Vol.2019-DPS-178 No.15
Vol.2019-CSEC-84 No.15
2019/3/4

画像特徴量による自己防衛機能を有したマルウェアの検知に関する検討

小寺 建輝[†1]　房安 良和[†1]　泉 隆[†1]

概要：近年，マルウェアの亜種生成が高速化され，パターンファイルの作成が追い付かない現状となっている．このような亜種検知に関する問題を解決するため，機械学習等により亜種を検知・分類する研究が多く取り組まれている．その中でも，機械語命令列等の静的解析情報をマルウェアの特徴とした研究では亜種を高い精度で検知・分類できることが確認されている．しかし，この手法はコンパイラの種類・最適化レベルの変更，ジャンクコードの挿入，パッキング等の機械語命令列に影響を与える自己防衛機能を有したマルウェア亜種の検知・分類に対しては精度が低下することが考えられる．本稿では，マルウェアのバイナリデータを画像として扱うことにより，機械語命令列のみに依存せず，上述の自己防衛機能を有したマルウェアの検知に有効な画像特徴量について検討を行った．

キーワード：マルウェア検知，画像特徴量，自己防衛機能

A Study on Malware Detection with Self Defense Function by Image Features

TATEKI KODERA[†1]　FUSAYASU YOSHIKAZU[†1]
IZUMI TAKASHI[†1]

Abstract: In recent years, the generation of subspecies has been accelerated, and the creation of the pattern file can not keep up with the present condition. In order to solve the problem related to such subspecies detection, many studies for detecting and classifying subspecies by machine learning are underway. Among them, it has been confirmed that subspecies can be detected and classified with high accuracy in studies that characterize static analysis information such as machine language instruction sequences as features of malware. However, this method is not suitable for detecting and classifying malware subspecies having a self-defense function that affects machine language instruction sequences such as change of compiler type / optimization level, junk code insertion, packing etc. It is thought that it will decrease. In this paper, we dealt with binary data of malware as an image, and examined image feature quantity effective for detection of malware having the above self - defense function, without relying solely on machine language instruction sequence.

Keywords: Malware Detection, Image Features, Self-Defense Function

1. はじめに

近年，マルウェア亜種自動生成ツールの流通や，マルウェアのソースコード流出により，亜種生成が簡易化・高速化され，ウイルス定義ファイル等のパターンファイルの作成・配布が追い付かない現状となっている．例えば，2016年10月には，IoTマルウェアであるMiraiのソースコードが作成者によって公開されており，それを改変した亜種が大量に作成され，多くのIoTデバイスが感染の被害にあった[1]．このような亜種検知に関する問題を解決するため，統計的手法や機械学習により亜種を検知・分類する研究が現在多く取り組まれている．その中でも，機械語命令列等の静的解析情報をマルウェアの特徴とした研究[2][3]では亜種を高い精度で検知･分類できることが確認されている．しかしこの手法は，コンパイラの最適化レベルの変更，ジャンクコードの挿入，パッキング等，機械語命令列に影響を与える自己防衛機能を有したマルウェア亜種の検知・分類に対する精度の低下が考えられる．これに対し，パッキング等の影響を受けにくい特徴量としてファイルヘッダの情報を用いた研究[4]があるが，特徴量に用いているヘッダ情報の内容を攻撃者に知られることにより，ヘッダ情報が書き換えられた検体が作成され，検知が回避される課題があることが指摘されている．

そこで本研究では，自己防衛機能の影響を受けやすい機械語命令列や改竄が容易なファイルヘッダ情報等，ファイルの一部のみに着目した特徴量を用いるのではなく，マルウェアのバイナリデータを画像化し，ファイル全体のバイト単位の情報（位置や値）から抽出される画像特徴量を用いたマルウェア亜種の検知・分類について検討する．

