Computer Security Symposium 2020
26 - 29 October 2020

データ欠損を起こしたマルウェアの機械学習による同定

小久保 博崇[1,a)]　江田 智尊[1]　大山 恵弘[2]

概要：サイバー攻撃の被害状況を特定するにあたって，何のマルウェアが攻撃に使われたかを知ることは重要である．しかし，調査妨害を目的とした自己消去を行うマルウェアも存在する．消去されたファイルはストレージの使用とともに情報が欠損していくため，フォレンジック技術を使用しても完全な形でマルウェアを復元できないこともある．本稿では，このようなデータの欠損を起こしたマルウェア（欠損マルウェア）に着目する．欠損マルウェアをアンチウイルスによって同定することは難しいという実験結果を，過去に著者らは示した．本稿では，欠損マルウェアからマルウェア名をどの程度同定できるかを，機械学習技術を用いて実験した．欠損マルウェアはデータの欠損を起こしており，ヘッダ情報や挙動解析情報などの有用な特徴が得られないため，分類には画像特徴量を使用した．その結果，アンチウイルスによる同定に致命的な悪影響を与えるファイル先頭の欠損があったとしても，欠損前のマルウェア検体を学習に用いれば約 97%の精度，欠損前のマルウェア検体と同種の検体を学習に用いれば約 48%の精度で，マルウェア名の同定が可能であることがわかった．

キーワード：マルウェア，機械学習，データ欠損，マルウェア同定，フォレンジック

Identification of Corrupted Malware using Machine Learning

Kokubo Hirotaka[1,a)]　Koda Satoru[1]　Oyama Yoshihiro[2]

Abstract: In order to know the damage situation of cyber attack, it is important to know what kind of malware was used for the attack. Some malware erases itself to prevent investigation. Erased malware can be recovered by digital forensics technology, but data loss can occur. In this paper, we identify malware names from corrupted malware binary by using machine learning. We use image feature values to perform identification because header information and behavior analysis information cannot be used due to data loss. As a result, it was found that the malware name can be identified with high accuracy even if the malware file head is missing.

Keywords: Malware, Machine Learning, Data Loss, Malware Identification, Digital Forensics

1. はじめに

サイバー攻撃を受けたとき，被害状況を特定するために，どのようなマルウェアが攻撃に使われたかを知ることは重要である．攻撃に使われたマルウェアが同定できれば，マルウェアの除去方法のみならず情報流出の可能性や組織内感染拡大の状況把握など，インシデントレスポンスに役立つ情報を得ることができる．どのようなマルウェアが使われたかを知る手段のひとつは，マルウェアに感染したマシンのストレージを調査することである．マルウェアがファイルとしてコンピュータ上に残存していれば，アンチウイルスソフトによるスキャンや専門家の調査により検体を取得し，どのマルウェアにより被害を受けたかを知ることができる．しかし，役目を終えたマルウェアは常にファイル

[1] 株式会社富士通研究所
FUJITSU LABORATORIES LTD.
[2] 筑波大学
University of Tsukuba
[a)] kokubo.hirotaka@fujitsu.com

として残存しているわけではない．不要になったマルウェアは，攻撃者やマルウェア自身によって削除されてしまうことがある．マルウェアに限らず，一度削除されてしまったファイルは通常のファイル操作では復元することはできないが，デジタルフォレンジック技術を用いることで復元できる場合がある [1]．ストレージがハードディスクの場合，FAT や NTFS など Windows 環境においてよく使用されるファイルシステムでは，OS 上からファイルを削除してもファイルのデータ自体はストレージ上に残存する．削除されたファイルは，格納されていた領域の削除フラグが立ち，OS 上から参照できなくなるだけであり，ただちにデータが完全に消去されるわけではない．よって，この領域に格納されているデータを集めて結合することで，元のファイルを復元することができる．しかし，削除フラグが立った領域は未使用領域として扱われるため，OS 上での新規ファイル生成や既存ファイルへの追記により，この領域に別ファイルのデータが書き込まれることがある．通常，コンピュータの使用にはファイル生成が伴うため，時間の経過とともに削除されたファイルが残存する領域は新しいデータにより上書きされてしまう可能性は高まる [2]．つまり，削除されてからある程度時間が経過してしまったマルウェアは，ストレージから復元しても完全な形で復元することはできず，データの欠損を起こしていることがある．このようなマルウェアのデータ欠損は，アンチウイルスによるマルウェア同定に著しい悪影響を与えることが著者らの研究で判明している [3]．しかし，攻撃による被害状況を特定する上では，データ欠損を起こしたマルウェア（以下，欠損マルウェアと呼ぶ）に対してもマルウェア同定を行えることが望ましい．

