version space reduction is another approach that relies on predictions from hypotheses sampled from the version space, and this approach is called a Query by Committee algorithm. Labels can be further requested for the examples which are estimated to reduce training error significantly (Sculley, 2007).

There is one significant advantage of Pool-based active learning over active-online learning. There is a possibility that pool-based active learning will recognise the optimal subset of training samples, as pool-based active learning actively considers an entire dataset wholly. The greedy strategy is used by active online learning which may not select the optimal subset. However, the greedy selection is employed in many pool-based active learning methods due to the expensive nature of constructing the optimal training set.

## **5.4 Ransomware analysis**

Ransomware detection, classification and prevention are the most fundamental aspect of any research done to analyse ransomware. This section explains the static and dynamic analysis of ransomware.

A tool named HelDroid has been proposed by researchers Andronio, Zanero and Maggi (2015) to identify locker ransomware and crypto-ransomware on mobile devices. Static analysis is used by this tool for tracking purposes of the file encryption operations. A model-checking technique is used by researchers Mercaldo et al. (2016), and the proposal inspects the byte code of the malicious code in the Android platform to detect crypto-ransomware and locker ransomware. There are three main sub-processes in the proposed methodology, which include the construction of the model, extraction of temporal logic properties and finally, detection of the ransomware family. Specific strings values are searched for in this technique to discover codes related to various infection stages.

An advantage of the proposed system is that de-compilation is avoided in this technique as the byte code of the ransomware is inspected instead of the source code. CryptoDrop is another tool proposed by researchers Scaife et al. (2016) that is an early detection system. The proposed methodology and tool target static features, such as entropy measurements and content similarity, of the malicious files.

When a user analyses ransomware using a passive approach in which a code sample and its payload are examined without executing the file is called static analysis (Galal, Mahdy and Atiea, 2016). The static analysis further covers procedures like extracting the structural features from the source code and examining the binary strings that uniquely represent the malicious code (Wang and Wang, 2015). Abundant information about all potential execution methodologies of the malware sample is generated using these approaches (Miao et al., 2016).

Static analysis technique suffers several flaws even though the methodology is safe, accurate and fast in detecting previously known ransomware samples. A major flaw is that malicious code can use obfuscation techniques to change their binary structures to avoid static analysis (Banescu et al., 2015). Static analysis also falters when the methodology comes in contact with ransomware that can be created using packers which encrypt and compress the malicious payload.

The dynamic analysis may be used to overcome the disadvantages of static analysis. Dynamic analysis is a process in which malicious code is analysed while the program is being executed in a virtual environment. The execution of the ransomware is conducted to observe the actual behaviour of the malicious code and the methods used by it to interact with the operating system and kernel (Kaur and Singh, 2014). The main advantage of dynamic analysis approach is that the approach is more effective in identifying and observing the operation of the malicious code and it overcomes resilience to evasion by studying the process of the code (Nauman, Azam and Yao, 2016).

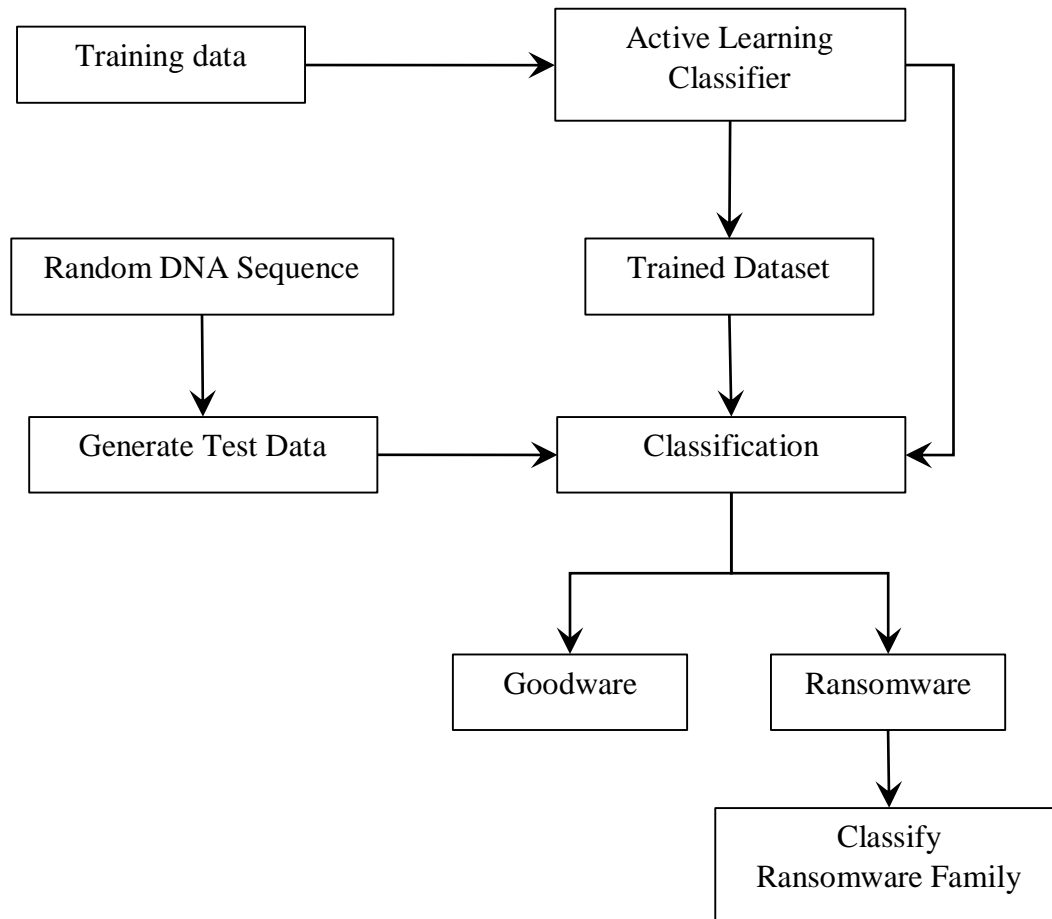
Additionally, dynamic analysis can use behavioural signatures of different ransomware families to detect unknown variants.

UNVEIL is a tool proposed by researchers Kharaz et al. (2016) that is based on dynamic analysis methodologies to detect locker ransomware and crypto-ransomware. The main algorithm of the proposed system focuses on three attributes of a system's operation which are; patterns of access, file system activities and I/O data buffers entropy. Similarly, a detection model has been proposed by researchers Song, Kim and Lee (2016), which uses dynamic analysis methodologies. The proposed system conducts ransomware detection by observing memory contents, CPU usage, input/output usage, and events related to file operations. Similarly, researchers Cabaj et al. (2015) have also used dynamic analysis methodologies by analysing the network traffic and identifying the infection chain in coordination with honeypot technologies. Additionally, a proposal by researchers Andronio, Zanero and Maggi (2015) also uses dynamic analysis to monitor network traffic from the infected host to the C&C server to identify the malicious code.

As explained before, during dynamic analysis, ransomware samples need to be executed in a controlled and safe environment. This process could be a limitation for dynamic analysis if the analyst is not careful analysing ransomware, there are possibilities of the analysis environment being infected by the ransomware. Additionally, the analysing environment is different from a production environment, as majorly virtual machines are used as the analysing environment. The runtime logs are different between the virtual machine and production machines (Shijo and Salim, 2015). Furthermore, the behaviour of ransomware may be different in production environment compared to a test environment as external factors may trigger specific actions in the code (Choudhary and Vidyarthi, 2015). The dynamic analysis may not be able to detect all the execution paths in ransomware, and therefore, malicious code can evade detection solutions put in place (Egele et al., 2012).

## 5.5 Proposed Ransomware Detection Methodology

This section explains the proposed methodology of ransomware detection using an active learning algorithm. Figure 5.3 shows the architecture of ransomware detection.



**Figure 5.3: Subsection of the proposed process and indicates Ransomware detection architecture**

The training dataset is generated based on the selected features with DNA design constraints and k-mer frequency. This dataset is trained using active learning classifier. DNA sequences are randomly generated and selected for test data. It also computes DNA constraints and k-mer frequency. The test data is classified as goodware or ransomware using an active learning classification algorithm. This section explains a detailed description.

### 5.5.1 Linear Regression Model

Regression task is adapted into supervised machine learning to propose a Linear Regression algorithm. Regression model proposed to detect the relations between forecasting and variables. Relationships between independent and dependent variables influence the type of regression models being used to address the issue on hand. Additionally, the regression model to be used is based on the number of independent variables being used.

Linear regression is used to predict a quantitative response  $Y$  from the predictor variable  $X$ . A simple linear regression model is shown below.

$$Y = \beta_0 + \beta_1 X + \epsilon$$

Where  $X$  is termed as an explanatory or independent variable and  $Y$  is termed as the study or dependent variable. The parameters of the model, known as regression coefficients, are terms  $\beta_0$  and  $\beta_1$  where the intercept term is  $\beta_0$  and the slope parameter is  $\beta_1$ . The component  $\epsilon$ , which is an unobservable error, accounts for the failure of data to stay on the straight line and characterises the variance between the true and detected realisation of  $Y$ . The effect of all deleted variables in the model is one of several reasons for such difference. The variables may be inherent randomness in the observations or qualitative.

The model aims to predict  $Y$  value by achieving the best-fit regression line. Also, the error difference between the true value and predicted value needs to be minimum. Therefore, the  $\beta_0$  and  $\beta_1$  values needs to be updated to reach the best value and minimise the error between true  $Y$  value ( $Y$ ) and predicted  $Y$  value (pred).

Root Mean Squared Error (RMSE) between true  $Y$  value ( $Y$ ) and predicted  $Y$  value (pred) is the Cost function ( $J$ ) of Linear Regression using the formula below.



$$J = \frac{1}{n} \sum_{i=1}^n (pred_i - Y_i)^2$$

### 5.5.2 Active Learning Model

A sequence of training instances is used by a learner to iteratively learns  $\{(X_t, Y_t) \mid t=1, \dots, T\}$ , where  $Y_t \in \{-1, +1\}$  is its true class label  $X_t \in \mathbb{R}^d$  is the feature vector of the  $t$ -th instance. The objective of online binary classification is to learn a linear classifier,

$$\hat{Y}_t = pre(W_t^T X_t)$$

where  $W_t \in \mathbb{R}^d$  is the weight vector at the  $t$ -th round.

When  $X_t$  is received, an online active learning algorithm needs to decide whether to query the true label  $Y_t$  or not, which is not the same as regular online supervised learning. An external expert will be asked to provide the true label if the algorithm chooses to request the true label. The algorithm may undergo some positive loss and implement traditional online learning techniques to update the model  $W_t$ , once the true label is observed.

If not met, the instance will be ignored by the algorithm and continue to process the next one. The pseudocode of the active learning algorithm for training data is presented below:

1. Initialise Learning Rate LR, Regularization Parameter RP and Smoothing Parameter SP
2. Read Train dataset Tr
3. for each instance (inst) in Tr
4.     pre= Predict the class value of inst using a linear regression model
5.     If (pre > 0)
6.         Ypre = 1
7.     Else Ypre = -1
8.     End If
9.     Compute  $r=1/RP$  ,  $v=1/pre$ ,  $c=0.5*(-LR/r+v)$
10.    Compute  $p=Abs(pre)+c$ ;

```

11.   If (p>0)
12.       Compute sm=SP/(SP+p);
13.       If (random (0, 1) < sm)
14.           Zt=true
15.       End If
16.   Else Zt=false
17.   End If
18.   If (Zt == true)
19.       Set inst class value as Ypre
20.   End If
21. End For

```

The training data is well trained using active online learning. DNA sequences are randomly generated and selected for test data. For each test DNA sequences, the design constraints and k-mer frequency are computed. The test data is predicted using a linear regression model based on active learning training data.

Consider the following training data:

```

AA,AC,AG,AT,CA,CC,CG,CT,GA,GC,GG,GT,TA,TC,TG,TT,GCC,AT_GC,TM,Cls
0.6,0.067,0,0,0,0,0.133,0,0.067,0.067,0,0,0,0,0,26.667,2.75,38,-1
0.2,0.133,0,0,0,0,0.267,0,0.2,0.133,0,0,0,0,0,60,0.667,48,-1
0.533,0,0,0,0,0,0.2,0,0.067,0.133,0,0,0,0,0,40,1.5,42,1
0,0.067,0,0,0,0,0.2,0.133,0.067,0.2,0,0,0,0,0.067,0.2,60,0.667,48,1
0,0,0,0,0,0,0.4,0.067,0,0.4,0,0,0,0,0.067,0.93.333,0.071,58,1
0.267,0.133,0,0,0,0,0.133,0.067,0.133,0.067,0,0,0,0,0.067,0.067,46.667,1.143,44,1
0.133,0.067,0,0,0,0,0.267,0.067,0.067,0.2,0,0,0,0,0.133,60,0.667,48,-1
0.2,0.067,0,0,0,0,0.267,0.067,0.067,0.2,0,0,0,0,0.067,60,0.667,48,-1
0.533,0.067,0,0,0,0,0.133,0,0.133,0.067,0,0,0,0,0,33.333,2,40,-1
0.333,0.133,0,0,0,0,0.267,0,0.067,0.133,0,0,0,0,0,53.333,0.875,46,1

```

This dataset is trained using an active learning algorithm with linear regression.

New training dataset after applying active learning algorithm

```
0.6,0.067,0,0,0,0,0.133,0,0.067,0.067,0,0,0,0,0,26.667,2.75,38,-1
0.2,0.133,0,0,0,0,0.267,0,0.2,0.133,0,0,0,0,0,60,0.667,48,-1
0.533,0,0,0,0,0.2,0,0.067,0.133,0,0,0,0,0,40,1.5,42,-1
0,0.067,0,0,0,0,0.2,0.133,0.067,0.2,0,0,0,0,0.067,0.2,60,0.667,48,-1
0,0,0,0,0,0.4,0.067,0,0.4,0,0,0,0,0.067,0.93.333,0.071,58,-1
0.267,0.133,0,0,0,0,0.133,0.067,0.133,0.067,0,0,0,0,0.067,0.067,46.667,1.143,44,-1
0.133,0.067,0,0,0,0,0.267,0.067,0.067,0.2,0,0,0,0,0.133,60,0.667,48,1
0.2,0.067,0,0,0,0,0.267,0.067,0.067,0.2,0,0,0,0,0.067,60,0.667,48,1
0.533,0.067,0,0,0,0,0.133,0,0.133,0.067,0,0,0,0,0,33.333,2,40,1
0.333,0.133,0,0,0,0,0.267,0,0.067,0.133,0,0,0,0,0,53.333,0.875,46,1
```

For test data, DNA sequences are randomly generated and selected. In the context of DNA sequence generation, the labelling cannot be done.

```
CGAGGGGCCCTACTA
GCCCCGAAGTGTCAC
GATAGCAGAACGCAC
AAAATATATATCATG
CGCATCGACGGGTTC
GCGAGCACGTAAACA
```

For each DNA sequence, the design constraints and k-mer frequency are generated.

```
AA,AC,AG,AT,CA,CC,CG,CT,GA,GC,GG,GT,TA,TC,TG,TT,GCC,AT_GC,TM,Cls
0,0.067,0.067,0,0,0.133,0.067,0.133,0.067,0.067,0.2,0,0.133,0,0,0,66.667,0.5,50,?
0.067,0.067,0.067,0,0.067,0.2,0.067,0,0.067,0.067,0,0.133,0,0.067,0.067,0,66.667,0.5,50,?
0.067,0.133,0.133,0.067,0.133,0,0.067,0,0.133,0.133,0,0,0.067,0,0,0,53.333,0.875,46,?
```

0.2,0,0,0.333,0.067,0,0,0,0,0,0,0.2,0.067,0.067,0,13.333,6.5,34,?  
 0,0.067,0,0.067,0.067,0,0.2,0,0.067,0.067,0,133,0.067,0,0.133,0,0.067,66.667,0.5,50,?  
 0.133,0.133,0.067,0,0.133,0,0.133,0,0.067,0.133,0,0.067,0.067,0,0,53.333,0.875,46,?

The prediction result for test data is,

CGAGGGGCCCTACTA	Goodware
GCCCCGAAGTGTCAC	Goodware
GATAGCAGAACGCAC	Ransomware
AAAATATATATCATG	Ransomware
CGCATCGACGGGTTC	Goodware
GCGAGCACGTAAACA	Ransomware

### 5.5.3 Ransomware Family Classification

After learning the DNA sequence in ransomware, the family of the ransomware needs to be analysed. This section explains the ransomware family classification using machine learning. The ransomware families are identified by the following code, as shown in Table 5.1.

**Table 5.1: Ransomware Family Code**

Family	ID
Goodware	0
Critroni	1
CryptLocker	2
CryptoWall	3
KOLLAH	4
Kovter	5

<b>Locker</b>	6
<b>MATSNU</b>	7
<b>PGPCODER</b>	8
<b>Reveton</b>	9
<b>TeslaCrypt</b>	10
<b>Trojan-Ransom</b>	11

Random Forest, Sequential Minimal Optimization (SMO), and Naïve Bayes are machine learning algorithm that is used to analyse the ransomware family.

#### **- *Random Forest***

Breiman (2001) developed an algorithm for classification named Random forest that uses a group of classification trees. Each of the classification trees is built using a bootstrap sample of the data, and the candidate set of variables, is a random subset of the variables, at each split. Consequently, random forest uses a successful approach for combining unstable learners, called bagging, and random variable selection for tree building. Low-bias trees are obtained using each unpruned tree. Additionally, random variable selection and bagging result in a low correlation of the individual trees. The result of the algorithm is that a group is created that can achieve both low variance and low bias (Díaz-Uriarte and De Andres, 2006).

Tree-structured classifiers are combined to create a Random Forest. In the algorithm, every tree of the forest provides a unit vote, to define a most probable class by assigning each input. The algorithm is not susceptible to noise; methodology is fast and non-linear patterns of data can be identified and successfully grouped. Additionally, categorical and numerical data can be easily

handled. A significant advantage of Random Forest is that even if more trees are added to the forest, it does not suffer from overfitting (Chaudhary, Kolhe and Kamal, 2016).

The pseudocode for random forest classification (Kamath et al., 2016) as follows:

**Step1:** Draw  $n_{tree}$  bootstrap samples from the original data.

**Step2:** For each of the samples, produce an unpruned classification tree, with the following modification:

At each node, rather than choosing the best split among all predictors, randomly sample  $n_{try}$  of the predictors and choose the best split from among those variables.