本稿では，画像特徴量としてGist特徴量[5]，LBP(Local Binary Pattern)特徴量[6]，HLAC(Higher-order Local Auto-Correction)特徴量[7]の3種類を採用し，機械語命令列に基づく特徴量を用いたときよりも，自己防衛機能（コンパイラの最適化レベルの変更，ジャンクコードの挿入，パッキング）を有したマルウェアの検知に対して有効であることを検証した．

†1 日本大学
Nihon University

2. 関連研究と本研究の位置づけ

本章では，本研究の比較対象となる機械語命令列に着目した関連研究と本研究で自己防衛機能に有効な特徴量として検討している画像特徴量に着目した関連研究について述べる．

2.1 機械語命令列に着目した研究

マルウェアの識別，機能推定等に用いる特徴量として，機械語命令列に着目した関連研究を以下に述べる．

2.1.1 オペコードの N-gram を特徴にした研究

N-gram とは，N 個の連続した要素の出現頻度を特徴量とするものである．例えば 2-gram では，{ push, call, lea, push, call, cmp, je}というオペコードの並びがあったとき，2 つの連続したオペコード{ (push, call), (call, lea), (lea, push), (call, cmp), (cmp, je) }がそれぞれ何回出現したかを特徴量として表す．関連研究[2]では，この N-gram を用いてマルウェアの分類を行っている．しかし，N-gram では，オペコードの順序を考慮するため，オペコードの並べ替えが発生した際に特徴量に大きな影響を及ぼしてしまう．そこで，オペコードの順序を考慮しない N-perm についての提案も行っている．

2.1.2 バイトコードの出現頻度を特徴にした研究

関連研究[3]では，バイトコードの出現頻度を特徴としたマルウェアの類似度から機能推定することを目的としており，挙動が異なる 2 つのマルウェア間の類似度を低くするため，2byte 以上のオペコードに必須である 0x0f(prefix)の後ろの 1 byte に着目している．そしてマルウェア間の類似度(正規化相互相関を用いて計算)を算出する際に，0x0f の後の 1byte の出現頻度を用いることで，バイナリデータ全体のバイトコードの出現頻度を用いた場合よりも，動作が類似するマルウェアとの類似度を高く，動作が異なるマルウェアとの類似度を低く算出できることが報告されている．

2.2 画像特徴量に着目した研究

マルウェアの識別等に用いる特徴量として，画像特徴量に着目した関連研究を以下に述べる．

2.2.1 Gist 特徴量を用いた研究

関連研究[8]は，マルウェアを画像化し，画像認識によってマルウェア亜種をファミリごとに分類する手法を提案した最初の研究である．生成された画像がテクスチャ画像に類似していることから，画像特徴量としてテクスチャ画像の識別に有効な Gist 特徴量を用いており，高い識別精度でマルウェアを分類できたことが報告されている．これは，亜種が元のコードの一部のみを改変して作成されるため，元のマルウェアとその亜種，つまり同一ファミリでは視覚的に類似した画像(テクスチャ画像)が得られるためである．

しかし，正常ファイルとマルウェア亜種を識別して検知することに対する有効性ついては検証されていない．

2.2.2 LBP 特徴量を用いた研究

関連研究[9]は，2.2.1 項で述べた関連研究[8]と同様に，画像認識によってマルウェア亜種をファミリごとに分類する研究である．画像特徴量として LBP 特徴量を採用することで Gist 特徴量を用いた時よりも，識別精度が高くなったことが報告されている．また，関連研究[8]と同様に正常ファイルとマルウェア亜種を識別して検知することに対する有効性については検証されていない．

2.3 本研究の位置づけ

2.1 節に示した，機械語命令列に着目した研究では，コード領域である.text セクション等の情報のみに着目しているため，コンパイラの最適化レベルの変更等，機械語命令列に変更を加える自己防衛機能の影響が特徴量抽出に強く現れてしまう．一方で 2.2 節の画像特徴量に着目した研究では，コンパイラの最適化レベルの変更の影響を受けにくいセクション(.data, .rdata, .idata, .edata 等)を含むバイナリデータ全体から特徴抽出することが可能である．しかし関連研究では，正常ファイルとマルウェア亜種を識別して検知することに対する有効性については検証されていない．