そこで本研究では，機械学習を用いてこのような欠損マルウェアの同定を試み，フォレンジックによって復元したマルウェアが欠損していたとしても元のマルウェア名の同定を可能にする手法を提案する．対象とするマルウェアは x86 向けの Windows 用 PE 形式の 32bit バイナリプログラムとする．現実環境でのデータ欠損を模倣してマルウェアを人工的に欠損させることで欠損マルウェアを生成する．機械学習を用いてマルウェアの分類を行う際に使う特徴量は，表層解析情報や挙動解析情報由来の値がよく使われるが，本研究ではマルウェアバイナリをグレースケール画像化したものを使用する．マルウェアおよび欠損マルウェアを画像化した上でデータセットを生成し，Convolutional Neural Network（以下，CNN と呼ぶ）や Triplet Network [4]（以下，TN と呼ぶ）を訓練し，欠損マルウェアから元のマルウェアを同定するモデルを生成する．そして生成したモデルを使って欠損マルウェアの同定を行い，同定精度を示す．その結果，訓練データに含まれている無欠損マルウェアを 4,096 bytes 欠損させ同定を試みた場合は 97.3%，訓練データに含まれていない無欠損マルウェアを同量欠損させ同定を試みた場合は 48%の同定精度であった．

本論文の貢献は以下のとおりである．

(1) バイナリを画像化した上で行う機械学習によるマルウェア同定手法は，データ欠損にある程度の耐性があることを示した．
(2) 機械学習によるマルウェア名の推定根拠を可視化し，バイナリ中の様々な領域やファイルサイズを推定の手がかりとしていることが欠損耐性の向上に寄与している可能性を示した．

2. 関連研究

欠損のないマルウェアを画像化した上で分類を試みる研究としては，[5][6][7] が挙げられる．Nataraj らの研究 [5] では，パックされていないマルウェアバイナリをグレースケール画像化し，画像から GIST 特徴量を抽出し k-近傍法を用いてマルウェア分類を行っている．検体数 9,458 個のマルウェア群の分類を試みており，25 ファミリの分類問題で 98%の精度を示した．パックされたマルウェアに対しても同様の実験を行っているが，パックされたマルウェアを元のファミリとは独立した新しいファミリとして扱っている．Hsiao らの研究 [6] では，グレースケール画像化したマルウェアに対して Average Hash [8] を使って画像を再ラベリングした後，Siamese Network を用いて one-shot 学習を行った．結果，N-way one-shot 問題においては，N が 2 のとき 92%，N が 15 のとき 42%の精度であった．矢倉らの研究 [7] では，パックされたマルウェアを含むマルウェアバイナリ群を画像化し，注意機構を持つ CNN を適用することで，マルウェアの静的解析を効率化する手法を提案している．マルウェア画像を CNN に入力して分類を行う際に高い重要度を持つ画像領域を特定し，該当領域のコードあるいはデータを抽出し提示することで，マルウェア解析作業の効率化を図っている．CNN によるマルウェア分類精度の評価も行っており，542 ファミリの分類問題で Top-1 Error 率（最も当てはまる確率が高いと推定されたクラスが正解クラスと一致していない割合）は 50.97%であった．