**Step3:** Predict new data by aggregating the predictions of the  $n_{tree}$  trees.

$n_{tree}$  specifies how many trees are to be built to populate the random forest.

$n_{try}$  specifies the number of variables that will be considered at any time in deciding how to partition the dataset.

### - *Sequential Minimal Optimization*

Support Vector Machine (SVM) is trained using an algorithm called Sequential Minimal Optimisation (SMO), and the algorithm can be used to quickly solve the Support Vector Machine Quadratic Problem (SVM QP) (Platt, 1998). The SVM QP is solved without using numerical QP optimisation steps and without any extra matrix storage. The overall QP problem is divided into QP sub-problems using SMO. A time consuming numerical QP optimisation as an inner loop is not needed to solve these small QP problems. SMO can handle extensive training sets due to the linear amount of memory required for SMO training sets.

The algorithm involves three essential components:

- An analytic solution to the optimisation problem for the two chosen multipliers

- A heuristic for selecting the two multipliers to optimise at a given step
- A step to compute the offset

### - *Naïve Bayes*

The Naive Bayes algorithm is also termed as a simple probabilistic classifier. The algorithm counts the combinations of values and frequency in a given dataset and computes a set of probabilities. Given the value of the class variable and assuming all attributes to be independent, the algorithm uses Bayes theorem. The Naive characterisation is due to the property of the algorithm of that conditional independence assumption does not sustain in real-world applications, yet the algorithm performs well and learns rapidly while being used with different supervised classification problems.

The pseudocode for naïve Bayes classification (Saputra, Widiyaningtyas and Wibawa, 2018) as follows:

**Step1:** Read training dataset

**Step2:** Calculate the mean and standard deviation of the predictor variables in each class.

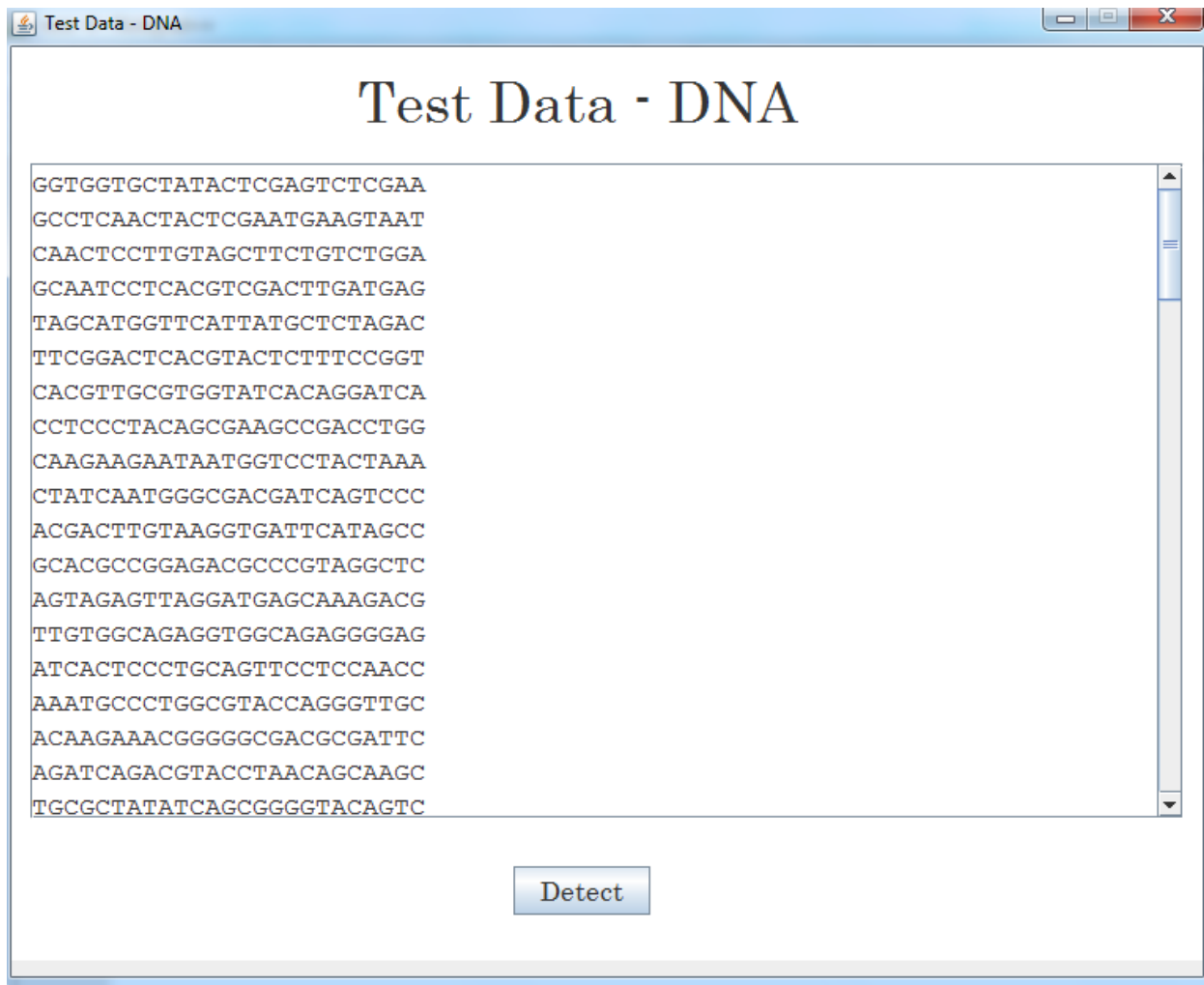
**Step3:** Calculate the probability using gauss density equation in each class until the probability of all predictor variable has been calculated.

**Step4:** Calculate the likelihood for each class

**Step5:** Get the highest likelihood.

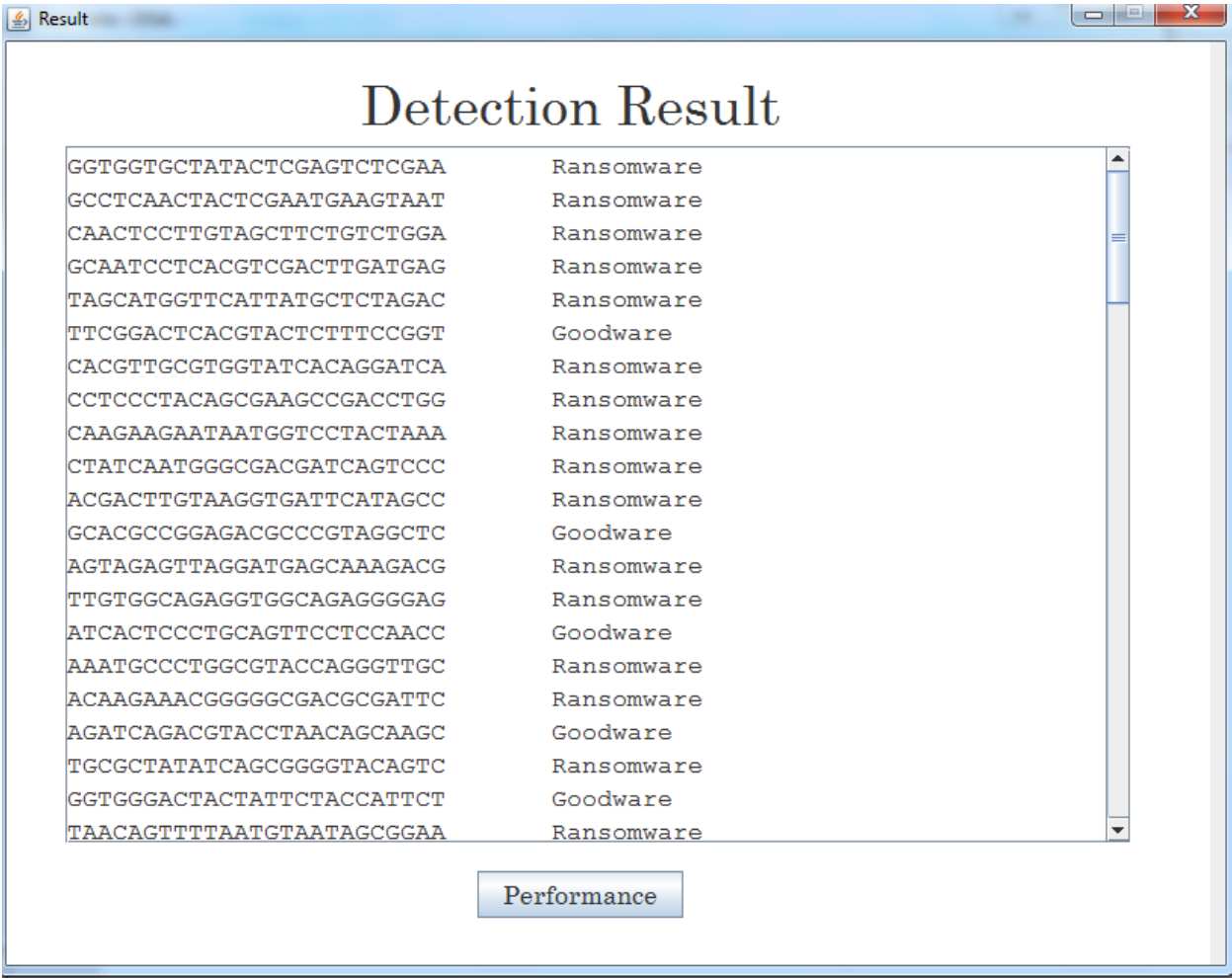
## 5.6 Product: Ransomware Detection

This section shows some sample screenshots of the product being developed. Figure 5.4 shows test data based on randomly generated and selected DNA sequence. Additionally, figure 5.5 shows the detection results for the test data.



**Figure 5.4:** Screenshot to show the test data DNA that is randomly generated and selected DNA sequence





Detection Result	
GGTGGTGCTATACTCGAGTCTCGAA	Ransomware
GCCTCAACTACTCGAATGAAGTAAT	Ransomware
CAACTCCTTGCTAGCTTCTGTCTGGA	Ransomware
GCAATCCTCACGTCGACTTGATGAG	Ransomware
TAGCATGGTTCATTATGCTCTAGAC	Ransomware
TTCGGACTCACGTACTCTTTCCGGT	Goodware
CACGTTGCGTGGTATCACAGGATCA	Ransomware
CCTCCCTACAGCGAAGCCGACCTGG	Ransomware
CAAGAAGAATAATGGTCCTACTAAA	Ransomware
CTATCAATGGGCGACGATCAGTCCC	Ransomware
ACGACTTGTAAGGTGATTCATAGCC	Ransomware
GCACGCCGAGACGCCCCGTAGGCTC	Goodware
AGTAGAGTTAGGATGAGCAAAGACG	Ransomware
TTGTGGCAGAGGTGGCAGAGGGGAG	Ransomware
ATCACTCCCTGCAGTTCCTCCAACC	Goodware
AAATGCCCTGGCGTACCAGGGTTGC	Ransomware
ACAAGAAACGGGGGCGACGCGATTG	Ransomware
AGATCAGACGTACCTAACAGCAAGC	Goodware
TGCGCTATATCAGCGGGGTACAGTC	Ransomware
GGTGGGACTACTATTCTACCATTCT	Goodware
TAACAGTTTTAATGTAATAGCGGAA	Ransomware

Performance

**Figure 5.5:** Screenshot to display the final detection results that classify the test data to either ransomware or goodware.

## 5.7 Summary

This chapter described an approach for detecting the sample data as either goodware or ransomware based on machine learning techniques based active learning algorithm. It uses a linear regression model with active learning algorithm for effective training of data. Based on the well-trained dataset, the randomly generated and selected DNA sequence is efficiently predicted.

## Chapter 6:

### Results and Discussions

#### 6.1 Introduction

This section clarifies the performance assessment of ransomware detection using an active learning algorithm. These research experiments are created with the help of Java (version 1.8). The real-world dataset is taken from the Internet for experiments. Initially, the dataset is preprocessed, and the particular features are extracted using GWO and cuckoo search algorithm. Subsequently, DNA sequences are generated based on the selected features, and DNA design constraints are computed, k-mer frequency is used to generate the new train dataset. Finally, the active learning algorithm is used to detect the ransomware.

In this chapter, section 6.2 explains the description of the dataset and section 6.3 explains evaluation metrics, Section 6.4 provide the experimental result of the research work, and finally, section 6.5 gives the summary of the work.

#### 6.2 Dataset Description

The ransomware dataset is collected from <https://github.com/PSJoshi/Notes/wiki/Datasets>. The dataset contains 1524 records and 30970 features with 582 ransomware and 942 goodware applications. In this dataset, only 300 records and 16383 features with 150 ransomware and 150 goodware applications are taken for preprocessing step. Table 6.1 shows sample features from the dataset.

**Table 6.1: Dataset- Sample Features**

Sample Features					
ID,	Label,	Ransomware	Family,	API:GetSystemDirectoryA,	API:NtOpenFile,
API:EnumWindows,	API:CreateThread,	API:InternetOpenW,	API:GetUserNameW,		

API:send, DROP:801, DROP:JPG, DROP:clf, DROP:xls, DROP:ucd, DROP:conf, DROP:xsd, DROP:sig, REG:DELETED, REG:OPENED, REG:READ, REG:WRITTEN, FILES:DELETED, FILES:OPENED, FILES:READ, FILES:WRITTEN, FILES\_EXT, DIR:CREATED, DIR:ENUMERATED, STR

The set of features are identified with the following codes is shown in Table 6.2.

**Table 6.2: Features Description Code**

ID	Description
API	API invocations
DROP	Extensions of the dropped files
REG	Registry Key operations
FILES	File operations
FILES_EXT	Extension of the files involved in file operations
DIR	File directory operations
STR	Embedded strings

Table 6.3 shows sample dataset with sample features used for the experiment.

**Table 6.3: Sample Dataset with only 40 features**

Sample Dataset																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
10001	Ransomware	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0</

10028,Ransomware,1,0,1,0,0,1,1,0,1,0,0,0,1,1,0,1,0,0,0,0,1,1,0,0,1,1,0,0,1,0,1
10050,Ransomware,0,0,1,0,1,1,0,0,0,0,1,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,0,1,1,0,0
20001,Goodware,0,0,0,0,0,1,1,0,1,0,0,1,0,0
20002,Goodware,1,0,1,0,0,0,1,0,1,0,0,1,0,0
20032,Goodware,0,0,1,0,1,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0
20098,Goodware,1,0,1,0,1,0,0,0,0,0,1,0,0,1,1,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0
20122,Goodware,0,0,1,0,1,0,1,1,0,0,1,0,1,0,0,1,0,0

### 6.3 Evaluation Metrics

This section explains the evaluation metrics used for the experiment. The evaluation metrics are Confusion Matrix, Positive and Negative Rate, Precision, Recall and F-Measure.

#### 6.3.1 Confusion Matrix

The performance of the classification is computed using the confusion matrix. Error matrix and contingency table are other names of a confusion matrix. The matrix is used to check the performance of a proposed algorithm and has a fixed table layout.

A confusion matrix holds data about predicted and actual classifications. Data in the matrix is used to evaluate the performance of such systems. A two-class classifier is represented in the following table as a confusion matrix (Kohavi and Provost, 1998).

The confusion matrix signifies the following sentences in the context of our study:

- “*a* is the number of **correct** predictions that an instance is **positive**,”
- “*b* is the number of **incorrect** predictions that an instance **negative**,”
- “*c* is the number of **incorrect** predictions that an instance is **positive**, and”

- “ $d$  is the number of **correct** predictions that an instance is **negative**.”

**Table 6.4: Sample Confusion Matrix**

		Predicted	
		Positive	Negative
Actual	Positive	a	c
	Negative	b	d

### 6.3.2 Positive and Negative Rate

#### - False Positive Rate (FP)

When a test falsely or incorrectly reports a positive result, it produces a false-positive result. For example, when real ransomware is predicted as a goodware.

$$FP = \frac{c}{c + d}$$

#### - False Negative Rate (FN)

When a test falsely or incorrectly reports a negative result, then a false negative result occurs. For example, a prediction result may return a negative result signifying that the application is not ransomware even though the application is ransomware.

$$FN = \frac{b}{a + b}$$

#### - True Positive Rate (TP)

True positive is the number of applications being detected as ransomware, and it is correctly predicted as ransomware.

$$TP = \frac{a}{a + b}$$

**- True Negative Rate (TN)**

True negative is the number of applications detected as goodwill, and it is correctly predicted as goodwill.

$$TN = \frac{d}{c + d}$$

**6.3.3 Recall, Precision, Accuracy and F-measure**

The results from each of the perspectives of accuracy, precision, recall, and F-measure are analyzed since each serves different purposes. Precision and Recall are two significance performance measures for evaluating classification algorithms (Cios et al, 1998). In this experiment, Precision refers to the proportion of data which is classified correctly using the classification algorithm. Here, Recall refers to the percentage of information which is relevant and is correctly classified. Accuracy is the percentage of instances which is classified correctly by classifiers (Han et al., 2011). F-Measure is another performance measure which combines Recall and Precision into a single measure (Kumar and Rathee, 2011) and is commonly used in classification.

**- Recall (R)**

Recall (R) is defined as the ratio of correctly predicted applications to the sum of correctly predicted applications plus false negative.

$$R = \frac{TP}{TP + FN}$$

**- Precision Rate (P)**

Precision is defined as the proportion of the true positives against all the positive results (both true positives and false positives).

Precision rate (P) is calculated as the ratio of correctly predicted applications to the sum of correctly predicted applications in addition to false positives.

$$P = \frac{TP}{TP + FP}$$

**- Accuracy (A)**

Overall accuracy is calculated using

$$A = \frac{TP + TN}{TP + TN + FP + FN}$$

**- F-Measure (F)**

F-Measure (F) is calculated using precision and recall.

$$F = 2 * \frac{P * R}{P + R}$$

**6.4 Reliability Test**

Reliability test is essential in ensuring that the selected features are consistent. Cronbach's alpha is widely used to measure internal consistency, also called the reliability of the features selected during the selection process. The reliability of the features determines whether the entire process of selecting the desired features is reliable and can be applied in the future by other researchers (Cooper and Schindler, 2014). For the current study, the reliability test for the features based on Cronbach's alpha was conducted to determine the internal consistency of the features selected. A Cronbach alpha of more than 0.7 for a given data is considered to have a high degree of reliability, as stated by Cooper and Schindler (2014).