そこで，本研究では画像特徴量を用いることで，コンパイラの最適化レベルの変更が施されたマルウェアの検知に加え，ジャンクコードの挿入・パッキング等の自己防衛機能を有するマルウェアの検知に対する検討を行う．

3. マルウェア画像化と画像特徴量

本章では，関連研究[8]で提案されたマルウェアのバイナリデータを画像に変換する手法と，本研究で利用するテクスチャ画像特徴量について述べる．

3.1 マルウェア画像化

マルウェアを画像化する手法を以下に示す．また実際にマルウェアを画像化した例を図 1 に示す.

(1) 対象ファイルを 1Byte(8bit)ずつ読み込み 1 次元配列に格納する
(2) ファイルサイズ(配列の要素数)に応じて幅を決定し，2 次元配列に変換する(表 1 参照)
(3) 配列の要素の値は 8bit であり, 0-255 の範囲であるため，その値を画素値として 256 階調のグレースケール画像を生成する

Vol.2019-DPS-178 No.15
Vol.2019-CSEC-84 No.15
2019/3/4

表 1　ファイルサイズと画像の幅
Table 1　File size and image width

ファイルサイズ	画像の幅[pixel]
<10kB	32
10kB – 30kB	64
30kB – 60kB	128
60kB – 100kB	256
100kB – 200kB	384
200kB – 500kB	512
500kB – 1000kB	768
> 1000kB	1024

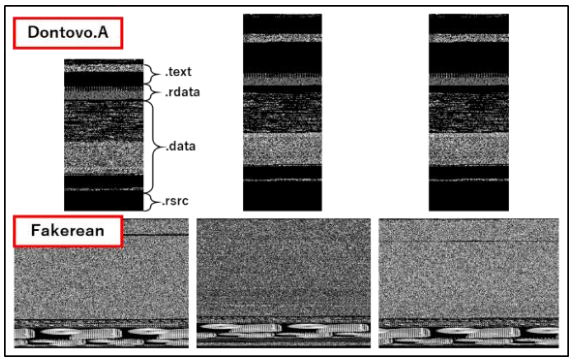

図 1　マルウェア画像化の例
Figure 1　Example of Malware Imaging

図 1 より，マルウェアを画像化するとセクション(.text や.data 等)ごとに異なるテクスチャパターンが現れることが確認できる．また，亜種が元のコードの一部のみを改変して作成されることから，同一ファミリ(Dontovo.A，Fakerean)で類似した画像が生成されていることも確認できる．

3.2 画像特徴量

本研究では，テクスチャ画像特徴量として，2.2 節に示した関連研究[8][9]で用いられている Gist 特徴量，LBP 特徴量に加え，テクスチャ認識の研究分野で高い成果を上げている HLAC 特徴量についても検討を行い，計 3 種類の画像特徴量を用いる．

3.2.1 Gist 特徴量[5]

Gist 特徴量は，画像にガボールフィルタを適用することで，取得するテクスチャ画像特徴量である．ガボールフィルタを用いることで，テクスチャの位置，向き，粗さなどを表現することができる．式(1)に周波数をλ，方向をφとしたときの 2 次元ガボールフィルタの式を示す．

$$G(x,y,\lambda,\varphi) = e^{-\frac{x^2+y^2}{2\sigma^2}} e^{2\pi\lambda(x\cos\varphi+y\sin\varphi)} \quad (1)$$

以下に，ガボールフィルタを用いて Gist 特徴量を抽出する手順を示す．

(1) N_λ種類の周波数とN_φ種類の方向の計$N_\lambda \times N_\varphi$種類のガボールフィルタを画像に適用する
(2) 画像を $W\times W$ の小領域に分割する．各小領域においてガボールフィルタ適用後のベクトルの平均を計算する
(3) 各小領域の平均ベクトルを結合することで，Gist 特徴量が算出される