マルウェアの欠損が同定に与える影響の研究として，著者らの過去の研究 [3] を挙げる．この研究では，マルウェアを自然な形で人工的に欠損させ，欠損がアンチウイルスによるマルウェア同定にどのような影響を与えるかを調査した．その結果，アンチウイルスや欠損箇所にもよるが，欠損前は同定できていた検体に対する同定成功率が，4,096bytes の欠損により 10%から 80%程度まで低下することを確認した．特にファイル先頭から 4,096 bytes の欠損はアンチウイルスによる同定に致命的な影響を与え，アンチウイルスによらず同定成功率はほぼ 0%まで低下した．

表 1 検体情報

項目	値
検体数	855 検体
マルウェア名種類数	171 種
パック済検体数	0 検体
サイズの最小値	5,598 bytes
サイズの平均値	679,089 bytes
サイズの最大値	4,184,486 bytes
サイズの標準偏差	812,625 bytes

3. 実験

本章では，機械学習により欠損マルウェアのマルウェア名同定を行う実験について詳細を述べる．

3.1 データセット生成

本節では，使用した無欠損マルウェア群の詳細，欠損マルウェアの生成方法，マルウェアの画像化手法について説明する．

3.1.1 使用する無欠損検体群

実験に使用する無欠損マルウェア群は，我々が独自に収集した x86 向けの PE 形式のマルウェア 855 検体である．検体の情報を**表 1** に示す．各検体の MD5 ハッシュ値はすべて異なる．これら検体群を VirusTotal [9] に与え Microsoft 社のアンチウイルス製品によって，無欠損マルウェア群のマルウェア名を命名した．検体群中にマルウェア名 [10] は 171 種存在しており，同一の検体がそれぞれ 5 検体ずつ含まれている．ファイルサイズの平均値は約 663KiB で，標準偏差は約 794KiB である．すべての検体に対して Cylance 社のパッカー検出ツール PyPackerDetect [11] を適用し，パックの痕跡が見つからないことを確認している．ただしこれは，全ての検体が確実にパックされていないことを保証するわけではなく，ツールによる検出を免れたパック済み検体が混入している可能性は残っている．

3.1.2 欠損検体群の生成

機械学習モデルの訓練および検証には，前項で述べた無欠損マルウェア群だけではなく，無欠損マルウェア群から生成した欠損マルウェア群も用いる．本項では，欠損マルウェアの生成方法について述べる．

著者らが過去に行った調査 [3] によると，NTFS において自然なデータ欠損はクラスタサイズの整数倍アドレスから発生する．新しく生成されたファイルデータによる古いファイルデータの上書きが当該アドレスから始まり，新しいファイルデータの書き込み終了後は，終了アドレスからクラスタサイズの整数倍に到達するまでゼロが書き込まれる．一部の環境ではクラスタサイズ単位ではなくセクタサイズ単位での欠損が起こったが，ゼロ埋めの挙動については変化はなかった．

この結果を受けて，本研究ではクラスタサイズの整数倍のファイルオフセットから，クラスタサイズ分のデータをゼロで上書きすることにより，人工的に欠損マルウェアを生成するものとする．我々の環境ではクラスタサイズは 4,096 bytes（0x1000 bytes）であるため，0x1000 の整数倍オフセットから 0x1000 bytes 分をゼロで上書きすることを，1 クラスタの欠損と呼称する．自然なデータ欠損では，欠損した領域がすべてゼロで上書きされるわけではなく，欠損領域の先頭には上書きに使われたファイルデータが書き込まれている．しかし，これを模して何らかの実ファイルデータで上書きすることで欠損処理を行った場合，その上書きしたデータの内容が実験結果に影響する可能性がある．よって，実ファイルデータによる上書きは採用せず，全検体を一律にゼロで欠損させるものとする．

3.1.3 検体群の画像化

前項までの処理で得た検体群は，画像化処理を施してから機械学習モデルの訓練や検証に使用する．検体を表層解析することにより得られるヘッダ情報やインポート情報，検体を挙動解析することで得られる API コール列などは，マルウェアの同定を行う上では本来非常に重要な情報である．しかし，欠損マルウェアはデータの欠損を起こしているため，それら重要な情報が破壊され取得できない可能性がある．特にファイル先頭付近の欠損は，PE ヘッダなどから得られる有用な表層解析情報を破壊する [3]．このような欠損マルウェアの特性を考慮し，他の特徴量よりもデータ欠損に耐性があると考えられるマルウェアバイナリ画像を特徴量として使用する．