**Cronbach's alpha Test Result**

The selected feature list are indicated as below

Att_4	Att_11	Att_38	Att_43	Att_59
Att_60	Att_73	Att_79	Att_82	Att_101

Att_103	Att_106	Att_115	Att_128	Att_151
Att_188	Att_195	Att_196	Att_211	Att_217
Att_218	Att_238	Att_250	Att_255	Att_258
Att_292	Att_301	Att_314		

The dataset is taken after feature selection, and the variance and covariance are computed. The formula for computing the Cronbach's alpha is as shown below:

$$\alpha = \frac{N\bar{c}}{\bar{v} + (N - 1)\bar{c}}$$

Where the values used for computation are as below:

$$N = 28$$

$$c = 0.023081711803096497$$

$$v = 0.13959407426744114$$

The calculations produce a Cronbach's alpha as being 0.8472570559541961

The overall Cronbach's Alpha ( $\alpha$ ) was 0.847, which is more than 0.35. The results show a high level of reliability and therefore, strong internal consistency of the selected features.

## 6.5 Experimental Results

### 6.5.1 Feature Selection Result

This section analyses preprocessing and feature selection results. Feature selection is used to eliminate redundant, noisy and irrelevant features, to reduce the number of features, and select the most representative ones. Searching for the best set of features is a challenging and complex issue



due to the vast search space in scenarios where the number of features is extensive (Faris et al., 2018).

Table 6.5 shows the features count before and after preprocessing. In that dataset, most of the column values are '0', and these features are not useful so it must be removed in the preprocessing step.

**Table 6.5: Features Count**

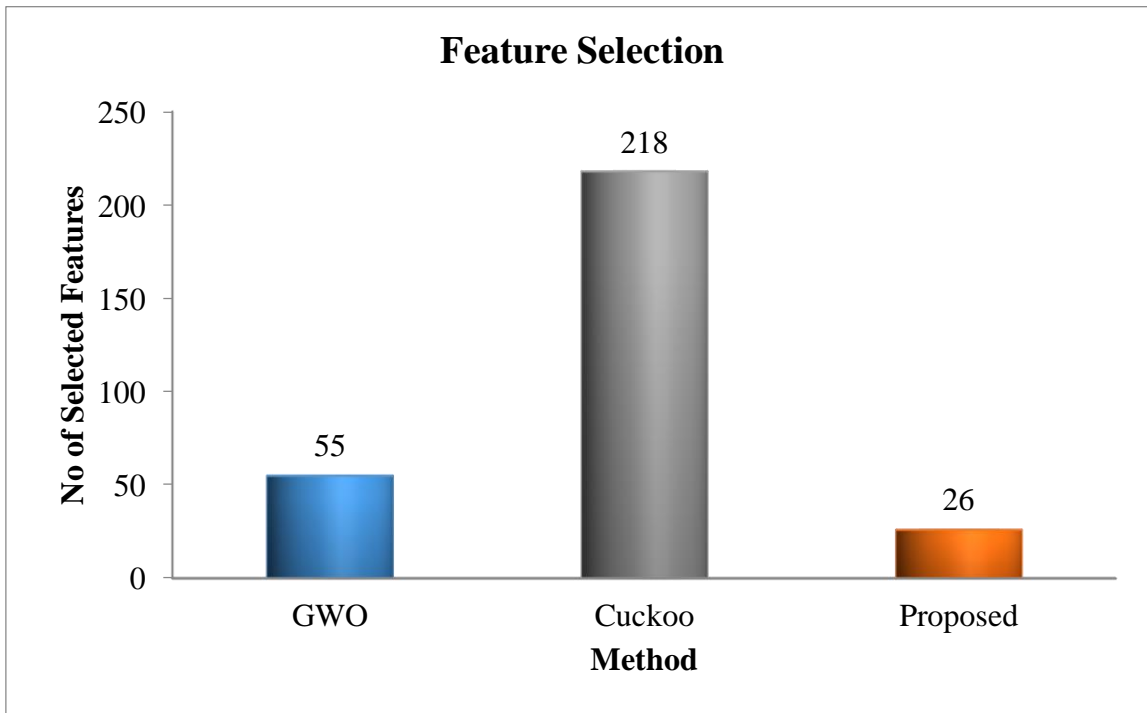
Number of Features	
Before Preprocessing	After Preprocessing
16383	426

Table 6.6 shows the number of selected features for the different methods

**Table 6.6: Feature Selection**

Method	Features
GWO	55
Cuckoo Search	218
Proposed	26

Figure 6.6 shows the feature selection comparison of GWO, cuckoo search and proposed method. In that comparison, GWO and proposed methods reduce more features compared to the cuckoo search method. The proposed method selects the features based on the intersection of GWO and cuckoo search. Only common features are selected for the next process.

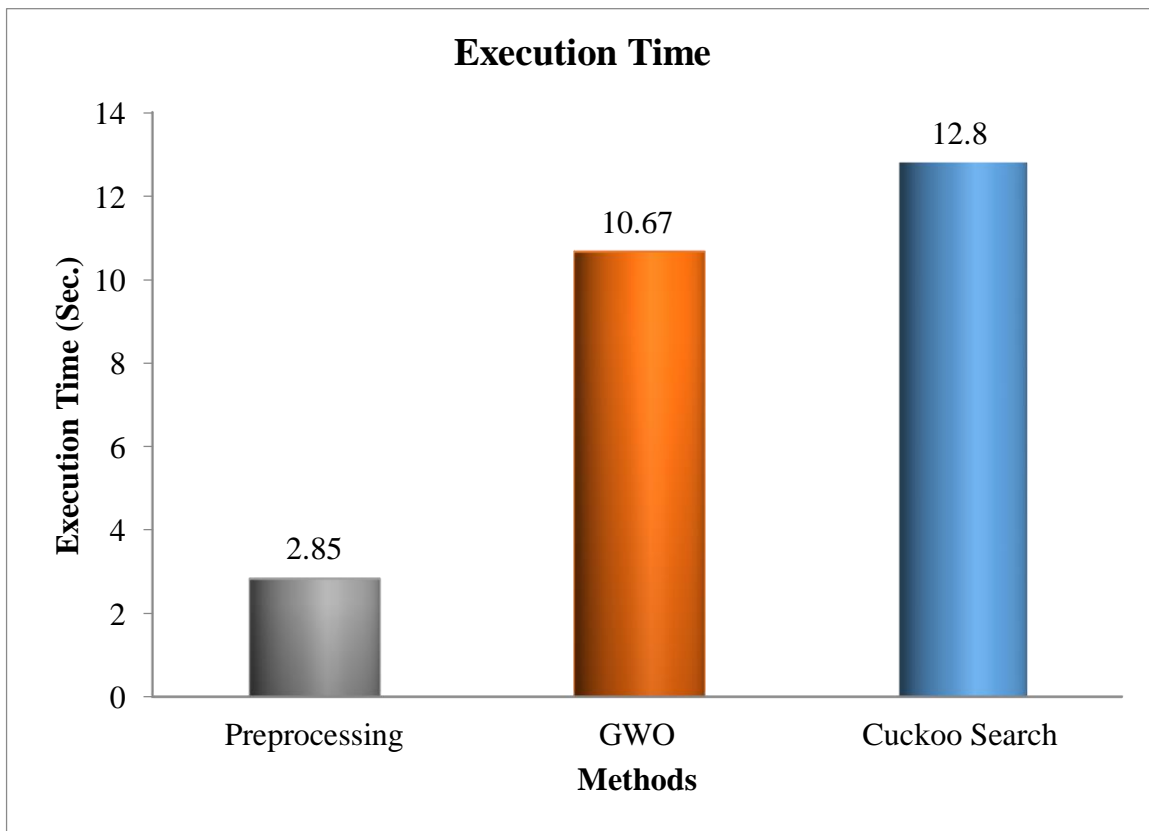


**Figure 6.1: Feature Selection Comparison of GWO, cuckoo search and the proposed method**

Table 6.7 shows the execution time of preprocessing and feature selection methods (GWO and Cuckoo Search)

**Table 6.6: Execution time of Preprocessing and Feature selection**

Method	Execution Time (Sec.)
Preprocessing	2.85
GWO	10.67
Cuckoo Search	12.80



**Figure 6.2: Execution Time Comparison of GWO, cuckoo search and proposed method**

**Table 6.8: Selected Features**

Selected Features
API:GetVolumeNameForVolumeMountPointW
API:InternetOpenA
API:RegCreateKeyExW
API:NtOpenThread
API:VirtualFreeEx
API:NtDuplicateObject
API:NtCreateMutant
API:SendNotifyMessageA

API: FindWindowW
REG:OPENED:HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\NetBT\
REG:OPENED:HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Cryptography\OID\
REG:OPENED:HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Internet Explorer\Main\FeatureControl\
REG:OPENED:HKEY_CLASSES_ROOT\CLSID\
REG:OPENED:HKEY_LOCAL_MACHINE\Software\Microsoft\Rpc\
REG:OPENED:HKEY_CURRENT_USER\Software\Policies\Microsoft\Windows\
REG:OPENED:HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Uninstall\
REG:READ:HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\COM3\
REG:READ:HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Advanced\
REG:READ:HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\WinSock2\Parameters\Protocol_Catalog9\Catalog_Entries\000000000011\
REG:READ:HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\WinSock2\Parameters\NameSpace_Catalog5\Catalog_Entries\000000000003\
REG:READ:HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\WinSock2\Parameters\Protocol_Catalog9\Catalog_Entries\000000000009\
REG:READ:HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\CPC\Volume\{e7136b30-a421-11e5-b597-806d6172696f}\
FILES:OPENED:C:\Documents and Settings\MyUser\Start Menu\Programs\

FILES:READ:?\\PIPE\\
FILES_EXT:OPENED:cab
DIR:CREATED:C:\\Documents and Settings\\MyUser\\Local Settings\\Temp\\is-VTBO7.tmp\\

Table 6.8 shows the selected features using GWO and Cuckoo Search feature selection algorithm. New data is generated using selected features. Table 6.9 shows the sample dataset after preprocessing and feature selection.

**Table 6.9: Sample Data set after Preprocessing and Feature Selection**

Sample Dataset
0,0,0,1,0,1,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,Ransomware
0,1,1,0,1,1,1,0,0,0,1,0,0,0,1,0,0,0,0,0,0,0,1,0,1,Ransomware
1,0,1,0,1,1,1,0,1,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,1,1,Ransomware
0,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,Ransomware
1,0,0,0,1,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0,0,Ransomware
0,0,0,0,1,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,Goodware
1,0,1,0,0,1,1,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,1,Goodware
0,0,1,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,Goodware
1,0,1,0,1,1,1,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,1,0,0,1,Goodware
0,0,1,1,0,1,1,0,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,1,0,1,Goodware

### 6.5.2 DNA Sequence Result

Based on the selected features (GWO and Cuckoo Search), a new dataset is generated. The DNA sequences are generated using this dataset (sample shown in Table 6.9). Table 6.10 shows the sample DNA sequences for sample dataset. The generation of the DNA sequence is explained in section 4.4.1, and it is generated from the dataset features.

**Table 6.10: DNA Sequence for sample data**

Sample Data	DNA Sequence
0,0,0,1,0,1,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0 ,0,Ransomware	TGAAACGACGCGACTTGAAAAAAA CGAAAAAAAAAAAAA
0,1,1,0,1,1,1,0,0,0,1,0,0,0,1,0,0,0,0,0,0,1,0 ,1,Ransomware	TGCTTTTGCTTGCTTTGCTTGAAAA CGAAAAACTGAC
1,0,1,0,1,1,1,0,1,0,0,0,0,1,0,0,0,0,0,0,1,0,0,1 ,1,Ransomware	ACGCGCTGACTGCTGCTTGACGAAA CGCGAAAAAACT
0,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0 ,0,Ransomware	TGAAGTACGCTTTGCGAAAAAAC GAAAAAAAAAAAAA
1,0,0,0,1,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0 ,0,Ransomware	AAAAACGACGCGACGCGAAACGAA AAAAAAAAAACGAA
0,0,0,0,1,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0 ,1,Goodware	TGAAACGACGCGCTGCGAAAAAAA AAAAAAAAAAAAA
1,0,1,0,0,1,1,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0 ,1,Goodware	TTGAACGACGCGCTGCGAAAAAAA AAAAAAAAAAAAA
0,0,1,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0 ,0,Goodware	CGAAACGAAACGACGCGAAAAAAA AAAAAAAAAAAAAC

1,0,1,0,1,1,1,1,0,0,0,0,0,0,1,0,0,0,0,0,1,0,0,1,0,0,1,Goodware	GCGCGCGACGCGCTGCGCGAAAAA AAAAAAAAAAAAA
0,0,1,1,0,1,1,0,0,0,1,0,0,1,0,0,0,0,0,0,0,0,1,0,1,0,1,Goodware	AAAAACGAAAAACGCGAAAAAA AAAAAAAAAAAAA

DNA design constraints are constructed for each DNA sequences. Table 6.11 shows the sample DNA design constraints.

**Table 6.11: Sample DNA Design Constraints**

DNA Sequence	GCC	AT_GC	TM
TGAAACGACGCGACTTGAAAAAACGAAAA AAAAAA	44	1.273	72
TGCTTTTGCTTGCTTTGCTTGAAAACGAAAAAC TGAC	8	11.5	54
ACGCGCTGACTGCTGCTTGACGAAACGCGAAA AAACT	44	1.273	72
TGAACTGACGCTTTGCGAAAAAACGAAAAAA AAAAA	32	2.125	66
AAAAACGACGCGACGCGAAACGAAAAAA AACGAA	28	2.571	64
TGAAACGACGCGCTGCGAAAAAA AAAAAA	32	2.125	66
TTGAACGACGCGCTGCGAAAAAA AAAAAA	24	3.167	62

CGAAACGAAACGACGCGAAAAAAAAAAAAAAAAA AAAAAC	48	1.083	74
GCGCGCGACGCGCTGCGCGAAAAAAAAAAAAAAAAA AAAAAA	48	1.083	74
AAAAACGAAAAACGCGAAAAAAAAAAAAAAAAA AAAAAA	56	0.786	78

Table 6.12 shows the sample k-mer frequency for a DNA sequence.

**Table 6.12: k-mer frequency for sample dataset**

DNA Sequence	AA,AC,AG,AT,CA,CC,CG,CT,GA,GC, GG,GT,TA,TC,TG,TT
TGAAACGACGCGACTTGAAAAAAAA CGAAAAAAAAAAAA	0.32,0.12,0,0,0,0,0.12,0.08,0.12,0.08,0,0,0, 0, 0.08,0.04
TGCTTTTGCTTGCTTTGCTTGAAAAC GAAAAACTGAC	0.72,0.04,0,0,0,0,0.04,0.04,0,0,0,0,0.04, 0.08
ACGCGCTGACTGCTGCTTGACGAAA CGCGAAAAAACT	0.32,0.12,0,0,0,0,0.12,0.08,0.12,0.08,0,0,0, 0, 0.08,0.04
TGAACTGACGCTTTGCGAAAAAACG AAAAAAAAAAAA	0.44,0.12,0,0,0,0,0.12,0.04,0.12,0.04,0,0,0, 0, 0.04,0.04
AAAAACGACGCGACGCGAAACGAA AAAAAAAAAACGAA	0.52,0.12,0,0,0,0,0.08,0.04,0.16,0,0,0,0,0. 04,0
TGAAACGACGCGCTGCGAAAAAAA AAAAAAAAAAAA	0.52,0.12,0,0,0,0,0.08,0.04,0.12,0.04,0,0,0, 0, 0.04,0
TTGAACGACGCGCTGCGAAAAAAA AAAAAAAAAAAA	0.64,0.08,0,0,0,0,0.04,0.04,0.08,0.04,0,0,0, 0, 0.04,0



CGAAACGAAACGACGCGAAAAAA AAAAAAAAAAAAAAC	0.28,0.12,0,0,0,0,0.16,0.04,0.12,0.12,0,0,0, 0, 0.04,0.08
GCGCGCGACGCGCTGCGCGAAAA AAAAAAAAAAAAA	0.32,0.12,0,0,0,0,0.16,0.04,0.12,0.12,0,0,0, 0, 0.04,0.04
AAAAACGAAAAACGCGAAAAAA AAAAAAAAAAAAA	0.12,0.16,0,0,0,0,0.12,0.12,0.16,0.12,0,0,0, 0, 0.12,0.04

Table 6.11 and Table 6.12 are combined to generate a training dataset. Table 6.13 shows the sample training dataset. In that dataset, the class label value Ransomware is set to -1 and Goodware is set to 1.

**Table 6.13: Sample Training Dataset**

Sample Training Data set
0.32,0.12,0,0,0,0,0.12,0.08,0.12,0.08,0,0,0,0.08,0.04,44,1.273,72,-1
0.72,0.04,0,0,0,0,0.04,0.04,0,0,0,0,0.04,0.08,8,11.5,54,-1
0.32,0.12,0,0,0,0,0.12,0.08,0.12,0.08,0,0,0,0.08,0.04,44,1.273,72,-1
0.44,0.12,0,0,0,0,0.12,0.04,0.12,0.04,0,0,0,0.04,0.04,32,2.125,66,-1
0.52,0.12,0,0,0,0,0.08,0.04,0.16,0,0,0,0,0.04,0.28,2.571,64,-1
0.52,0.12,0,0,0,0,0.08,0.04,0.12,0.04,0,0,0,0.04,0.32,2.125,66,1
0.64,0.08,0,0,0,0,0.04,0.04,0.08,0.04,0,0,0,0.04,0.24,3.167,62,1
0.28,0.12,0,0,0,0,0.16,0.04,0.12,0.12,0,0,0,0.04,0.08,48,1.083,74,1
0.32,0.12,0,0,0,0,0.16,0.04,0.12,0.12,0,0,0,0.04,0.04,48,1.083,74,1
0.12,0.16,0,0,0,0,0.12,0.12,0.16,0.12,0,0,0,0.12,0.04,56,0.786,78,1

### 6.5.3 Prediction Result

This section explains the prediction result. The active learning algorithm is used to predict the application, whether it is ransomware or goodware. The training data is trained used active learning algorithm. Table 6.14 shows the trained dataset.