3.2.2 LBP 特徴量[6]

LBP 特徴量では，まず対象画像の 3×3 ピクセル領域に着目する．その領域の中心画素と 8 近傍の各画素の値を比較して，中心画素の方が大きい場合は”0”，小さい場合は”1”として 8 近傍の画素を 2 値化する．そして，2 値化された 8 近傍の画素を任意の規則で並べて，8bit の 2 進数として置き換えたものが，中心画素の LBP 値となる．LBP 特徴量は，この処理を画像全体に対して行った際の，LBP 値の頻度を記述したヒストグラムの特徴量である．

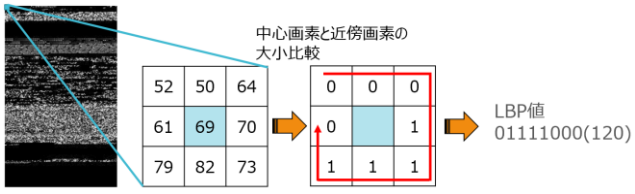

図 2　LBP 値の算出
Figure 2　Calculation of LBP value

3.2.3 HLAC 特徴量[7]

HLAC 特徴量は，自己相関関数を N 次へと拡張した N 次自己相関関数により計算される画像特徴量であり，テクスチャ認識で有効性を示している．着目画素を r，r からの相対的な変位方向を$(a_1,\cdots a_N)$としたときの高次自己相関関数を式(2)に示す．

$$x(a_1,\ldots,a_N) = \int f(r)f(r+a_1)\cdots f(r+a_N)dr \quad (2)$$

このとき，fは画素値を示す．高次自己相関関数の次数 N は任意に決めることが可能であり，本稿では N=2 とする．また，変位方向は 3×3 ピクセル領域に限定し，変位方向の重複を含んだ 35 個のマスクパターンにより特徴抽出を行う．このとき，計算される特徴量は，各マスクパターンのパワースペクトルを表している．N=2 としたときのマスクパターンを図 3 に示す．

Vol.2019-DPS-178 No.15
Vol.2019-CSEC-84 No.15
2019/3/4

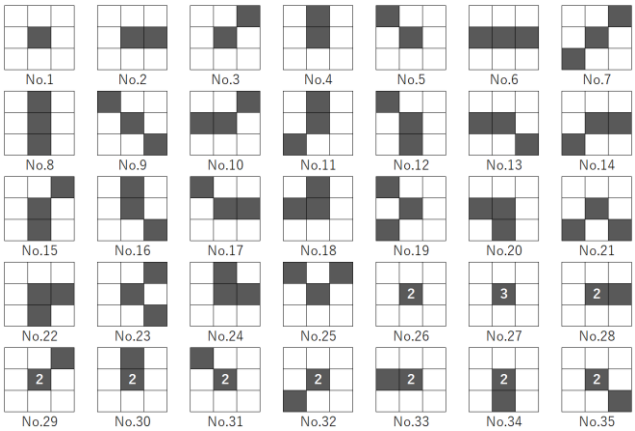

図 3　*N*=2 とした時のマスクパターン(重複を含む)

Figure 3　Mask pattern when *N*=2

4. 実験

本章では，画像特徴量として Gist 特徴量，LBP 特徴量，HLAC 特徴量の 3 種類を採用し，自己防衛機能（コンパイラの最適化レベルの変更，ジャンクコードの挿入，パッキング）を有したマルウェアの検知に関する有効性について機械語命令列に基づいた特徴量(2-gram, 3-gram, 0x0f 後のバイトコード出現頻度)を用いた場合との比較を行う．なお，4.1 節と 4.2 節は検体間の類似度からマルウェアを検知する手法，4.3 節は機械学習を用いてマルウェアを検知する手法に対しての検証を行っている．