画像化手法は，既存研究 [5][6] で用いられている手法を概ね踏襲し，以下のようにする．検体バイナリの各バイトの値を，グレースケール画像における各ピクセルの明度として使用して画像化を行う．値が 0 に近づくほど対応するピクセルは黒くなり，255 に近づくほど白くなる．画像の横幅は，検体バイナリのサイズによって変動させる．10KiB 未満ならば 32 ピクセル，30KiB 未満なら 64 ピクセル，60KiB 未満なら 128 ピクセル，100KiB 未満なら 256 ピクセル，200KiB 未満なら 384 ピクセル，500KiB 未満なら 512 ピクセル，1000KiB 未満なら 768 ピクセル，1000KiB 以上なら 1024 ピクセルとした．検体バイナリのサイズが画像の横幅の整数倍と一致しない場合，画像の横幅の整数倍と一致するまでゼロでパディングを行った．

このように生成したマルウェア画像を，さらに横幅 200 ピクセル，縦幅 200 ピクセルの固定サイズのマルウェア画像に変換する．元画像サイズが横幅と縦幅ともに 200 ピクセル以下である場合は，全ピクセルの明度が 0 である 200 × 200 ピクセルの背景画像を用意し，背景画像に対して元画像の左上を重ね合わせるように貼り付ける．元画像サイズの横幅または縦幅が 200 ピクセルを超える場合は，縦横比を保持したまま長辺が 200 ピクセルとなるように元画像を縮小する．縮小時の補間手法には BICUBIC 法を使用し

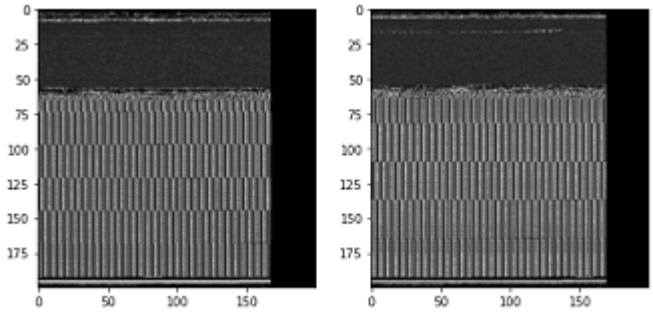

図 1 画像化した Worm:Win32/Gamarue

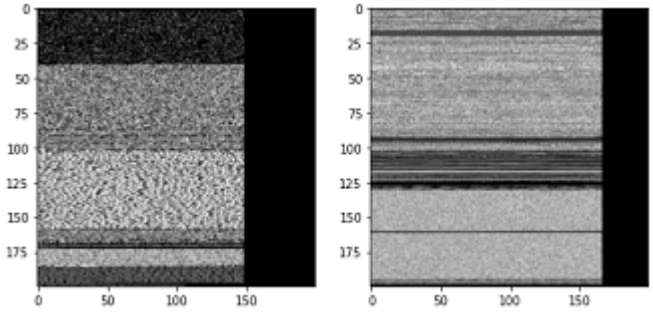

図 2 画像化した Trojan:Win32/Skeeyah.HK!MTB

た．縮小後は，こちらも同様に全ピクセルの明度が 0 である 200 × 200 ピクセルの背景画像を用意し，背景画像に対して元画像の左上を重ね合わせるように貼り付ける．