**Table 6.14: Trained Dataset**

Sample Trained Data set
0.32,0.12,0,0,0,0,0.12,0.08,0.12,0.08,0,0,0,0.08,0.04,44,1.273,72,1
0.72,0.04,0,0,0,0,0.04,0.04,0,0,0,0,0.04,0.08,8,11.5,54,-1
0.32,0.12,0,0,0,0,0.12,0.08,0.12,0.08,0,0,0,0.08,0.04,44,1.273,72,1
0.44,0.12,0,0,0,0,0.12,0.04,0.12,0.04,0,0,0,0.04,0.04,32,2.125,66,-1
0.52,0.12,0,0,0,0,0.08,0.04,0.16,0,0,0,0,0.04,0,28,2.571,64,-1
0.52,0.12,0,0,0,0,0.08,0.04,0.12,0.04,0,0,0,0.04,0,32,2.125,66,-1
0.64,0.08,0,0,0,0,0.04,0.04,0.08,0.04,0,0,0,0.04,0,24,3.167,62,1
0.28,0.12,0,0,0,0,0.16,0.04,0.12,0.12,0,0,0,0.04,0.08,48,1.083,74,1
0.32,0.12,0,0,0,0,0.16,0.04,0.12,0.12,0,0,0,0.04,0.04,48,1.083,74,1
0.12,0.16,0,0,0,0,0.12,0.12,0.16,0.12,0,0,0,0.12,0.04,56,0.786,78,1

For active learning algorithm, three main parameters are used: Learning rate, Regularization Parameter and Smoothing Parameter. Table 6.15 shows the value of the parameter.

**Table 6.15: Parameters Details**

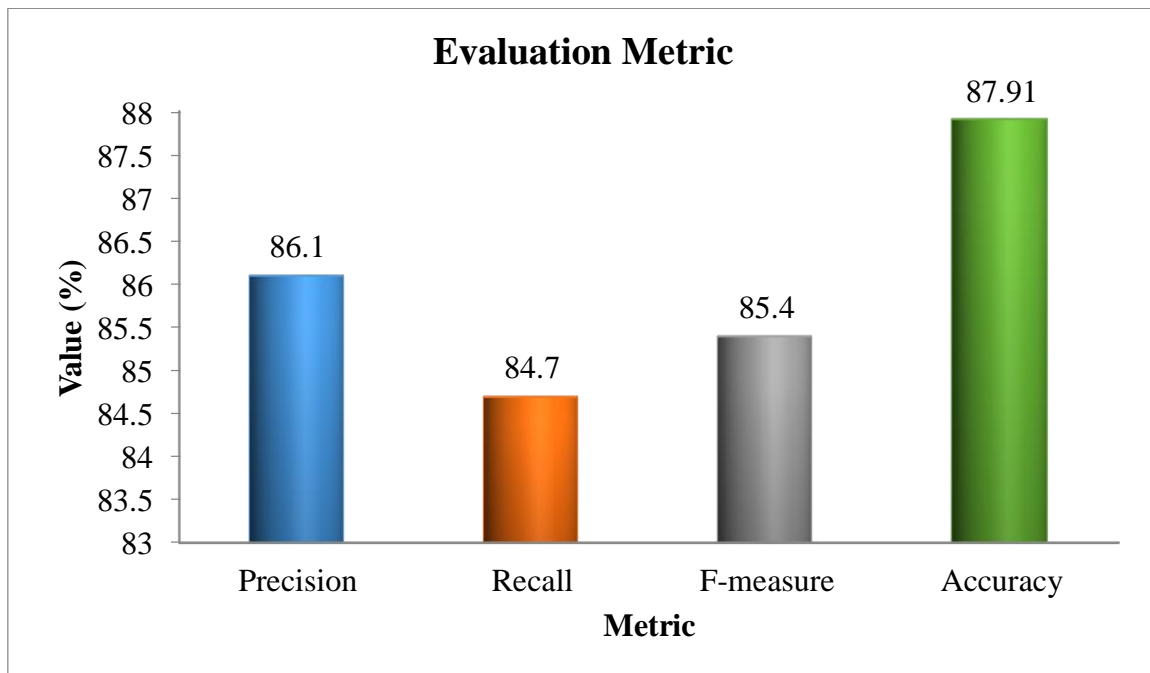
Name	Value
Learning Rate	10

Regularization	0.5
Smoothing Parameter	0.6

The dataset is evaluated using recall, precision, accuracy and f-measure.

**Table 6.16: Evaluation Metrics**

<b>Metric</b>	<b>Value</b>
<b>Precision</b>	86.1
<b>Recall</b>	84.7
<b>F-measure</b>	85.4
<b>Accuracy</b>	87.91



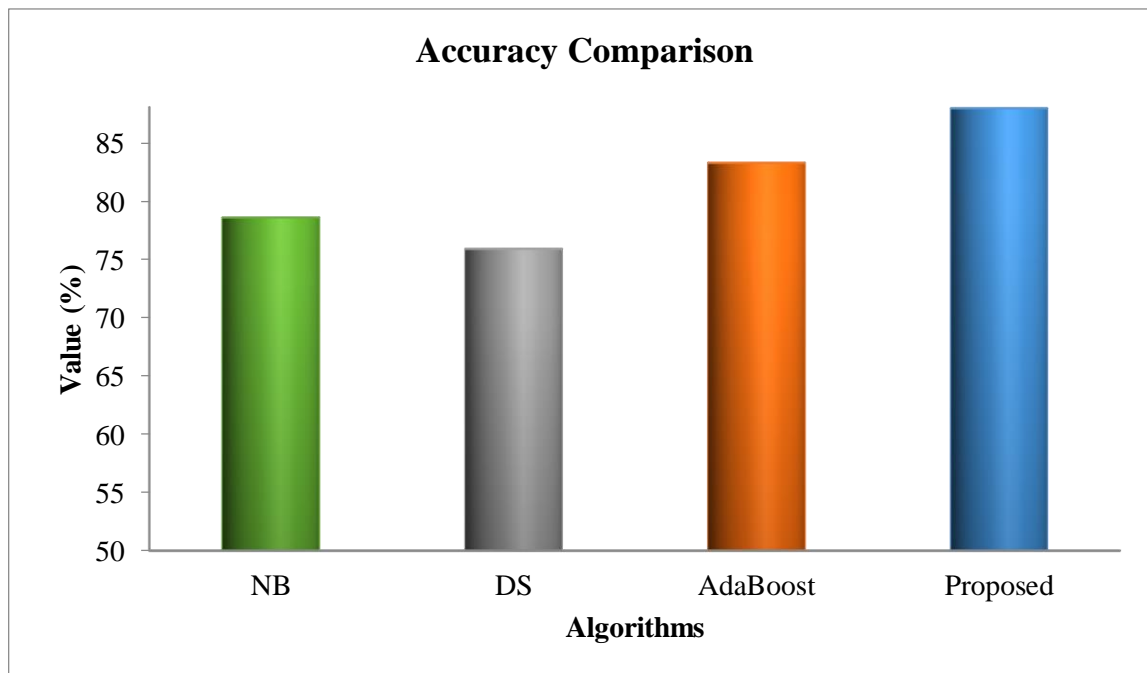
**Figure 6.3: Evaluation Metrics for Proposed Work**

The active learning algorithm is compared with the following classification algorithm Naïve Bayes, Decision Stump, AdaBoost.

Table 6.17 shows the accuracy comparison of algorithms.

**Table 6.17: Accuracy Comparison**

Algorithm	Accuracy
Naïve Bayes	78.52
Decision Stump	75.83
AdaBoost	83.22
Proposed Work	87.91



**Figure 6.4: Classification Accuracy Comparison with respect to Naïve Bayes, Decision Stump, AdaBoost and the proposed method.**

The results are based on the learning parameter, Regularization Parameter and Smoothing Parameter. To improve the performance, automatically readjust these parameters for achieving a good result.

**Table 6.18: Accuracy for different parameter result-1**

<b>Parameters</b>			<b>Accuracy</b>
<b>Learning Rate</b>	<b>Regularisation</b>	<b>Smoothing</b>	
1.0	0.96	0.65	87.7
2.0	0.68	0.14	87.5
3.0	0.25	0.45	89.2
4.0	0.13	0.3	87
5.0	0.03	0.87	86.8
6.0	0.89	0.83	89.7
7.0	0.32	0.72	89.8
8.0	0.77	0.26	90.3
9.0	0.44	0.2	90
10.0	0.64	0.44	89.7

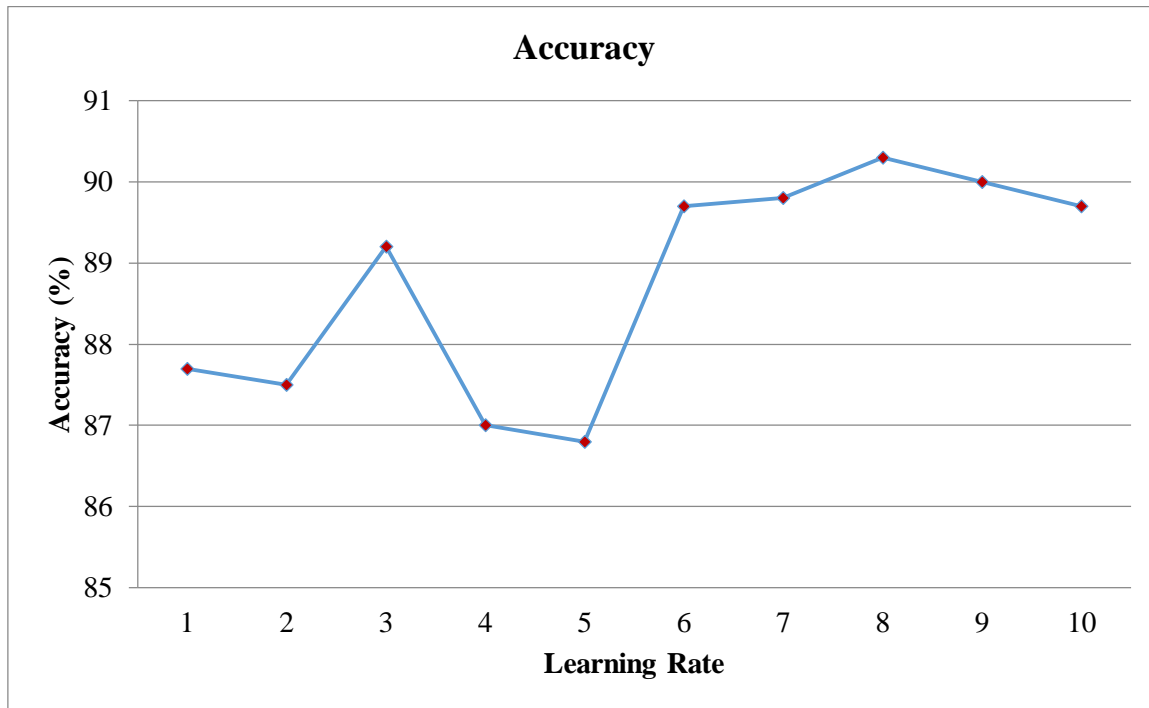


Figure 6.5: Accuracy Result-1

Table 6.19: Accuracy for different parameter result-2

Parameters			Accuracy
Learning Rate	Regularisation	Smoothing	
9.0	0.1	0.14	91.8
2.0	0.2	0.55	92.9
4.0	0.3	0.01	93.6
8.0	0.4	0.66	95
7.0	0.5	0.94	95.1

7.0	0.6	0.89	93.6
8.0	0.7	0.7	95.1
4.0	0.8	0.09	95.1
9.0	0.9	0.19	94.3
5.0	1	0.66	95.2

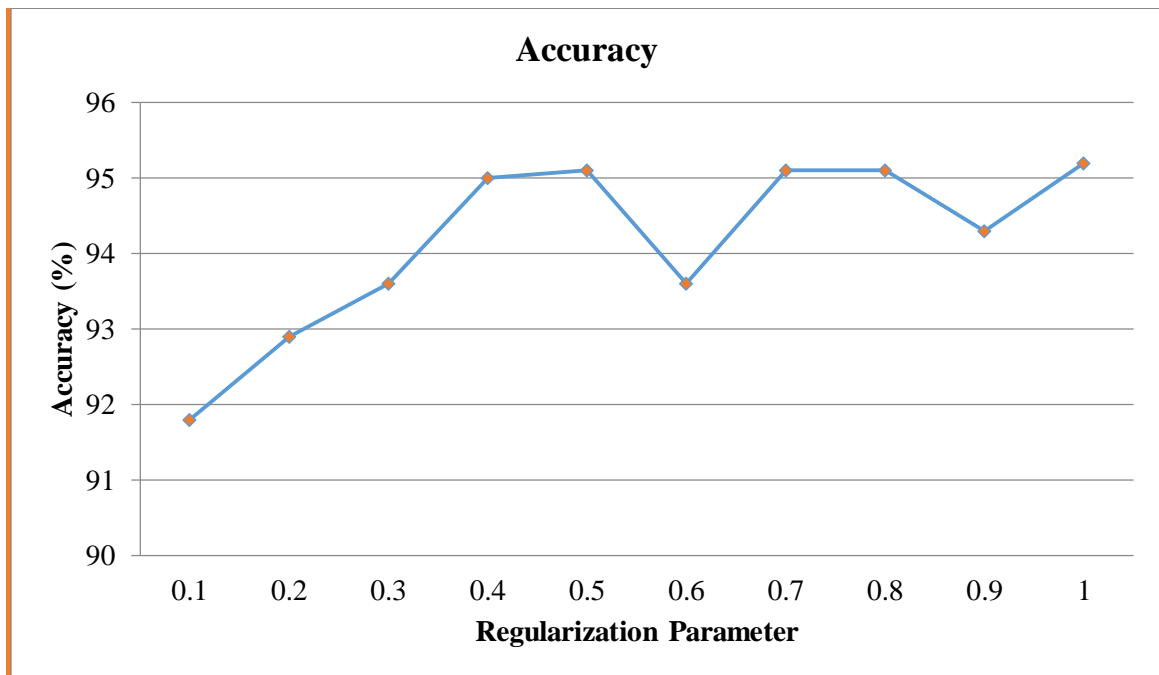
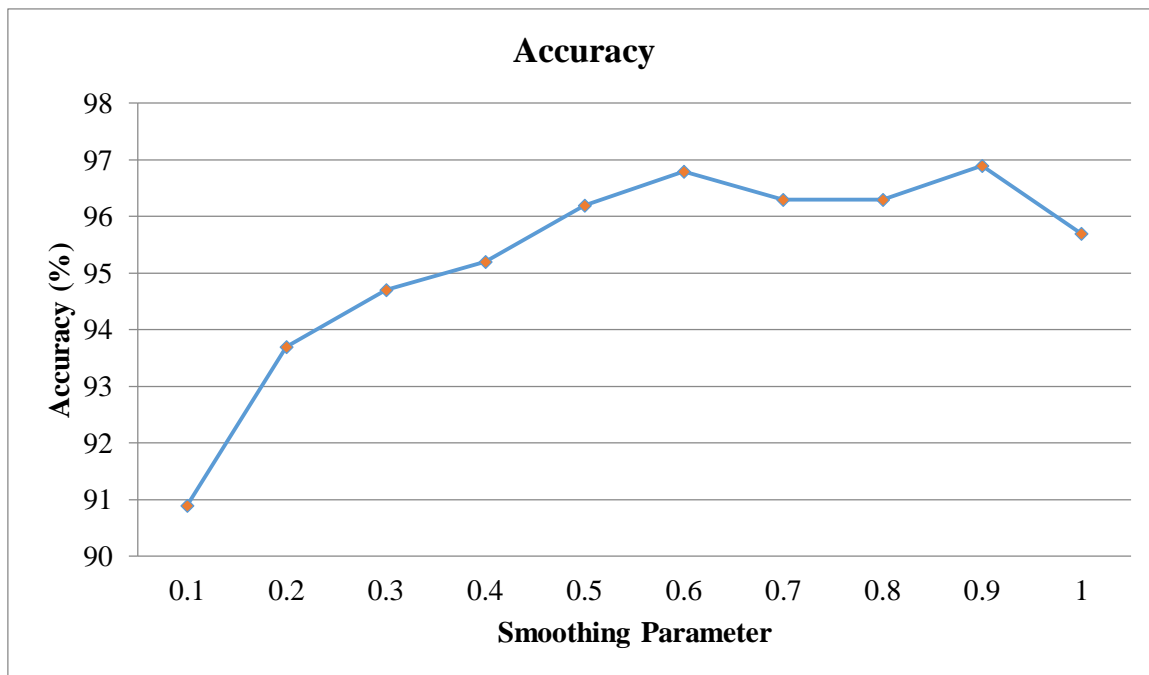


Figure 6.6: Accuracy Result-2

Table 6.20: Accuracy for different parameter result-3

Parameters			Accuracy
Learning Rate	Regularisation	Smoothing	
1.0	0.1	0.1	90.9

2.0	0.2	0.2	93.7
3.0	0.3	0.3	94.7
4.0	0.4	0.4	95.2
5.0	0.5	0.5	96.2
6.0	0.6	0.6	96.8
7.0	0.7	0.7	96.3
8.0	0.8	0.8	96.3
9.0	0.9	0.9	96.9
10.0	1	1	95.7



**Figure 6.7: Accuracy Result-3**

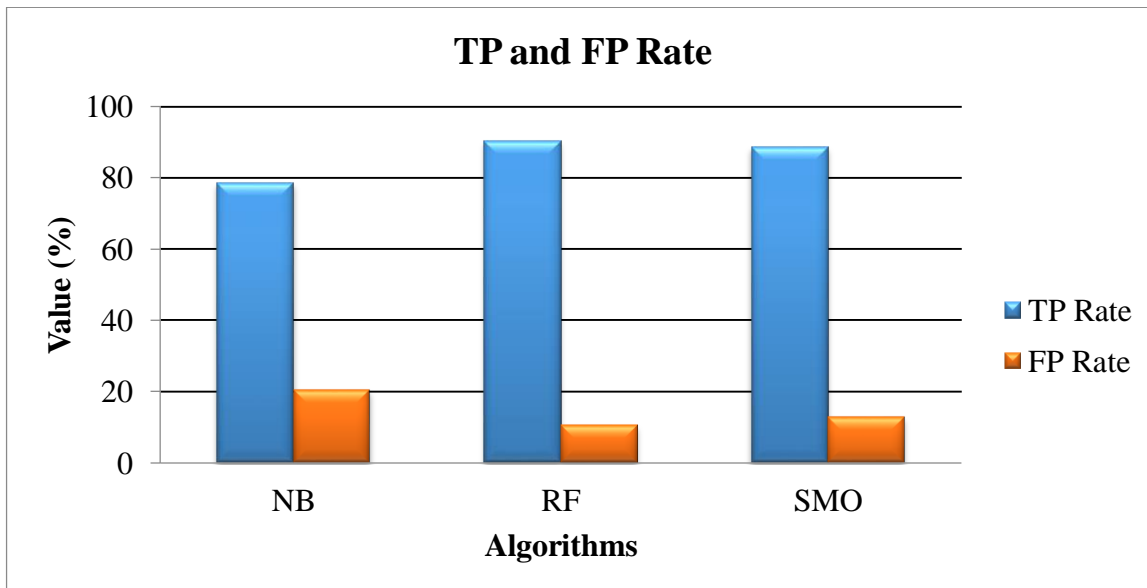


#### 6.5.4 Classification Result

This section gives the result analysis of ransomware family classification. After learning the application is ransomware, it must classify the type of ransomware. The classification algorithms Sequential Minimal Optimization (SMO), Random Forest (RF) and Naïve Bayes (NB), are used to analyse the ransomware family. This section shows the comparison result of True Positive Rate, False Positive Rate, Precision, Recall, F-measure and Accuracy for three classification algorithms (NB, RF, and SMO).