4.1 コンパイラの最適化レベルの変更

gcc の最適化レベル-O0 でコンパイルしたマルウェア(検体番号 0)と最適化レベルを-O1, -O2, -O3, -Ofast と変更してコンパイルしたマルウェア(検体番号 1~4)の類似度(cos 類似度で計算)を各特徴量で算出し比較を行った．この際，利用する gcc のバージョンは 5.4.0 である．gcc における各最適化レベルの説明を表 2 に示す．また本実験では，評価対象のマルウェアとして，IoT マルウェアの 1 種でソースコードが流出している Bashlite を用いた．同一ソースコードで，コンパイルの最適化レベルが異なる 5 つのマルウェア(Bashlite)を画像化したものを図 4 に示し，各特徴量で検体番号 0 のマルウェアと検体番号 1~4 のマルウェアの類似度を算出した結果を図 5 に示す．

表 2　検体番号と gcc の最適化レベル

Table 2　Malware number and gcc optimization level

検体番号	最適化レベル	説明
0	-O0	最適化しない
1	-O1	実行時間とサイズを短縮するように最適化
2	-O2	-O1 をさらに最適化
3	-O3	-O2 をさらに最適化
4	-Ofast	実行速度重視で最適化

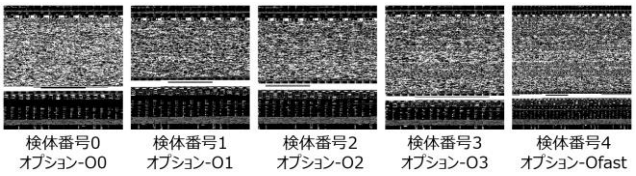

図 4　最適化レベルが異なる 5 つの Bashlite の画像

Figure 4　Images of 5 Bashlite with defferent optimization levels

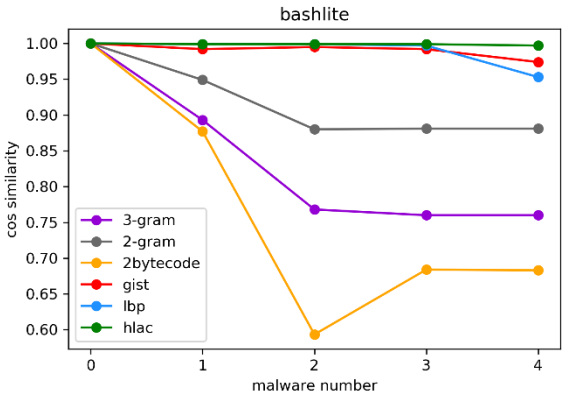

図 5　類似度の比較 1

Figure 5　Comparison of similarity -1

図 5 より，各画像特徴量(Gist, LBP, HLAC)を用いたときの検体番号 0 のマルウェアと検体番号 1~4 のマルウェアの類似度は全て 0.95 以上を示しており，機械語命令列に着目して抽出した特徴量(2-gram, 3-gram, 0x0f 後のバイトコード出現頻度)を用いた場合よりも高い類似度を示すことが確認された．機械語命令列に基づいた特徴量では，最適化レベル-O2 以上にしたときの類似度の低下が顕著であり，特に 3-gram, 0x0f 後のバイトコード出現頻度を用いた場合では類似度が 0.85 以下となった．また，図 4 よりコンパイルの最適化レベルが異なる 5 のマルウェアの画像を比較すると，画像間のテクスチャの見た目の類似性が維持されており，最適化レベルの違いの影響が弱いことが確認できる．これらのことからも，画像特徴量によるマルウェア間の類似度を用いて，コンパイラの最適化レベルが変更されたマルウェアを検知できることが考えられる．