この手法でマルウェアを画像化した例を**図 1** および**図 2** に示す．図 1 はハッシュ値の異なる 2 検体の Worm:Win32/Gamarue を画像化した図であり，図 2 はハッシュ値の異なる 2 検体の Trojan:Win32/Skeeyah.HK!MTB を画像化した図である．同一のマルウェア名を持つマルウェア群であっても，図 1 のペアのように似たに画像になる検体もあれば，図 2 のペアのように人間の目には別種のマルウェアのように見える画像になる検体もある．この問題に対処するために Hsiao ら [6] は画像化した後に，Average Hash [8] を使って同一ファミリ内の画像データセットを，似た見た目の画像が集まったグループにさらに分類しているが，我々の研究ではそれは行わない．Average Hash による再分類はおそらく実験環境での精度の向上につながるが，実用時も同じマルウェア名かつ同じ見た目のマルウェアを訓練データとして用意しなければならなくなるおそれがあり，実用をより困難にするためである．Average Hash による再分類を行わずに高い精度を出せるならば，実用時は画像化時の見た目を意識せずとも同じマルウェア名のマルウェアを訓練データとして用意すれば十分となる．

3.2 実験設計

欠損マルウェア同定のための各機械学習モデルの訓練時は，無欠損マルウェアに対して最低 1 クラスタ，最大 10 クラスタ部分の欠損処理を施すことで生成した欠損マルウェア群を使用する．このとき欠損クラスタ数およびその開始位置は毎回重複を許容しランダムに決定し，動的に欠損マルウェアを生成して訓練に用いる．

3.1.3 項で述べた手法により画像化した検体群を使い，以下の 2 つのシナリオに沿って TN および CNN を訓練する．

シナリオ 1：訓練および検証時に同一の無欠損マルウェア群を使用して欠損マルウェアをランダム生成し，ニューラルネットワークへの入力とする．本シナリオで欠損マルウェアから元のマルウェア名をよく同定できるということは，ある無欠損マルウェアを訓練データとして保持している場合，そのマルウェアがデータ欠損を起こしても元のマルウェア名をよく同定できるということを表す．

シナリオ 2：各マルウェア名に所属する 5 つの無欠損マルウェアを，訓練時と検証時でそれぞれ 3:2 の割合で分割し，それらから欠損マルウェアをランダム生成し，ニューラルネットワークへの入力とする．本シナリオで欠損マルウェアから元のマルウェア名をよく同定できるということは，あるマルウェア名に属する無欠損マルウェアを訓練データとして保持している場合，そのマルウェア名を持つなんらかのマルウェアがデータ欠損を起こしても元のマルウェア名を同定できるということを表す．

2 つのシナリオともに，検証時には，検証用の無欠損マルウェア群に対して，1 検体ごとに 0 から 9，10，20，30，40，50 箇所のクラスタをランダムに欠損させてから画像化を行うという作業を 10 回繰り返し，検証用の欠損マルウェア画像を生成する．つまり，1 検体あたり欠損箇所数ごとに 10 枚の欠損マルウェア画像が得られる．欠損箇所は 1 つの欠損マルウェアの中で重複が起こらないようにランダムに決定する．元検体のファイルサイズが，欠損クラスタ数 $\times 4,096$ bytes 以下である場合，全クラスタが欠損する．このようにして生成した検証データセット A を訓練したモデルに入力し，171 クラス分類問題をどれだけの精度で解くことができるか検証する．

また，著者らの過去の研究 [3] で，マルウェアの先頭クラスタがアンチウイルスによる同定に致命的な悪影響を及ぼすことがわかっている．このような先頭クラスタ欠損に対する耐性を確かめるために，検証用の無欠損マルウェア群の先頭クラスタを欠損させ，かつ前述のランダムなクラスタ欠損処理を施した検証データセット B も生成した．検証データセット B も同様にモデルへ入力し 171 クラス分類問題を解く．

3.3 マルウェア同定機械学習モデル

3.2 節で定義したシナリオに従い，以下のように Triplet Network および Convolutional Neural Network の訓練を行う．