**Table 6.21: TP and FP Rate**

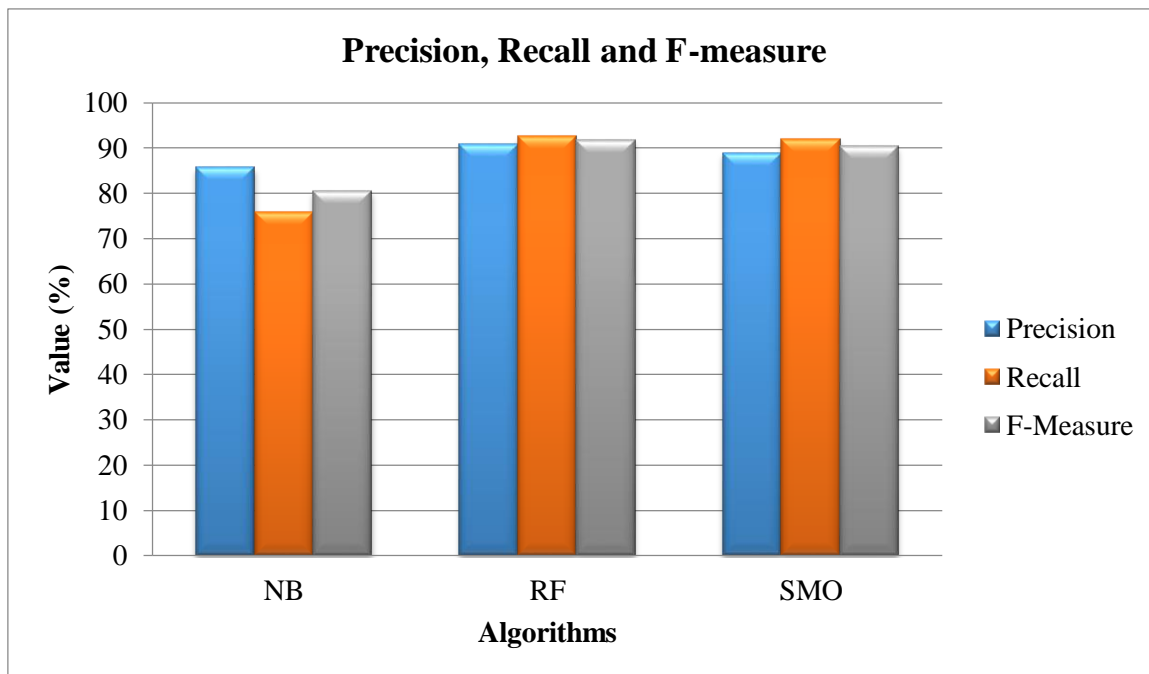
Algorithm	TP Rate	FP Rate
NB	78.5	20.4
RF	90.3	10.6
SMO	88.6	12.8



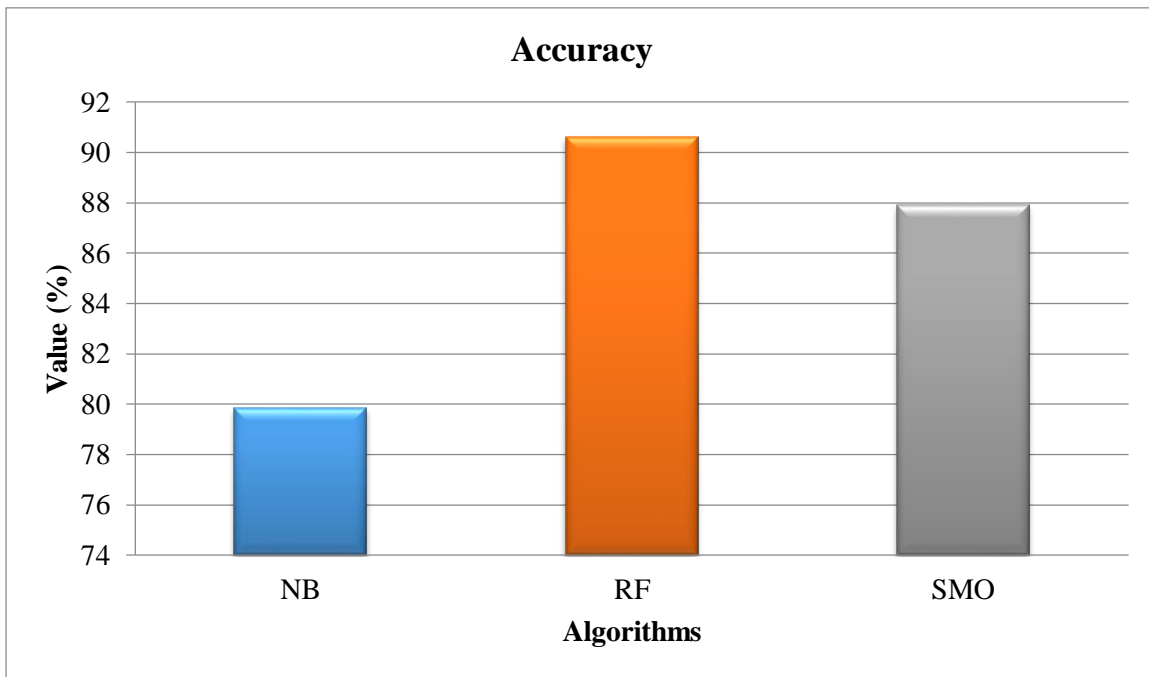
**Figure 6.8: True Positive and False Positive Rate Comparison with respect to Sequential Minimal Optimization (SMO), Random Forest (RF) and Naïve Bayes (NB)**

**Table 6.22: Precision, Recall, and F-measure**

Algorithm	Precision	Recall	F-Measure
NB	85.7	75.9	80.5
RF	91	92.5	91.7
SMO	88.9	92	90.4

**Figure 6.9: Precision, Recall and F-measure Comparison with respect to Sequential Minimal Optimization (SMO), Random Forest (RF) and Naïve Bayes (NB)****Table 6.23: Classification Accuracy**

Algorithm	Accuracy
NB	79.85
RF	90.62
SMO	87.91



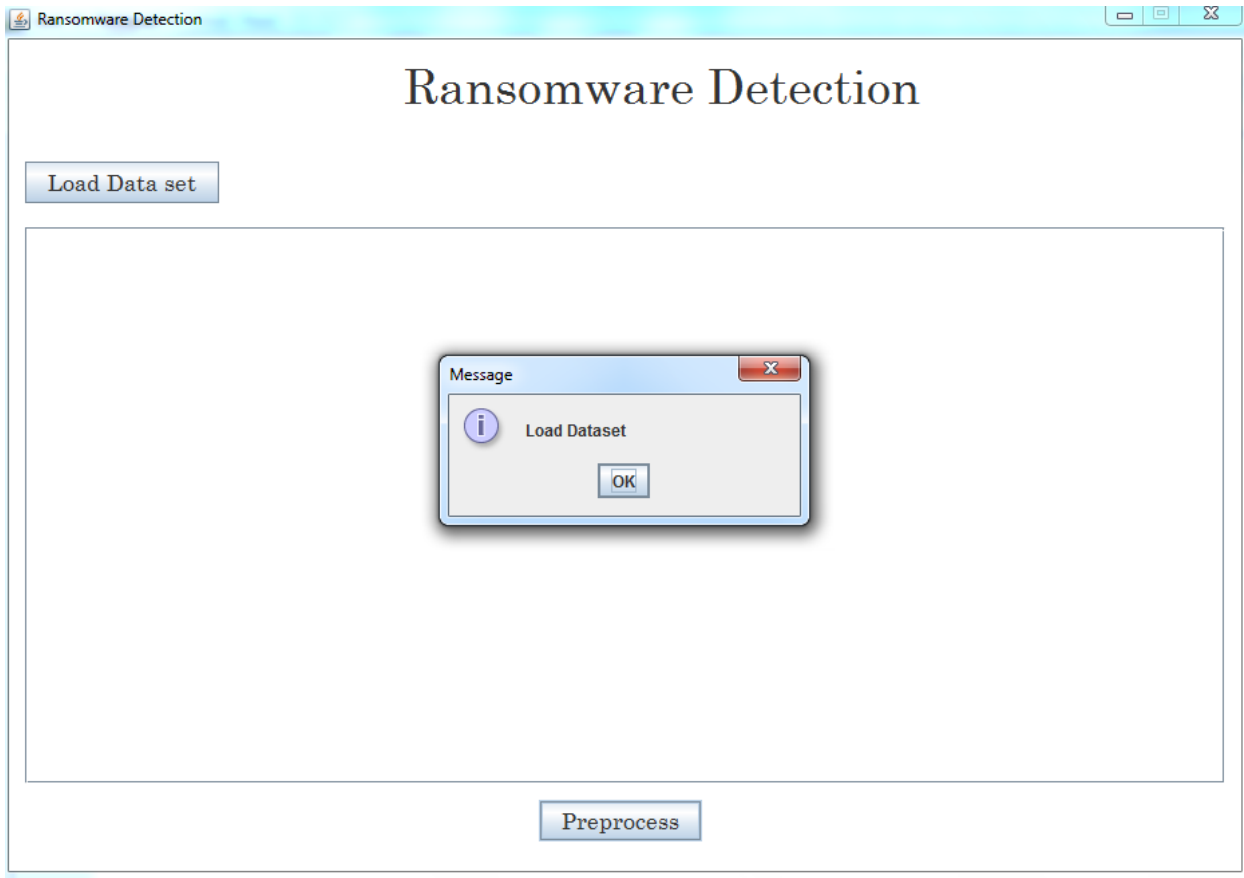
**Figure 6.10: Accuracy Comparison with respect to Sequential Minimal Optimization (SMO), Random Forest (RF) and Naïve Bayes (NB)**

## 6.6 Testing

### *Validation Testing*

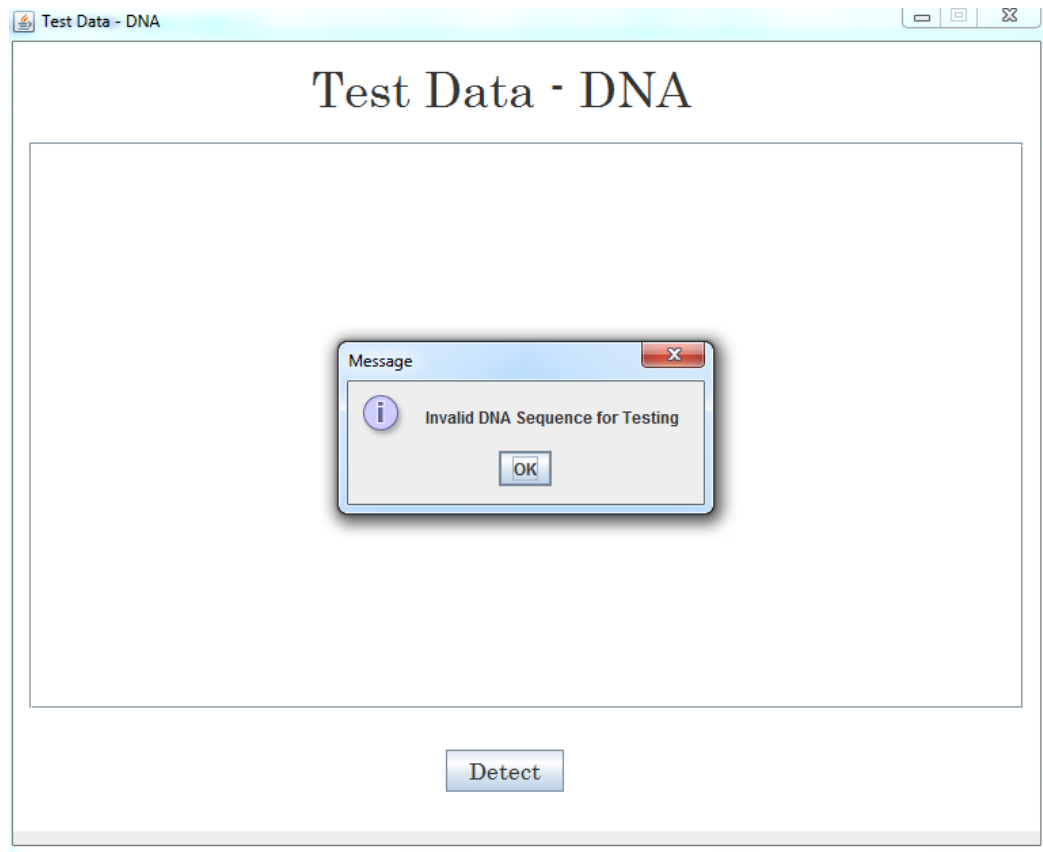
Validation testing is the procedure of ensuring if the developed and tested software satisfies the client or user needs. During validation testing, all business requirements and scenarios need to be tested in detail. Successful validation testing has been conducted of the developed product with all the critical functionalities of an application having been tested.

Figure 6.11 shows the validation testing screenshot. It validates whether the dataset is loaded or not.



**Figure 6.11: Screenshot displaying the Validation Testing Screen -1**

Figure 6.12 shows the validation of Test Data DNA.



**Figure 6.12: Screenshot displaying Validation Testing Screen-2**

## **6.7 Summary**

This chapter explains the performance of the research work. It explored and predicted the behaviour of the application (ransomware or goodware). It describes the dataset information and evaluation metrics. The result evaluation contains three parts, results for feature selection, DNA sequence and prediction.

## **Chapter 7:**

### **Conclusion and Future Enhancement Works**

#### **7.1 Introduction**

In this chapter which is also the last chapter of this study, the findings and contributions of the current thesis are briefly summarised, the limitation of the proposed work is explained, and also the possible future works are mentioned.

#### **7.2 Summary of the Research**

The main contribution of this thesis is to propose a new method for ransomware detection. This work consists of three main phases. Preprocessing with Feature Selection, DNA Sequence Generation and Ransomware Detection.

In the first phase, data preprocessing and feature selection technique is applied to the collected dataset. The preprocessing of data includes remove missing value records and remove columns that all are having the value as zero. The feature selection uses the Grey Wolf Optimisation and cuckoo binary search algorithm for selecting the best features from the dataset.

DNA Sequence generation is the second phase, and it uses design constraints of DNA sequence and k-mer frequency vector. A newly generated dataset after feature selection is used to generate the DNA sequence. The design constraint of DNA is computed, and k-mer frequency vector is generated for the DNA sequence. Based on these computations and vector, a new dataset is generated for ransomware detection training phase.

The final phase is the ransomware detection. In this phase, the new dataset is trained using active learning concept, and the test data is generated and selected by using random DNA sequences method. These data are classified as either ransomware or goodware using a learning algorithm. After classification, the type of ransomware family is analysed.

The proposed methods have been evaluated based on the performance of precision, recall, f-measure and accuracy. Experiments and results show that the proposed work efficiently predict the ransomware.

### **7.3 Testing the thesis hypotheses**

The research was driven by five hypotheses which are described below.

#### **7.3.1 Hypothesis H1 and H2**

Hypothesis H1 stated that *‘it is possible to detect the malicious behaviour of ransomware by conducting Digital DNA sequencing of a piece of software’*. Furthermore, hypothesis H2 stated that *‘Digital DNA mapping of software can be used for classifying and identifying ransomware’*. Both hypotheses were fulfilled by investigations conducted in chapter 4 through the creation of a new ransomware dataset that was derived using the mapping procedure. K-mer frequency vector and DNA sequence design criteria were a critical element in this task. As a whole, this research works fulfil the hypothesis H1 and H2, where it is possible to detect and classify the malicious behaviour of Ransomware by conducting digital DNA sequencing of a piece of software.

#### **7.3.2 Testing hypotheses H3 and H4**

Hypotheses H3 and H4 were tested through multiple analysis of results derived after ransomware detection. Performance assessment of the ransomware detection using active learning algorithm has been conducted in chapter 6. Several evaluation metrics have been used, which has contributed to result from evaluation consisting of three parts; results for feature selection, DNA sequence and prediction. Overall, the hypothesis H3 and H4 is also fulfilled by increasing the accuracy of detection of Ransomware with the use of an active learning algorithm for detecting a Ransomware.

The remainder of this chapter discusses the contributions of the thesis research and possible future work in ransomware detection and future research directions.

## 7.4 Findings

The following findings can be gained using the experiments carried out in chapter 6; Tests were conducted using real-time ransom and goodwill dataset.

- The performance of the active learning-based ransomware detection is better than other classification algorithms like Naïve Bayes, Decision Stump and AdaBoost.
- The accuracy value is varied and improved when readjust the following parameters: learning, regularisation and smoothing parameter.
- Random Forest classification algorithm efficiently predicts the ransomware family compared to Naïve Bayes and SMO classification algorithm.
- By having more training sample, the accuracy of the classifiers improved.