4.2 ジャンクコードが挿入されたマルウェア

ジャンクコードの挿入とは無駄なコードや偽の命令をバイナリに追加することである．例えば，本実験で用いる Lolyda.AA3 というファミリのマルウェアでは図 6 に示すような単純なジャンクコードが挿入されている．検体番号 0 と検体番号 1 は同様のジャンクコードが挿入されており，検体番号 2, 3 は検体番号 0, 1 とは異なるジャンクコードが挿入されている．検体番号 0~3 を画像化したものを図 7 に

示し，各特徴量で検体番号 0 のマルウェアと検体番号 1~3 のマルウェアの類似度を算出した結果を図 8 に示す．

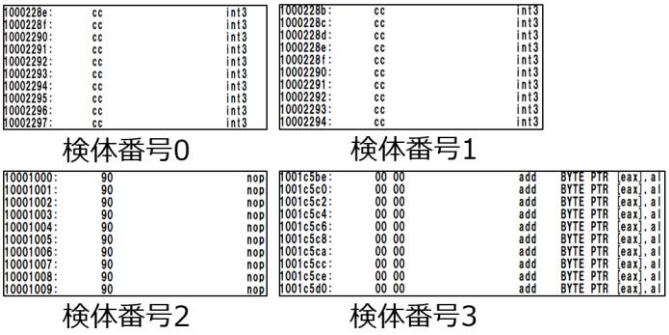

図 6　各検体に挿入されたジャンクコード

Figure 6　Junk code inserted in each malware

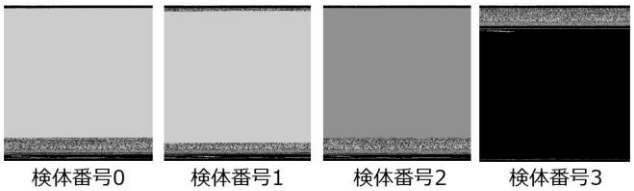

図 7　ジャンクコードが挿入された 4 つの Lolyda.AA3 の画像

Figure 7　Images of 4 Lolyda.AA3 with junk code inserted

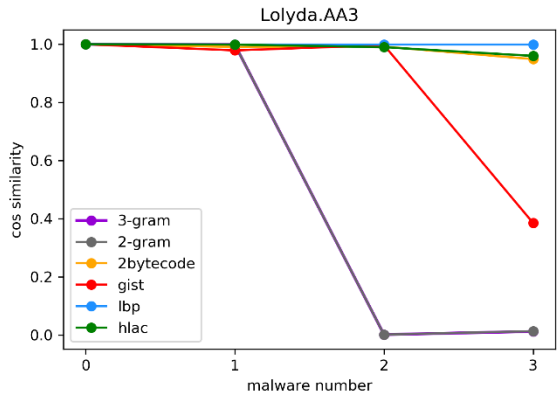

図 8　類似度の比較 2

Figure 8　Comparison of similarity -2

図 8 より，検体番号 0 と同様のジャンクコードが挿入された検体番号 1 に関しては，全ての特徴量で高い類似度を示した．また，LBP 特徴量，HLAC 特徴量及び 0x0f 後のバイトコード出現頻度を特徴量にしたときの検体番号 0 のマルウェアと検体番号 2, 3 のマルウェアの類似度は全て 0.95 以上を示しており，機械語命令列の 2-gram, 3-gram を特徴量とした場合よりも高い類似度を示すことが確認された．Gist 特徴量に関しては，検体番号 3 に対しては低い類似度を示したものの，検体番号 2 までのマルウェアには高い類似度を示した．また，4.1 節の結果とは異なり，本実験では機械語命令列に基づいている 0x0f 後のバイトコード出現頻度を特徴量として用いたとき，検体 1~3 すべてに対して高い類似度を示した．これは，今回用いたジャンクコードに 2byte 以上のオペコードが用いられていないことから，特徴量の算出にジャンクコードの影響が及ばなかったためである．しかし，ジャンクコードに 2byte 以上の様々なオペコードが用いられた場合に，検体間の類似度が低下することが考えられる．また，本実験において LBP 特徴量は，全ての検体で 0.99 以上の類似度を示し，他の画像特徴量よりも高い類似度を示した．これは，LBP 特徴量では中心画素と周辺画素の輝度差を利用して LBP 値を算出するため，輝度変化に頑健であり，図 6 のような異なる値のジャンクコードが挿入されたとしても同一の特徴量として捉えることができるためである．