3.3.1 Triplet Network

Triplet Network は 3 つ組のサンプルを入力組とするニューラルネットワークであり，サンプル間の関係性を

学習することに用いられる．3 つ組みの入力は anchor，positive，negative サンプルで構成される．anchor は訓練データの任意のサンプルである．positive/negative サンプルはそれぞれ，anchor サンプルと同/異クラスに属するサンプルである．訓練時には，この 3 入力をそれぞれベースネットワークに入力して得られる出力に対し，anchor サンプル-positive サンプル間の距離を近く，anchor サンプル-negative サンプル間の距離を小さくするようネットワークを学習する．これにより，サンプル間の関係性を適切に表現する特徴量を出力するニューラルネットワークが構成される．

図 3 に Triplet Network の構造を示す．anchor サンプル，positive サンプル，negative サンプルがそれぞれベースネットワークを通りベクトル化され，Triplet Loss 損失関数により損失が計算される．3 つのベースネットワークは重みを共有している．**図 4** にベースネットワークの構造を示す．この構造の前半部分は [6] で使われているものと同一である．各二次元畳み込み層（Conv2D）の活性化関数には Rectified Linear Unit（relu）を使用した．末尾の Lambda では L2 正規化を行っている．オプティマイザには Adam（学習率 0.001）を利用した．anchor サンプルとして欠損マルウェア画像を，positive サンプルとして anchor サンプルと同一のマルウェア名を持つ無欠損マルウェア画像を，negative サンプルとして anchor サンプルと異なるマルウェア名を持つ無欠損マルウェア画像を与える．より詳細には，訓練データセットとして，各マルウェア名ごとに 30 パターンの anchor サンプル-positive サンプルのペアを，1 パターンの anchor サンプル-posirive サンプルのペアごとに 5 つの negative サンプルを毎回ランダム生成した．このように，エポックごとに毎回 anchor サンプル-positive サンプル-negative サンプルの三つ組みをランダム生成し，Triplet Network の訓練に使用する．以上の条件で，Triplet Network の訓練を 100 エポック実施した．

Triplet Network の訓練終了後，訓練済みのベースネットワークに入力を伝播して得られるベクトルを特徴量とし，クラス分類用のモデルを訓練する．クラス分類用のモデルとして今回は，中間層 1 層（ユニット数 128）の 3 層ニューラルネットワークを利用した．

3.3.2 Convolutional Neural Network

Convolutional Neural Network とは，画像解析分野でよく使われているニューラルネットワークであり，二次元 CNN の場合は画像の X 軸・Y 軸方向にフィルタをスライドして畳み込み結果を得る．

今回使用した CNN の構造は，Triplet Network のベースネットワークである図 4 の構造とほぼ同一である．図 4 の末尾の Lambda の後ろに，マルウェア名のクラス数と同数である 171 ユニットの全結合層を追加し，活性化関数として softmax を使用した．オプティマイザは Triplet Network と同様に Adam（学習率 0.001）とした．

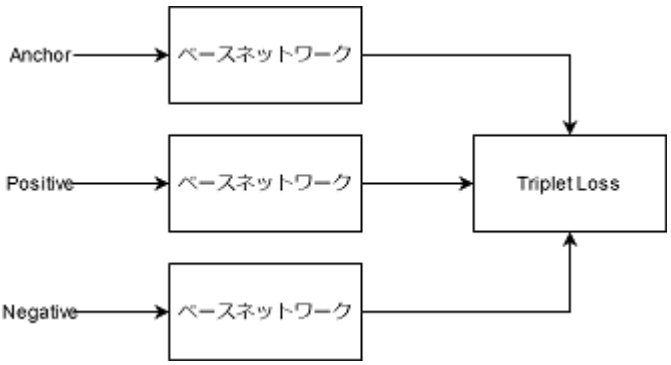

図 3 Triplet Network の構造

各マルウェア名ごとに 5 つ（シナリオ 1）または 3 つ（シナリオ 2）の無欠損検体画像，1 つの無欠損検体ごとに 5 パターンの欠損マルウェア画像をランダム生成し，訓練データセットとした．すなわち，1 マルウェア名ごとに 30 パターン（シナリオ 1）または 18 パターン（シナリオ 2）のマルウェア画像が生成される．Triplet Network での実験と同様に，エポックごとにこれら訓練データセットはランダムに再生成され，順次 CNN へ入力される．以上の条件で，訓練を 100 エポック実施した．