## 7.5 Limitation of proposed work

The proposed work has the following limitations:

- The binary cuckoo search algorithm takes more time to execute and also it select more attributes compared to GWO. Use any other optimisation-based feature selection instead of binary cuckoo search.
- The DNA sequence generation uses only three criteria for DNA sequence design, namely, melting temperature, GC content and AT\_GC ratio. For better results use all the constraints explained in section 4.2.2
- Inactive learning algorithm, the prediction results are based on the learning parameters. It is a crucial problem for this algorithm.

## 7.6 Recommendations for future works

The validation and experiments have shown that the proposed method was successful for ransomware detection compared to other methods. However, the algorithm can still be improved



further. Following are the recommendations proposed for improving ransomware detection performance.

- To increase the size of the dataset
- Use the meta-heuristic algorithm for feature selection
- Use some Novel Classification algorithm for detection.
- Use an expanded dataset including generic cases

## **7.7 Summary**

This research presents a ransomware detection method. The machine learning algorithm is effectively applied for ransomware detection. The real-time dataset is employed to verify the effectiveness and efficiency of the proposed work. A set of evaluation measures were utilised to evaluate the proposed work. The proposed research work was compared with several existing algorithms.

The proposed active learning algorithm is compared to Naïve Bayes, decision stump and AdaBoost classification algorithms. The experiment results show 78.5 % detection accuracy for Naïve Bayes, 75.8% for Decision stump, 83.2% for AdaBoost and 87.9% for the proposed active learning algorithm. The experiment proves that active learning classifiers can efficiently detect ransomware.

By reviewing the summary of the proposed method and its results, can conclude that the objectives of the thesis have been accomplished. For further improvements in ransomware detection problem, some possible future works were mentioned in this chapter.

## References

- Abdel-Basset, M., El-Shahat, D., El-henawy, I., de Albuquerque, V.H.C. & Mirjalili, S. (2020). A new fusion of grey wolf optimizer algorithm with a two-phase mutation for feature selection. *Expert Systems with Applications*, 139, p.112824.
- Adamov, A. & Carlsson, A. (2017). September. The state of ransomware. Trends and mitigation techniques. In *2017 IEEE East-West Design & Test Symposium (EWDTS)* (pp. 1-8). IEEE.
- Aharon, M., Elad, M. & Bruckstein, A. (2006). K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on signal processing*, 54(11), pp.4311-4322.
- Ahmadi, M., Ulyanov, D., Semenov, S., Trofimov, M. & Giacinto, G. (2016). Novel feature extraction, selection and fusion for effective malware family classification. In *Proceedings of the sixth ACM conference on data and application security and privacy* (pp. 183-194). ACM.
- Al-Obaidi, A.T.S. (2013). Improved scatter search using cuckoo search. *International Journal of Advanced Research in Artificial Intelligence*, 2(2), pp.61-67.
- Al-rimy, B.A.S., Maarof, M.A. & Shaid, S.Z.M. (2019). Crypto-ransomware early detection model using novel incremental bagging with enhanced semi-random subspace selection. *Future Generation Computer Systems*.
- Alam, M., Bhattacharya, S., Mukhopadhyay, D. & Chattopadhyay, A. (2018). Rapper: Ransomware prevention via performance counters. *arXiv preprint arXiv:1802.03909*.
- Alasadi, S.A. & Bhaya, W.S. (2017). Review of data preprocessing techniques in data mining. *Journal of Engineering and Applied Sciences*, 12(16), pp.4102-4107.
- Aldeid. (2019). LordPE, Retrieved March 15, 2019, from <https://www.aldeid.com/wiki/LordPE>.
- Alhawi, O.M., Baldwin, J. & Dehghantanha, A. (2018). Leveraging machine learning techniques for windows ransomware network traffic detection. In *Cyber Threat Intelligence* (pp. 93-106). Springer, Cham.

- Alzahrani, A., Alshehri, A., Alshahrani, H., Alharthi, R., Fu, H., Liu, A. & Zhu, Y. (2018). RanDroid: Structural Similarity Approach for Detecting Ransomware Applications in Android Platform. In *2018 IEEE International Conference on Electro/Information Technology (EIT)* (pp. 0892-0897). IEEE.
- Ambusaidi, M.A., He, X., Nanda, P. & Tan, Z. (2016). Building an intrusion detection system using a filter-based feature selection algorithm. *IEEE transactions on computers*, 65(10), pp.2986-2998.
- Andronio, N., Zanero, S. & Maggi, F. (2015). Heldroid: Dissecting and detecting mobile ransomware. In *International Symposium on Recent Advances in Intrusion Detection* (pp. 382-404). Springer, Cham.
- Anjana, T.K. (2017). Discussion On Ransomware, Wannacry Ransomware and Cloud Storage Services Against Ransom Malware Attacks. *International Journal for Research Trends and Innovation*, 2, pp.310-314.
- Arita, M., Nishikawa, A., Hagiya, M., Komiya, K., Gouzu, H. & Sakamoto, K. (2000). Improving sequence design for DNA computing. In *Proceedings of the 2nd Annual Conference on Genetic and Evolutionary Computation* (pp. 875-882). Morgan Kaufmann Publishers, Inc.
- Arita, M. & Kobayashi, S. (2002). DNA sequence design using templates. *New Generation Computing*, 20(3), pp.263-277.
- Arora, S. & Anand, P. (2019). Binary butterfly optimization approaches for feature selection. *Expert Systems with Applications*, 116, pp.147-160.
- Asahiro, Y. (2005). Simple greedy methods for DNA word design. In *Proc. 9th World Multi-Conference on Systemics, Cybernetics and Informatics*, 2005 (pp. 186-191).
- Baldwin, J. & Dehghantanha, A. (2018). Leveraging support vector machine for opcode density based detection of crypto-ransomware. In *Cyber Threat Intelligence* (pp. 107-136). Springer, Cham.
- Banescu, S., Wuchner, T., Salem, A., Guggenmos, M., Ochoa, M. & Pretschner, A. (2015). A framework for empirical evaluation of malware detection resilience against behavior

- obfuscation. In *2015 10th International Conference on Malicious and Unwanted Software (MALWARE)* (pp. 40-47). IEEE.
- Baskar, S.S., Arockiam, L. & Charles, S. (2013). A systematic approach on data pre-processing in data mining. *Compusoft*, 2(11), p.335.
- Becker's Healthcare. (2016). First known ransomware attack in 1989 also targeted healthcare. Retrieved July 1, 2019, from <https://www.beckershospitalreview.com/healthcare-information-technology/first-known-ransomware-attack-in-1989-also-targeted-healthcare.html>.
- Bennasar, M., Hicks, Y. & Setchi, R. (2015). Feature selection using joint mutual information maximisation. *Expert Systems with Applications*, 42(22), pp.8520-8532.
- Bi Zone. (2017). Bad Rabbit ransomware. Retrieved June 2, 2019, from <https://www.bi.zone/news/bad-rabbit-ransomware/>.
- Bisht S.S. & Panda A.K. (2014). DNA Sequencing: Methods and Applications. In: *Ravi I., Baunthiyal M., Saxena J. (eds) Advances in Biotechnology*. Springer, New Delhi
- BleepingComputer (2014). CryptoWall and Help\_Decrypt Ransomware Information Guide and FAQ, Retrieved June 2, 2019, from <https://www.bleepingcomputer.com/virus-removal/cryptowall-ransomware-information>.
- Breiman, L. (2001). Random forests. *Machine learning*, 45(1), pp.5-32.
- Borders, K. & Prakash, A. (2004). Web tap: detecting covert web traffic. In *Proceedings of the 11th ACM conference on Computer and communications security* (pp. 110-120). ACM.
- Calyptix. (2015). Critroni Ransomware Decryption: Not an Option, Retrieved June 2, 2019, from <https://www.calyptix.com/malware/critroni-ransomware-decryption-not-an-option/>.
- Cabaj, K., Gawkowski, P., Grochowski, K. & Osojca, D. (2015). Network activity analysis of CryptoWall ransomware. *Przegląd Elektrotechniczny*, 91(11), pp.201-204.
- Cabaj, K., Gregorczyk, M. & Mazurczyk, W. (2018). Software-defined networking-based crypto ransomware detection using HTTP traffic characteristics. *Computers & Electrical Engineering*, 66, pp.353-368.

- Capture BAT. (2019). Capture BAT. Retrieved March 15, 2019, from <https://www.honeynet.org/project/CaptureBAT>.
- Cesa-Bianchi, N., Conconi, A. & Gentile, C. (2005). A second-order perceptron algorithm. *SIAM Journal on Computing*, 34(3), pp.640-668.
- Chadha, S. & Kumar, U. (2017). Ransomware: Let's fight back!. In *2017 International Conference on Computing, Communication and Automation (ICCCA)* (pp. 925-930). IEEE.
- Chen, G. & Chen, J. (2015). A novel wrapper method for feature selection and its applications. *Neurocomputing*, 159, pp.219-226.
- Chen, J.C. & Li, B. (2015). Evolution of exploit kits-exploring past trends and current improvements. Retrieved January 12, 2019, from <https://www.trendmicro.de/cloud-content/us/pdfs/security-intelligence/white-papers/wp-evolution-of-exploit-kits.pdf>.
- Chen, K., Zhou, F.Y. & Yuan, X.F. (2019). Hybrid particle swarm optimization with spiral-shaped mechanism for feature selection. *Expert Systems with Applications*, 128, pp.140-156.
- Chen, Z.G., Kang, H.S., Yin, S.N. & Kim, S.R. (2017). Automatic ransomware detection and analysis based on dynamic API calls flow graph. In *Proceedings of the International Conference on Research in Adaptive and Convergent Systems* (pp. 196-201). ACM.
- Chormunge, S. & Jena, S. (2018). Correlation based feature selection with clustering for high dimensional data. *Journal of Electrical Systems and Information Technology*, 5(3), pp.542-549.
- Christensen, J.B. & Beuschau, N. (2017). Ransomware detection and mitigation tool.
- Chaudhary, A., Kolhe, S. & Kamal, R. (2016). An improved random forest classifier for multi-class classification. *Information Processing in Agriculture*, 3(4), pp.215-222.
- Choudhary, S.P. & Vidyarthi, M.D. (2015). A simple method for detection of metamorphic malware using dynamic analysis and text mining. *Procedia Computer Science*, 54, pp.265-270.

- Cios, K.J., Pedrycz, W. and Swiniarski, R.W. (1998). Data mining and knowledge discovery. In *Data mining methods for knowledge discovery* (pp. 1-26). Springer, Boston, MA.
- Combs, G. (2019). Wireshark, Retrieved March 15, 2019, from <https://www.wireshark.org/>
- Comodo. (2019). How to detect ransomware. Retrieved June 2, 2019, from <https://enterprise.comodo.com/how-to-detect-ransomware.php>.
- Cooper, D. R., and Schindler, P. S. (2014). *Business Research Methods*. © The McGraw– Hill Companies.
- Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S. & Singer, Y. (2006). Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7(Mar), pp.551-585.
- Crammer, K., Kulesza, A. & Dredze, M. (2009). Adaptive regularization of weight vectors. In *Advances in neural information processing systems* (pp. 414-422).
- Cui, G. & Li, X. (2010). The optimization of DNA encodings based on modified PSO/GA algorithm. In *2010 International Conference On Computer Design and Applications* (Vol. 1, pp. V1-609). IEEE.
- De Groot, J. (2019). A History of Ransomware Attacks: The Biggest and Worst Ransomware Attacks of All Time. Retrieved July 1, 2019, from <https://digitalguardian.com/blog/history-ransomware-attacks-biggest-and-worst-ransomware-attacks-all-time>.
- Delaney, D. (2017). How to detect the presence of WannaCry Ransomware and SMBv1 servers on your network, Retrieved June 2, 2019, from <https://www.netfort.com/blog/detect-wannacry-ransomware/>.
- Deloitte. (2016). Ransomware Holding Your Data Hostage, Retrieved March 30, 2019, from <https://www2.deloitte.com/content/dam/Deloitte/us/Documents/risk/us-aers-ransomware.pdf>
- Denning, P. J. (1988). The science of computing: Computer viruses. *American Scientist*, Vol. 76, Iss. 3, pp. 236–238.

- Dharmarajan, R. & Vijayasanthi R. (2015). An Overview on Data Preprocessing Methods in Data Mining. *Int J. Sci. Research Dev. (IJSRD)*, Vol.3, Iss.3, pp. 3544-3546
- Díaz-Uriarte, R. and De Andres, S.A. (2006). Gene selection and classification of microarray data using random forest. *BMC bioinformatics*, 7(1), p.3.
- Dredze, M., Crammer, K. & Pereira, F. (2008). Confidence-weighted linear classification. In *Proceedings of the 25th international conference on Machine learning* (pp. 264-271). ACM.
- Dunn, J. E., Macaulay, T. & Magee, T. (2018). The worst types of ransomware attacks. Retrieved June 2, 2019, from <https://www.computerworlduk.com/galleries/security/worst-ransomware-attacks-3641916/>
- Egele, M., Scholte, T., Kirda, E. & Kruegel, C. (2012). A survey on automated dynamic malware-analysis techniques and tools. *ACM computing surveys (CSUR)*, 44(2), p.6.
- Emary, E., Yamany, W., Hassanien, A.E. & Snasel, V. (2015). Multi-objective gray-wolf optimization for attribute reduction. *Procedia Computer Science*, 65, pp.623-632.
- Emary, E., Zawbaa, H.M., Grosan, C. & Hassenian, A.E. (2015). Feature subset selection approach by gray-wolf optimization. In *Afro-European Conference for Industrial Advancement* (pp. 1-13). Springer, Cham.
- Emary, E., Zawbaa, H.M. & Hassanien, A.E. (2016). Binary grey wolf optimization approaches for feature selection. *Neurocomputing*, 172, pp.371-381.
- Faris, H., Aljarah, I., Al-Betar, M.A. and Mirjalili, S. (2018). Grey wolf optimizer: a review of recent variants and applications. *Neural computing and applications*, 30(2), pp.413-435.
- Fujii, K. & Kashima, H. (2016). Budgeted stream-based active learning via adaptive submodular maximization. In *Advances in Neural Information Processing Systems* (pp. 514-522).

- Gaidhane, P.J. & Nigam, M.J. (2018). A hybrid grey wolf optimizer and artificial bee colony algorithm for enhancing the performance of complex systems. *Journal of computational science*, 27, pp.284-302.
- Galal, H.S., Mahdy, Y.B. & Atiea, M.A. (2016). Behavior-based features model for malware detection. *Journal of Computer Virology and Hacking Techniques*, 12(2), pp.59-67.
- Gallo, T. & Liska, A. (2017) Introduction to Ransomware, Retrieved November 1, 2018, from <https://www.oreilly.com/library/view/ransomware/9781491967874/ch01.html>.
- Gandotra, E., Bansal, D. & Sofat, S. (2014). Malware analysis and classification: A survey. *Journal of Information Security*, 5(02), p.56.
- Gangwar, K., Mohanty, S. & Mohapatra, A.K. (2017). Analysis and Detection of Ransomware Through Its Delivery Methods. In *International Conference on Recent Developments in Science, Engineering and Technology* (pp. 353-362). Springer, Singapore.
- García, S., Ramírez-Gallego, S., Luengo, J., Benítez, J.M. & Herrera, F. (2016). Big data preprocessing: methods and prospects. *Big Data Analytics*, 1(1), p.9.
- Garzon, M.H., 2008. *DNA Computing: 13th International Meeting on DNA Computing, DNA13, Memphis, TN, USA, June 4-8, 2007, Revised Selected Papers* (Vol. 4848). Springer Science & Business Media.
- Gherboudj, A., Layeb, A. and Chikhi, S. (2012). Solving 0-1 knapsack problems by a discrete binary version of cuckoo search algorithm. *International Journal of Bio-Inspired Computation*, 4(4), pp.229-236.
- Gómez-Hernández, J.A., Álvarez-González, L. & García-Teodoro, P. (2018). R-Locker: Thwarting ransomware action through a honeyfile-based approach. *Computers & Security*, 73, pp.389-398.
- Goncharov, M. (2011). Traffic direction systems as malware distribution tools. Retrieved December 12, 2018, from [http://index-of.co.uk/Various/rpt\\_malware-distribution-tools.pdf](http://index-of.co.uk/Various/rpt_malware-distribution-tools.pdf).



- Gonzalez, D. & Hayajneh, T. (2017). Detection and prevention of crypto-ransomware. In *2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON)* (pp. 472-478). IEEE.
- Guangzhao, C., Yunyun, N., Yanfeng, W., Xuncai, Z. & Linqiang, P. (2007). A new approach based on PSO algorithm to find good computational encoding sequences. *Progress in Natural Science*, 17(6), pp.712-716.
- Guo, Y. & Schuurmans, D. (2008). Discriminative batch mode active learning. In *Advances in neural information processing systems* (pp. 593-600).
- Hampton, N., Baig, Z. & Zeadally, S. (2018). Ransomware behavioural analysis on windows platforms. *Journal of information security and applications*, 40, pp.44-51.
- Han, J., Pei, J. and Kamber, M. (2011). *Data mining: concepts and techniques*. Elsevier.
- Hancer, E., Xue, B. & Zhang, M. (2018). Differential evolution for filter feature selection based on information theory and feature ranking. *Knowledge-Based Systems*, 140, pp.103-119.
- Hanneke, S. & Yang, L. (2015). Minimax analysis of active learning. *The Journal of Machine Learning Research*, 16(1), pp.3487-3602.
- Hao, S., Lu, J., Zhao, P., Zhang, C., Hoi, S.C. & Miao, C. (2017). Second-order online active learning and its applications. *IEEE Transactions on Knowledge and Data Engineering*, 30(7), pp.1338-1351.
- Hasherezade. (2016). Look Into Locky Ransomware, Retrieved June 2, 2019, from <https://blog.malwarebytes.com/threat-analysis/2016/03/look-into-locky/>.
- Hayakawa, K., Gerding, E.H., Stein, S. & Shiga, T. (2015). Online mechanisms for charging electric vehicles in settings with varying marginal electricity costs. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.
- HB Gary, (2018). Digital DNA Retrieved January 21, 2019, from [https://moonsoft.net/materials/hbgary\\_digital\\_dna.pdf](https://moonsoft.net/materials/hbgary_digital_dna.pdf).
- Hex-rays. (2019). IDA, Retrieved March 15, 2019, from <https://www.hex-rays.com/products/ida>.