本実験ではジャンクコードが挿入される前のマルウェアとの類似度を算出していないため対象は限定されるが，Lolyda.AA3 のようにジャンクコードが挿入されていることが前提のファミリのマルウェアの検知に対して，LBP や HLAC 等の画像特徴量の類似度を用いることが有効であることを確認した．

4.3　自己防衛機能を有するマルウェアの検知実験

検体間の類似度ではなく機械学習を用いてマルウェアを検知する際に，画像特徴量が自己防衛機能を有するマルウェアの検知に有効か否かを判断するためは，マルウェアと正常ファイルを識別し，その精度を検証する必要がある．

本実験では，以下の手順により，ファミリごとにマルウェアの検知率と正常ファイルの誤検知率を検証する．

(1)　対象ファミリ内の全データを学習用データと検証用データに分割する
(2)　学習用データを用いて，対象ファミリの亜種検知モデルを構築する
(3)　構築した亜種検知モデルに，対象ファミリの検証用データと正常ファイルのデータを入力することで識別を行い，検知率と誤検知率を算出する

このとき，亜種検知モデルの構築及びマルウェアと正常ファイルの識別に利用する機械学習アルゴリズムとして Isolation Forest[10]を採用した．また，利用するデータセットの概要を表 3 に示し，10 分割交差検証によってファミリごとに検知率と誤検知率を算出した結果を表 4 に示す．

表 3　データセットの概要

Table 3　Dataset overview

タイプ	検体数	概要
Lolyda.AA3	123	全ての検体にジャンクコードが挿入されている
Lolyda.AT	159	全ての検体が UPX でパッキングされている
正常ファイル	913	Windows 8, 10 (32bit),MinGW,OSDN から収集した PE ファイルを利用

表 4　検知率と誤検知率の比較
Table 4　Comparison of detection rate and false detection rate

	Lolyda.AA3		Lolyda.AT	
	検知率	誤検知率	検知率	誤検知率
2-gram	97.6%	0.0%	88.7%	1.1%
3-gram	97.6%	0.0%	65.4%	0.2%
bytecode	65.9%	0.0%	52.2%	3.7%
Gist	98.4%	0.0%	99.4%	0.0%
LBP	98.4%	0.1%	75.5%	2.6%
HLAC	90.2%	0.1%	97.5%	0.3%

表 4 より，Lolyda.AA3 では，Gist 特徴量，LBP 特徴量，HLAC 特徴量及び N-gram を特徴量にしたときの検知率が 90%以上，誤検知率が約 0%と高い精度を示した．また Lolyda.AT では，Gist 特徴量，HLAC 特徴量を用いたときに，検知率 97%以上，誤検知率が約 0%で高い精度を示している．いずれのファミリにおいても，画像特徴量 (Lolyda.AA3 では Gist，LBP，HLAC 特徴量，Lolyda.AT では Gist, HLAC 特徴量)を用いることで高い精度でマルウェアを検知できることが確認された．

上述の結果と学習・検証に用いたデータセットの内容を考慮すると，画像特徴量を用いる手法においては，ジャンクコードが挿入されたマルウェアまたは，パッキングされたマルウェアを学習データとして利用することにより，同一ファミリで同じようにジャンクコードが挿入されたマルウェアやパッキングされたマルウェアを検知できるようになることが確認された．