- Hoi, S.C., Wang, J. & Zhao, P. (2014). Libol: A library for online learning algorithms. *The Journal of Machine Learning Research*, 15(1), pp.495-499.
- Homayoun, S., Dehghantanha, A., Ahmadzadeh, M., Hashemi, S., Khayami, R., Choo, K.K.R. & Newton, D.E. (2019). DRTHIS: Deep ransomware threat hunting and intelligence system at the fog layer. *Future Generation Computer Systems*, 90, pp.94-104.
- Hosfelt, D.D. (2015). Automated detection and classification of cryptographic algorithms in binary programs through machine learning. *arXiv preprint arXiv:1503.01186*.
- Ibrahim, Z., Khalid, N.K., Lim, K.S., Buyamin, S. & Mukred, J.A.A. (2011). A binary vector evaluated particle swarm optimization based method for DNA sequence design problem. In *2011 IEEE Student Conference on Research and Development* (pp. 160-164). IEEE.
- Jackson, J. (2016). User Behavior Analytics (UBA) & Ransomware Analytics. Retrieved June 2, 2019, from <https://itknowledgeexchange.techtarget.com/virtual-ciso/user-behavior-analytics-uba-ransomware-analytics/>.
- Jadhav, S., He, H. & Jenkins, K. (2018). Information gain directed genetic algorithm wrapper feature selection for credit rating. *Applied Soft Computing*, 69, pp.541-553.
- Jahedpari, F. (2015). Artificial prediction markets for online prediction. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.
- Jain, D. & Singh, V. (2018). An efficient hybrid feature selection model for dimensionality reduction. *Procedia Computer Science*, 132, pp.333-341.
- Jiang, Q., Zhao, X. & Huang, K. (2011). A feature selection method for malware detection. In *2011 IEEE International Conference on Information and Automation* (pp. 890-895). IEEE.
- Kamath, R.S., Dongale, T.D., Pawar, P. & Kamat, R.K. (2016). Random Forest Modeling for Mice Down Syndrome Through Protein Expression: A Supervised Learning Approach. *Research Journal of Pharmaceutical, Biological and Chemical Sciences*. 7(4), pp.830-836.

- Kardile, A.B. (2017). Crypto Ransomware Analysis and Detection Using Process Monitor (*Doctoral dissertation*).
- Kaspersky Labs (2019). Machine Learning for Malware Detection. Retrieved June 10, 2019, from <https://media.kaspersky.com/en/enterprise-security/Kaspersky-Lab-Whitepaper-Machine-Learning.pdf>.
- Kaur, R. & Singh, M. (2014). A survey on zero-day polymorphic worm detection techniques. *IEEE Communications Surveys & Tutorials*, 16(3), pp.1520-1549.
- Kawashimo, S., Ono, H., Sadakane, K. & Yamashita, M. (2007). Dynamic neighborhood searches for thermodynamically designing DNA sequence. In *International Workshop on DNA-Based Computers* (pp. 130-139). Springer, Berlin, Heidelberg.
- Khalid, N.K., Kurniawan, T.B., Ibrahim, Z., Yusof, Z.M., Khalid, M. & Engelbrecht, A.P. (2008). A model to optimize DNA sequences based on particle swarm optimization. In *2008 Second Asia International Conference on Modelling & Simulation (AMS)* (pp. 534-539). IEEE.
- Khalid, N.K., Zainal, M.S., Khalid, M. & Engelbrecht, A.P. (2008). DNA Sequence Optimization Based on Continuous Particle Swarm Optimization for Reliable DNA Computing and DNA Nanotechnology 1.
- Khammassi, C. & Krichen, S. (2017). A GA-LR wrapper approach for feature selection in network intrusion detection. *computers & security*, 70, pp.255-277.
- Kharaz, A., Arshad, S., Mulliner, C., Robertson, W. & Kirda, E. (2016). {UNVEIL}: A Large-Scale, Automated Approach to Detecting Ransomware. In *25th {USENIX} Security Symposium ({USENIX} Security 16)* (pp. 757-772).
- Klijnsma Y. (2015). Analysis of a piece of ransomware in development: the story of 'cryptoapp'. Retrieved February 20, 2019, from <http://blog.0x3a.com/post/126900680679/analysis-of-a-piece-of-ransomware-in-development>.

- Kohli, M. & Arora, S. (2018). Chaotic grey wolf optimization algorithm for constrained optimization problems. *Journal of computational design and engineering*, 5(4), pp.458-472.
- Kohavi, R. and Provost, F. (1998). Confusion matrix. *Machine learning*, 30(2-3), pp.271-274.
- Kolodenker, E., Koch, W., Stringhini, G. & Egele, M. (2017). April. PayBreak: defense against cryptographic ransomware. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security* (pp. 599-611). ACM.
- Kolter, J.Z. & Maloof, M.A. (2006). Learning to detect and classify malicious executables in the wild. *Journal of Machine Learning Research*, 7(Dec), pp.2721-2744.
- Kornblum, J. (2006). Identifying almost identical files using context triggered piecewise hashing. *Digital investigation*, 3, pp.91-97.
- Kumar, A., Kuppusamy, K.S. & Aghila, G. (2017). A learning model to detect maliciousness of portable executable using integrated feature set. *Journal of King Saud University-Computer and Information Sciences*.
- Kumar, B.R., 2012. DNA representation. In *DNA Sequencing-Methods and Applications*. IntechOpen.
- Kumar, P. (2019). DNA Sequencing: 7 Methods Used for DNA Sequencing. Retrieved June 2, 2019, from <http://www.biologydiscussion.com/dna/dna-sequencing/dna-sequencing-7-methods-used-for-dna-sequencing/11840>.
- Kumar, V. and Rathee, N. (2011). Knowledge discovery from database using an integration of clustering and classification. *International Journal of Advanced Computer Science and Applications*, 2(3), pp.29-33.
- Kurniawan, T.B., Khalid, N.K., Ibrahim, Z., Abidin, M.S.Z. & Khalid, M. (2009). Sequence design for direct-proportional length-based DNA computing using population-based ant colony optimization. In *2009 ICCAS-SICE* (pp. 1486-1491). IEEE.
- Kurniawan, T.B., Khalid, N.K., Ibrahim, Z., Khalid, M. & Middendorf, M. (2008). Evaluation of ordering methods for DNA sequence design based on ant colony system. In *2008*

- Second Asia International Conference on Modelling & Simulation (AMS)* (pp. 905-910). IEEE.
- Labani, M., Moradi, P., Ahmadizar, F. & Jalili, M. (2018). A novel multivariate filter method for feature selection in text classification problems. *Engineering Applications of Artificial Intelligence*, 70, pp.25-37.
- Learn Cryptography. (2019). What Are Hash Functions. Retrieved June 2, 2019, from <https://learncryptography.com/hash-functions/what-are-hash-functions>.
- Lee, I.H., Park, J.Y., Jang, H.M., Chai, Y.G. & Zhang, B.T. (2002). DNA implementation of theorem proving with resolution refutation in propositional logic. In *International Workshop on DNA-Based Computers* (pp. 156-167). Springer, Berlin, Heidelberg.
- Li, Y. & Long, P.M. (2000). The relaxed online maximum margin algorithm. In *Advances in neural information processing systems* (pp. 498-504).
- Liska, A. & Gallo, T. (2016). Ransomware: Defending against digital extortion. *O'Reilly Media, Inc.*
- Liu, K., Wang, B., Lv, H., Wei, X. & Zhang, Q. (2019). A BPSON Algorithm Applied to DNA Codes Design. *IEEE Access*, 7, pp.88811-88821.
- Lu, J., Zhao, P. and Hoi, S.C. (2016). Online passive-aggressive active learning. *Machine Learning*, 103(2), pp.141-183.
- Lu, T., Zhang, L., Wang, S. & Gong, Q. (2017). Ransomware detection based on v-detector negative selection algorithm. In *2017 International Conference on Security, Pattern Analysis, and Cybernetics (SPAC)* (pp. 531-536). IEEE.
- Mafarja, M. & Mirjalili, S. (2018). Whale optimization approaches for wrapper feature selection. *Applied Soft Computing*, 62, pp.441-453.
- Manbari, Z., AkhlaghianTab, F. & Salavati, C. (2019). Hybrid fast unsupervised feature selection for high-dimensional data. *Expert Systems with Applications*, 124, pp.97-118.
- Maniath, S., Ashok, A., Poornachandran, P., Sujadevi, V.G., Sankar, A.P. & Jan, S. (2017). Deep learning LSTM based ransomware detection. In *2017 Recent Developments in Control, Automation & Power Engineering (RDCAPE)* (pp. 442-446). IEEE.

- Maxam, A.M. & Gilbert, W. (1977). A new method for sequencing DNA. *Proceedings of the National Academy of Sciences*, 74(2), pp.560-564.
- Mercaldo, F., Nardone, V., Santone, A. & Visaggio, C.A. (2016). Ransomware steals your phone. formal methods rescue it. In *International Conference on Formal Techniques for Distributed Objects, Components, and Systems* (pp. 212-221). Springer, Cham.
- Miao, J. & Niu, L. (2016). A survey on feature selection. *Procedia Computer Science*, 91, pp.919-926.
- Miao, Q., Liu, J., Cao, Y. & Song, J. (2016). Malware detection using bilayer behavior abstraction and improved one-class support vector machines. *International Journal of Information Security*, 15(4), pp.361-379.
- Microsoft. (2019). Process Explorer, Retrieved March 15, 2019, from <https://docs.microsoft.com/en-us/sysinternals/downloads/process-explorer>.
- Microsoft. (2019). Process Monitor, Retrieved March 15, 2019, from <https://docs.microsoft.com/en-us/sysinternals/downloads/procmon>.
- Mimoso, M. (2017). Leaked NSA Exploit Spreading Ransomware World Wide, Retrieved January 12, 2019, from <https://threatpost.com/leaked-nsa-exploit-spreading-ransomware-worldwide/125654/>.
- Mirjalili, S., Mirjalili, S.M. & Lewis, A. (2014). Grey wolf optimizer. *Advances in engineering software*, 69, pp.46-61.
- Mirjalili, S., Saremi, S., Mirjalili, S.M. and Coelho, L.D.S. (2016). Multi-objective grey wolf optimizer: a novel algorithm for multi-criterion optimization. *Expert Systems with Applications*, 47, pp.106-119.
- Mlakar, U., Fister Jr, I. & Fister, I. (2016). Hybrid self-adaptive cuckoo search for global optimization. *Swarm and Evolutionary Computation*, 29, pp.47-72.
- Mohri, M. & Rostamizadeh, A. (2013). Perceptron mistake bounds. *arXiv preprint arXiv:1305.0208*.

- Monrose, F., Dacier, M., Blanc, G. & García-Alfaro, J. (2016). Research in Attacks, Intrusions, and Defenses. In *Proc. of the 19th International Symposium (RAID), Paris, France* (Vol. 9854).
- Moore, C. (2016). August. Detecting ransomware with honeypot techniques. In *2016 Cybersecurity and Cyberforensics Conference (CCC)* (pp. 77-81). IEEE.
- Murphy, R. (2018). Guide to Detecting and Preventing Ransomware, Retrieved June 2, 2019, from <https://www.blackstratus.com/guide-detecting-preventing-ransomware/>.
- Nauman, M., Azam, N. & Yao, J. (2016). A three-way decision making approach to malware analysis using probabilistic rough sets. *Information Sciences*, 374, pp.193-209.
- Nieuwenhuizen, D., 2017. A behavioural-based approach to ransomware detection. *Whitepaper. MWR Labs Whitepaper*.
- Nikam, R. (2011). Introduction to Malware & Malware Analysis, Retrieved March 23, 2019, from <https://www.chmag.in/articles/momsguide/introduction-to-malware-malware-analysis/>.
- OllyDbg. (2019). OllyDbg. Retrieved March 15, 2019, from <http://www.ollydbg.de/>
- OllyDump. (2019). OllyDump. Retrieved March 15, 2019, from <http://www.woodmann.com/collaborative/tools/index.php/OllyDump>.
- Osanaiye, O., Cai, H., Choo, K.K.R., Dehghantanha, A., Xu, Z. & Dlodlo, M. (2016). Ensemble-based multi-filter feature selection method for DDoS detection in cloud computing. *EURASIP Journal on Wireless Communications and Networking*, 2016(1), p.130.
- Palmer, D. (2017). Bad Rabbit: Ten things you need to know about the latest ransomware outbreak, Retrieved June 2, 2019, from <https://www.zdnet.com/article/bad-rabbit-ten-things-you-need-to-know-about-the-latest-ransomware-outbreak/>.
- Pantic, M., 2005. Introduction to machine learning & case-based reasoning. *London: Imperial College*.

- Passi, H. (2018). Introduction to Malware: Definition, Attacks, Types and Analysis, Retrieved March 23, 2019, from <https://www.greycampus.com/blog/information-security/introduction-to-malware-definition-attacks-types-and-analysis>.
- Paul, C.B. (2017). Entropy based file type identification and partitioning. *Naval Postgraduate School* Monterey United States.
- Payne, R.B. & Sorensen, M.D. (2005). The cuckoos (Vol. 15). *Oxford University Press*.
- Pedersen, J., Bastola, D., Dick, K., Gandhi, R. & Mahoney, W. (2012). Blast your way through malware analysis assisted by bioinformatics tools. In *Proceedings of the International Conference on Security and Management (SAM) (p. 1)*. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp).
- Pedersen, J., Bastola, D., Dick, K., Gandhi, R. & Mahoney, W. (2013). Fingerprinting Malware using Bioinformatics Tools Building a Classifier for the Zeus Virus (Computer Security track, Virus Detection). In *Proceedings of the International Conference on Security and Management (SAM) (p. 1)*. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp).
- Pillai, A., Kadikar, R., Vasanthi, M.S. & Amutha, B. (2018). Analysis of AES-CBC Encryption for Interpreting Crypto-Wall Ransomware. In *2018 International Conference on Communication and Signal Processing (ICCSP)* (pp. 0599-0604). IEEE.
- Platt, J. (1998). Sequential minimal optimization: A fast algorithm for training support vector machines.
- Qiu, Q., Mukre, P., Bishop, M., Burns, D. & Wu, Q. (2007). Hardware acceleration for thermodynamic constrained DNA code generation. In *International Workshop on DNA-Based Computers* (pp. 201-210). Springer, Berlin, Heidelberg.
- Quick Heal. (2014). Introduction to Malware & Malware Analysis, Retrieved March 30, 2019, from



- [http://dlupdate.quickheal.com/documents/technical\\_papers/introduction\\_to\\_malware\\_and\\_malware\\_analysis.pdf](http://dlupdate.quickheal.com/documents/technical_papers/introduction_to_malware_and_malware_analysis.pdf)
- Raman, R. (2018). Seven ways to detect ransomware beyond antivirus, Retrieved June 2, 2019, from <https://economictimes.indiatimes.com/small-biz/security-tech/security/seven-ways-to-detect-ransomware-beyond-antivirus/articleshow/64596762.cms?from=mdr>.
- Rodrigues, D., Pereira, L.A., Almeida, T.N.S., Papa, J.P., Souza, A.N., Ramos, C.C. and Yang, X.S. (2013). BCS: A binary cuckoo search algorithm for feature selection. In *2013 IEEE International Symposium on Circuits and Systems (ISCAS2013)* (pp. 465-468). IEEE.
- Ruvolo, P. & Eaton, E. (2014). Online multi-task learning via sparse dictionary optimization. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*.
- Salvi, M.H.U. & Kerkar, M.R.V. (2016). Ransomware: A cyber extortion. *Asian Journal For Convergence In Technology (AJCT)*, 2.
- Sanger, F. & Coulson, A.R. (1975). A rapid method for determining sequences in DNA by primed synthesis with DNA polymerase. *Journal of molecular biology*, 94(3), pp.441-448.
- Sanger, F., Nicklen, S. & Coulson, A.R. (1977). DNA sequencing with chain-terminating inhibitors. *Proceedings of the national academy of sciences*, 74(12), pp.5463-5467.
- SantaLucia, J. (1998). A unified view of polymer, dumbbell, and oligonucleotide DNA nearest-neighbor thermodynamics. *Proceedings of the National Academy of Sciences*, 95(4), pp.1460-1465.
- Saputra, M.F.A., Widiyaningtyas, T. & Wibawa, A.P. (2018). Illiteracy Classification Using K Means-Naïve Bayes Algorithm. *JOIV: International Journal on Informatics Visualization*, 2(3), pp.153-158.
- Scaife, N., Carter, H., Traynor, P. & Butler, K.R. (2016). Cryptolock (and drop it): stopping ransomware attacks on user data. In *2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS)* (pp. 303-312). IEEE.

- Sculley, D. (2007). August. Online active learning methods for fast label-efficient spam filtering. In *CEAS* (Vol. 7, p. 143).
- Senthilkumar, M., Ramasamy, V., Sheen, S., Veeramani, C., Bonato, A. and Batten, L. (2016). Computational Intelligence, Cyber Security and Computational Models.
- Settles, B. (2009). Active learning literature survey. *University of Wisconsin-Madison Department of Computer Sciences*.
- Sgandurra, D., Muñoz-González, L., Mohsen, R. & Lupu, E.C. (2016). Automated dynamic analysis of ransomware: Benefits, limitations and use for detection. *arXiv preprint arXiv:1609.03020*.
- Sharp, R. (2017). An Introduction to Malware. Retrieved March 23, 2019, from <https://orbit.dtu.dk/files/139067614/malware.pdf>
- Shaukat, S.K. & Ribeiro, V.J. (2018). January. RansomWall: A layered defense system against cryptographic ransomware attacks using machine learning. In *2018 10th International Conference on Communication Systems & Networks (COMSNETS)* (pp. 356-363). IEEE.
- Sheng, V.S., Provost, F. & Ipeirotis, P.G. (2008). Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 614-622). ACM.
- Shijo, P.V. & Salim, A. (2015). Integrated static and dynamic analysis for malware detection. *Procedia Computer Science*, 46, pp.804-811.
- Shin, S.Y., Lee, I.H., Kim, D. & Zhang, B.T. (2005). Multiobjective evolutionary optimization of DNA sequences for reliable DNA computing. *IEEE transactions on evolutionary computation*, 9(2), pp.143-158.
- Singh, N. & Singh, S.B. (2017). A novel hybrid GWO-SCA approach for optimization problems. *Engineering Science and Technology, an International Journal*, 20(6), pp.1586-1601.

- Singh, S. (2018). Ransomware Command and Control Detection using Machine Learning, Retrieved January 12, 2019, from <https://www.acalvio.com/ransomware-command-and-control-detection-using-machine-learning>.
- Sinitzyn, F. (2015). Teslacrypt 2.0 disguised as cryptowall, Retrieved December 12, 2018, from <https://securelist.com/blog/research/71371/teslacrypt-2-0-disguised-as-cryptowall/>.
- Solorio-Fernández, S., Carrasco-Ochoa, J.A. & Martínez-Trinidad, J.F. (2016). A new hybrid filter–wrapper feature selection method for clustering based on ranking. *Neurocomputing*, 214, pp.866-880.
- Song, S., Kim, B. & Lee, S. (2016). The effective ransomware prevention technique using process monitoring on android platform. *Mobile Information Systems*, 2016.
- SourceForge. (2019). Process Hacker, Retrieved March 15, 2019, from <https://processhacker.sourceforge.io/>.
- SourceForge. (2019). Regshot, Retrieved March 15, 2019, from <https://sourceforge.net/projects/regshot/>.
- Stefanis, C., Alexopoulos, A., Voidarou, C., Vavias, S. and Bezirtzoglou, E. (2013). Principal methods for isolation and identification of soil microbial communities. *Folia microbiologica*, 58(1).
- Sultan, H., Khalique, A., Alam, S.I, and Tanweer, S. (2018). A Survey on Ransomware: Evolution, Growth and Impact. *International Journal of Advanced Research in Computer Science*, 9(2).
- Symantec. (2014). CryptoDefense, the CryptoLocker Imitator, Makes Over \$34,000 in One Month. Retrieved June 2, 2019, from <https://www.symantec.com/connect/blogs/cryptodefense-cryptolocker-imitator-makes-over-34000-one-month>.
- Symantec. (2016). Internet Security Threat Report, Retrieved March 30, 2019, from <https://www.symantec.com/content/dam/symantec/docs/reports/istr-21-2016-en.pdf>.

- Symantec Security Response. (2017). Petya ransomware outbreak: Here's what you need to know. Retrieved June 2, 2019, from <https://www.symantec.com/blogs/threat-intelligence/petya-ransomware-wiper>.
- Takeuchi, Y., Sakai, K. & Fukumoto, S. (2018). August. Detecting ransomware using support vector machines. In *Proceedings of the 47th International Conference on Parallel Processing Companion* (p. 1). ACM.
- Tanaka, F., Nakatsugawa, M., Yamamoto, M., Shiba, T. & Ohuchi, A. (2001). Developing support system for sequence design in DNA computing. In *International Workshop on DNA-Based Computers* (pp. 129-137). Springer, Berlin, Heidelberg.
- Tandon, A. and Nayyar, A. (2019). A Comprehensive Survey on Ransomware Attack: A Growing Havoc Cyberthreat. In *Data Management, Analytics and Innovation* (pp. 403-420). Springer, Singapore.
- Tawhid, M.A. & Ali, A.F. (2017). A Hybrid grey wolf optimizer and genetic algorithm for minimizing potential energy function. *Memetic Computing*, 9(4), pp.347-359.
- Tawhid, M.A. & Dsouza, K.B. (2018). Hybrid binary bat enhanced particle swarm optimization algorithm for solving feature selection problems. *Applied Computing and Informatics*.
- Tong, S. (2001). *Active learning: theory and applications* (Vol. 1). USA: Stanford University.
- Tong, S. & Koller, D. (2001). Support vector machine active learning with applications to text classification. *Journal of machine learning research*, 2(Nov), pp.45-66.
- TrendMicro. (2019). Ransomware. Retrieved June 2, 2019, from <https://www.trendmicro.com/vinfo/in/security/definition/ransomware>.
- Tseng, A., Chen, Y., Kao, Y. & Lin, T. (2016). Deep learning for ransomware detection. *IEICE Tech. Rep.*, 116(282), pp.87-92.
- Tu, Q., Chen, X. & Liu, X. (2019). Multi-strategy ensemble grey wolf optimizer and its application to feature selection. *Applied Soft Computing*, 76, pp.16-30.

- Tulpan, D.C., Hoos, H.H. & Condon, A.E. (2002). Stochastic local search algorithms for DNA word design. In *International Workshop on DNA-Based Computers* (pp. 229-241). Springer, Berlin, Heidelberg.
- Ucci, D., Aniello, L. & Baldoni, R. (2018). Survey of machine learning techniques for malware analysis. *Computers & Security*.
- Wan, C. (2019), Feature Selection Paradigms. In *Hierarchical Feature Selection for Knowledge Discovery. Advanced Information and Knowledge Processing, Springer, Cham*. pp. 17-23
- Wan, J., Wu, P., Hoi, S.C., Zhao, P., Gao, X., Wang, D., Zhang, Y. & Li, J. (2015). Online learning to rank for content-based image retrieval. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.
- Wang, J., Zhao, P. & Hoi, S.C. (2012). Exact soft confidence-weighted learning. *arXiv preprint arXiv:1206.4612*.
- Wang, B. & Pineau, J. (2015). Online boosting algorithms for anytime transfer and multitask learning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.
- Wang, P. & Wang, Y.S. (2015). Malware behavioural detection and vaccine development by using a support vector model classifier. *Journal of Computer and System Sciences*, 81(6), pp.1012-1026.
- Wang, Y., Shen, Y., Zhang, X., Cui, G. & Sun, J. (2018). An Improved Non-dominated Sorting Genetic Algorithm-II (INSGA-II) applied to the design of DNA codewords. *Mathematics and Computers in Simulation*, 151, pp.131-139.
- Watson, J.D. and Crick, F.H. (1953), January. The structure of DNA. In *Cold Spring Harbor symposia on quantitative biology* (Vol. 18, pp. 123-131). Cold Spring Harbor Laboratory Press.
- Watson, J.D. & Crick, F.H.C. (1993). Molecular structure of nucleic acids: a structure for deoxyribose nucleic acid. *Jama*, 269(15), pp.1966-1967.

- Weckstén, M., Frick, J., Sjöström, A. & Järpe, E. (2016). A novel method for recovery from Crypto Ransomware infections. In *2016 2nd IEEE International Conference on Computer and Communications (ICCC)* (pp. 1354-1358). IEEE.
- Wei, W., Xuedong, Z., Qiang, Z. & Jin, X. (2007). The optimization of DNA encodings based on GA/SA algorithms. *Progress in Natural Science*, 17(6), pp.739-744.
- Wetmur, J.G. (1991). DNA probes: applications of the principles of nucleic acid hybridization. *Critical reviews in biochemistry and molecular biology*, 26(3-4), pp.227-259.
- Wikibooks. (2017). Structural Biochemistry/Nucleic Acid/DNA/DNA Denaturation, Retrieved June 2, 2019, from [https://en.wikibooks.org/wiki/Structural\\_Biochemistry/Nucleic\\_Acid/DNA/DNA\\_Denaturation](https://en.wikibooks.org/wiki/Structural_Biochemistry/Nucleic_Acid/DNA/DNA_Denaturation)
- Xiao, J.H., Jiang, Y., He, J.J. & Cheng, Z. (2013). A dynamic membrane evolutionary algorithm for solving DNA sequences design with minimum free energy. *MATCH Communications in Mathematical and in Computer Chemistry*, 70(3), pp.971-986.
- Xiao, J., Xu, J., Chen, Z., Zhang, K. & Pan, L. (2009). A hybrid quantum chaotic swarm evolutionary algorithm for DNA encoding. *Computers & Mathematics with Applications*, 57(11-12), pp.1949-1958.
- Xiao, J., Zhang, X. & Xu, J. (2012). A membrane evolutionary algorithm for DNA sequence design in DNA computing. *Chinese Science Bulletin*, 57(6), pp.698-706.
- Xu, C., Zhang, Q., Wang, B. & Zhang, R. (2008). Research on the DNA sequence design based on GA/PSO algorithms. In *2008 2nd International Conference on Bioinformatics and Biomedical Engineering* (pp. 816-819). IEEE.
- Yamany, W., El-Bendary, N., Hassanien, A.E. & Emary, E. (2016). Multi-objective cuckoo search optimization for dimensionality reduction. *Procedia Computer Science*, 96, pp.207-215.
- Yamany, W., Emary, E. & Hassanien, A.E. (2016). New rough set attribute reduction algorithm based on grey wolf optimization. In *The 1st International Conference on*

- Advanced Intelligent System and Informatics (AIS2015)*, November 28-30, 2015, Beni Suef, Egypt (pp. 241-251). Springer, Cham.
- Yang, X.S. & Deb, S. (2009). Cuckoo search via Lévy flights. In *2009 World Congress on Nature & Biologically Inspired Computing (NaBIC)* (pp. 210-214). IEEE.
- Zahra, A. & Shah, M.A. (2017). IoT based ransomware growth rate evaluation and detection using command and control blacklisting. In *2017 23rd International Conference on Automation and Computing (ICAC)* (pp. 1-6). IEEE.
- Zavarsky, P. & Lindskog, D. (2016). Experimental analysis of ransomware on windows and android platforms: Evolution and characterization. *Procedia Computer Science*, 94, pp.465-472.
- Zhang, H., Xiao, X., Mercaldo, F., Ni, S., Martinelli, F. & Sangaiah, A.K. (2019). Classification of ransomware families with machine learning based on N-gram of opcodes. *Future Generation Computer Systems*, 90, pp.211-221.
- Zhang, J., Xiong, Y. & Min, S. (2019). A new hybrid filter/wrapper algorithm for feature selection in classification. *Analytica chimica acta*, 1080, pp.43-54.
- Zhang, K., Xu, J., Geng, X., Xiao, J. & Pan, L. (2008). Improved taboo search algorithm for designing DNA sequences. *Progress in Natural Science*, 18(5), pp.623-627.
- Zhang, L., Mistry, K., Lim, C.P. & Neoh, S.C. (2018). Feature selection using firefly optimization for classification and regression models. *Decision Support Systems*, 106, pp.64-85.
- Zhang, Q. & Wang, B. (2011). On the bounds of DNA coding with H-distance. *MATCH Commun. Math. Comput. Chem*, 66, pp.371-380.
- Zhang, Q., Wang, B. & Wei, X. (2011). Evaluating the different combinatorial constraints in DNA computing based on minimum free energy. *MATCH Commun. Math. Comput. Chem*, 65, pp.291-308.
- Zhang, X., Wang, Y., Cui, G., Niu, Y. & Xu, J. (2009). Application of a novel IWO to the design of encoding sequences for DNA computing. *Computers & Mathematics with Applications*, 57(11-12), pp.2001-2008.

- Zhang, Y., Cheng, S., Shi, Y., Gong, D.W. & Zhao, X. (2019). Cost-sensitive feature selection using two-archive multi-objective artificial bee colony algorithm. *Expert Systems with Applications*, 137, pp.46-58.
- Zhao, F., Zhao, J., Niu, X., Luo, S. & Xin, Y. (2018). A Filter Feature Selection Algorithm Based on Mutual Information for Intrusion Detection. *Applied Sciences*, 8(9), p.1535.
- Zhao, P., Hoi, S. and Zhuang, J. (2013). Active learning with expert advice. *arXiv preprint arXiv:1309.6875*.
- Zorz Z. (2015) Chimera crypto-ransomware is hitting german companies. Retrieved February 20, 2019, from <https://www.helpnetsecurity.com/2015/11/03/chimera-crypto-ransomware-is-hitting-german-companies/>.
- Zuhair, H. & Selamat, A. (2017). Phishing classification models: issues and perspectives. In *2017 IEEE Conference on Open Systems (ICOS)* (pp. 26-31). IEEE.



## **Appendix A**

### **Product- Ransomware Detector Software**

A standalone Ransomware detector software has been developed using Java. In this software, the Folder Shield feature protects files present in local folders. The software developed was free, standalone, and it protects the folders and their subfolders. The software was built in such a way that no unauthorised program can delete or modify files in the protected zone, though file creation is permitted.

In addition to behaviour-based malware detection, the software protects effectively against ransomware. The software is light on system resources. It provides a fast scan, and the size of the application is tiny in size. As the size is small, it gives a light touch on the system's resources. The application provides an excellent desktop interface where the notifications are shown as popups in the taskbar.

The software will be running in the background, so it detects and quarantines ransomware strictly based on behaviour. When the software is installed on a user's PC or laptop, it creates a folder on a local disk (C:/) named Ransomware. If any ransomware attack is detected, the encrypted file is moved and stored in the folder.

## Appendix B

### Snippets of Code used

#### Feature Selection

```

public String find()
{
    String sel="";
    try
    {
        int n=dt.gmodData[0].length;
        double alpha_pos[]=new double[n];
        double alpha_scr=Double.POSITIVE_INFINITY;

        double beta_pos[]=new double[n];
        double beta_scr=Double.POSITIVE_INFINITY;

        double delta_pos[]=new double[n];
        double delta_scr=Double.POSITIVE_INFINITY;

        double position[][]=new double[dt.gmodData.length][n];

        Random rn=new Random();
        for(int i=0;i<dt.gmodData.length;i++)
        {
            for(int j=0;j<n;j++)
            {
                //double e1=rn.nextDouble()*(dt.upper[j]-dt.lower[j])+dt.lower[j];
                double e1=dt.gmodData[i][j]*(dt.upper[j]-dt.lower[j])+dt.lower[j];
                position[i][j]=e1;
            }
        }

        int iter=0;
        dt.bestScr=new double[dt.Max_iteration];

        while(iter<dt.Max_iteration)
        {
            for(int i=0;i<position.length;i++)
            {
                int flagub[]=new int[position[0].length];
                int flaglb[]=new int[position[0].length];

                for(int j=0;j<flagub.length;j++)
                {
                    if(position[i][j]>dt.upper[j])

```

```

        flagub[j]=1;
    else
        flagub[j]=0;

    if(position[i][j]<dt.lower[j])
        flaglb[j]=1;
    else
        flaglb[j]=0;
    }

    for(int j=0;j<n;j++)
    {
        int c1=flagub[j];
        int c2=flaglb[j];
        if(c1+c2==0)
            c1=1;
        else
            c1=0;

        position[i][j]=(position[i][j]*c1)+dt.upper[j]*flagub[j]+dt.lower[j]*flaglb[j];

    }
    double fitness=0;
    for(int j=0;j<n;j++)
    {
        fitness=fitness+(-20*Math.exp(-
        .2*Math.sqrt(position[i][j]*position[i][j]/n))-
        Math.exp((Math.cos(2*Math.PI*position[i][j]))/n)+20+Math.exp(1));
    }

    if(fitness<alpha_scr)
    {
        alpha_scr=fitness;
        alpha_pos=position[i];
    }
    if(fitness>alpha_scr && fitness<beta_scr )
    {
        beta_scr=fitness;
        beta_pos=position[i];
    }
    if(fitness>alpha_scr && fitness>beta_scr && fitness<delta_scr)
    {
        delta_scr=fitness;
        delta_pos=position[i];
    }

}

```

## Digital DNA Sequence Generation

```
public class DNAFunction
{
    DNAFunction()
    {

    }

    public double GC_content(String g)
    {
        double e1=0;
        try
        {
            int gc=0;
            for(int i=0;i<g.length();i++)
            {
                if(g.charAt(i)=='G' || g.charAt(i)=='C')
                    gc++;
            }
            if(gc==0)
                e1=0;
            else
                e1=((double)gc/(double)g.length())*100;

        }
        catch(Exception e)
        {
            e.printStackTrace();
        }
        return e1;
    }

    public double Temp_Melting(String g)
    {
        double e1=0;
        try
        {
            int gc=0;
            int at=0;
            for(int i=0;i<g.length();i++)
            {
                if(g.charAt(i)=='G' || g.charAt(i)=='C')
                    gc++;
                if(g.charAt(i)=='A' || g.charAt(i)=='T')
                    at++;
            }
        }
    }
}
```

```

        e1=(2*at)+(4*gc);
    }
    catch(Exception e)
    {
        e.printStackTrace();
    }
    return e1;
}
public double AT_GC_Ratio(String g)
{
    double e1=0;
    try
    {
        int gc=0;
        int at=0;
        for(int i=0;i<g.length();i++)
        {
            if(g.charAt(i)=='G' || g.charAt(i)=='C')
                gc++;
            if(g.charAt(i)=='A' || g.charAt(i)=='T')
                at++;
        }
        if(at==0 || gc==0)
            e1=0;
        else
            e1=(double)at/(double)gc;
    }
    catch(Exception e)
    {
        e.printStackTrace();
    }
    return e1;
}
}

```

## Ransomware Detection

```

public class Details
{
    //String inPath="test1.txt";
    String inPath="input1.txt";

    static String preData[][];
    static ArrayList ids=new ArrayList();
    static ArrayList cls=new ArrayList();
    static double gmodData[][];

```

```
static ArrayList gdataList=new ArrayList();
static double lower[];
static double upper[];
int Max_iteration=100;
static double bestScr[];
static ArrayList clsLt=new ArrayList();

static String gwocol[][];
static ArrayList selgwo=new ArrayList();

static String gwoFt="";
static String bsFt="";

static ArrayList selFt=new ArrayList();

static String newData[][];
static ArrayList DNA1=new ArrayList();
static ArrayList DNAcls=new ArrayList();
static ArrayList preInd=new ArrayList();

static ArrayList DNAFn=new ArrayList();
static String kmer[]={ "AA", "AC", "AG", "AT", "CA", "CC", "CG", "CT", "GA",
"GC", "GG", "GT", "TA", "TC", "TG", "TT"};

static Instances trainInst;
static double learnRate1=10;
static double regPara1=0.5;//0.1;
static double smoothPara1=0.6;//0.5;

static ArrayList testDNA=new ArrayList();

static double Tp1=0;
static double Tn1=0;
static double Fp1=0;
static double Fn1=0;
static double acc1=0;
static int afterPre=0;
static int afterFS=0;
static int gwofs=0;
static int binfs=0;

static double pre=0;
static double rec=0;
static double fmes=0;
static double gracc[];
static String para[];
static double bestAcc=0;
```

```
static double lnRate=0;  
static double rePara=0;  
static double smPara=0;
```

```
}
```