5. まとめ

本稿では，画像特徴量が自己防衛機能(コンパイラの最適化レベルの変更，ジャンクコードの挿入，パッキング)を有するマルウェアの検知に有効か否かを検証するため，テクスチャ認識に有効な Gist 特徴量，LBP 特徴量，HLAC 特徴量という 3 種類の画像特徴量について検討を行った．このように，テクスチャ認識に有効な画像特徴量を用いたのは，画像化したマルウェアがテクスチャ画像に類似しているためである．そして，これらの画像特徴量を用いて，異なる最適化レベルでコンパイルしたマルウェア間の類似度や異なるジャンクコードを挿入したマルウェア間の類似度を算出した結果，機械語命令列に基づいた特徴量よりも高い類似度を示すことができた．このことら，画像特徴量によるマルウェア間の類似度を用いて，コンパイラの最適化レベルの変更や異なるジャンクコードの挿入等の自己防衛機能を有したマルウェアを検知できることが考えられる．

また，検体間の類似度ではなく機械学習を用いてマルウェアを検知する際に，画像特徴量が自己防衛機能を有するマルウェアの検知に有効か否かを検証するため，検証対象の各ファミリ(ジャンクコードの挿入やパッキング等の自己防衛機能を有したマルウェアのファミリ)で亜種検知モデルを構築し，ファミリごとに検知率と誤検知率を算出した．その結果，いずれのファミリにおいても，画像特徴量を用いて高い精度でマルウェアを検知できることが確認された. このことから, 画像特徴量を用いる手法においては，ジャンクコードの挿入またはパッキングされたマルウェアを学習データとして利用することにより，同一ファミリで同じようにジャンクコードの挿入やパッキングされたマルウェアを検知できることが考えられる．

今後は，本稿では検証が行えなかったジャンクコードが挿入される前後のマルウェア間の類似度やパッキングされる前後のマルウェア間の類似度の比較を行う．また，画像のテクスチャを乱し，画像特徴量を用いたマルウェア検知を回避する自己防衛機能に対する検討を行う．

参考文献

[1] トレンドマイクロ株式会社：“Web アプリの脆弱性を利用する「Miori」など複数の「Mirai」亜種の拡散を確認”. https://blog.trendmicro.co.jp/archives/20045, (2019-02).
[2] Md.Karim, E., Walenstein, A., Lakhotia, A. and Parida, .L:”Malware phylogeny generation using permutations of code”, European Research Journal of Computer Virology, Vol.1, No.1-2, pp.13-23 (2005-12).
[3] 大久保 諒, 森井 昌克, 伊沢 亮一, 中尾 康二, 井上 大介. : マルウェアの部分コードによる類似度判定と機能推定. FIT2012 第 11 回情報科学技術フォーラム, L-029(2012).
[4] 田中 恭之，後藤 厚宏：”統計的方法を用いた未知マルウェアの検出手法の提案と評価”，情報処理学会論文誌, Vol.57，No.9，pp.2003-2011(2016-09).
[5] A. Olivia and A. Torralba : ” Modeling the shape of a scene: a holistic representation of the spatial envelope”, Intl. Journal of Computer Vision, Vol.42, No.3, pp.145–175(2001)
[6] DC.He and L.Wang : “Texture Unit, Texture Spectrum, And Texture Analysis”, Geoscience and Remote Sensing, IEEE Transactions on, Vol.28, pp.509-512(1990)
[7] N.Otsu and T.Kurita : ”A new scheme for practical flexible and intelligent vision systems”, Proc. IAPR Workshop on Computer vision, pp.431-435(1998-10)
[8] L. Nataraj, et al. : ” Malware Images:Visualization and Automatic Classification”，VizSec’11(2011-07)
[9] Jhu-Sin Luo, Dan Chia-Tien Lo:"Binary Malware image Classification using Machine Learning with Local Binary Pattern", Proceedings of 2017 IEEE International Conference on Big Data (BIGDATA), pp. 4664-4667(2017)
[10] Fei Tony Liu, et al. : “Isolation-Based Anomaly Detection”, ACM Transactions on Knowledge Discovery from Data(TKDD), Vol.6, No.1, pp.1-39(2013-03)