4. 結果

図 5 に，訓練後の Triplet Network のベースネットワークに無欠損マルウェア画像 855 枚 171 クラスを入力して得られる特徴量を，t-SNE [12] を用いて 2 次元まで次元削減し可視化した図を示す．マルウェア名が同一の検体に対して同じ色を割り当てて図示している．図中において距離が近いほど，特徴量の傾向が近いことを表している．同じマルウェア名を持つ検体の特徴量間の距離は近く，異なるマルウェア名を持つ検体の特徴量間の距離は遠くなっており，訓練済みの Triplet Network により得られる特徴量は，それを入力として受け取る分類器の精度向上に寄与していることがわかる．

図 6 および**表 2** 前半に，検証データセット A に対して TN と CNN を用いた，欠損マルウェア同定の結果を示す．同定精度とは，欠損マルウェアが与えられたとき，そのマルウェアが欠損する前のマルウェア名を 171 種のマルウェア名の中から当てることのできた割合である．図表中の TN および CNN は検証に使用したモデルを表し，それぞれ Triplet Network と Convolutional Neural Network である．末尾に_sep がついていない場合はシナリオ 1，すなわち訓練時と検証で同じ無欠損マルウェア群から欠損マルウェア群をランダム生成したことを表し，_sep がついている場合はシナリオ 2，すなわち訓練時と検証でそれぞれ別の無欠損マルウェア群から欠損マルウェア群をランダム生成したことを表す．

表 2 同定精度の詳細

全箇所をランダムに欠損させた場合の同定精度（検証データセット A を使用）

	欠損クラスタ数														
	0	1	2	3	4	5	6	7	8	9	10	20	30	40	50
TN	0.974	0.973	0.964	0.958	0.954	0.935	0.926	0.918	0.908	0.881	0.867	0.716	0.590	0.481	0.394
CNN	0.973	0.973	0.969	0.962	0.958	0.939	0.934	0.928	0.921	0.907	0.890	0.787	0.693	0.614	0.542
TN_sep	0.485	0.480	0.479	0.473	0.467	0.456	0.452	0.449	0.445	0.428	0.423	0.333	0.273	0.214	0.178
CNN_sep	0.456	0.454	0.453	0.457	0.453	0.440	0.442	0.443	0.439	0.433	0.419	0.354	0.316	0.280	0.236

先頭クラスタを必ず欠損させた場合の同定精度（検証データセット B を使用）

	先頭クラスタ以外の欠損クラスタ数														
	0	1	2	3	4	5	6	7	8	9	10	20	30	40	50
TN	0.972	0.969	0.963	0.959	0.951	0.933	0.927	0.916	0.903	0.885	0.869	0.724	0.587	0.478	0.396
CNN	0.972	0.970	0.967	0.961	0.953	0.939	0.932	0.927	0.920	0.905	0.890	0.786	0.695	0.616	0.540
TN_sep	0.491	0.480	0.476	0.470	0.462	0.455	0.453	0.449	0.444	0.435	0.415	0.334	0.269	0.213	0.177
CNN_sep	0.453	0.453	0.453	0.453	0.453	0.442	0.439	0.439	0.439	0.430	0.419	0.354	0.317	0.273	0.236

シナリオ 1 の場合，欠損クラスタ数が 8 クラスタ以下であるならば，TN，CNN ともに 90%以上の精度でマルウェア名の同定ができている．シナリオ 2 の場合，欠損クラスタ数が 10 クラスタ以下であるならば TN，CNN ともに 41%から 48%程度の精度でマルウェア名の同定ができている．欠損箇所が少ないうちは TN と CNN で同定精度にあまり差は無いが，欠損箇所が増えるにつれ CNN の同定精度が勝る．

図 7 および表 2 後半に，検証データセット B に対して TN と CNN を用いた，欠損マルウェア同定の結果を示す．先頭クラスタの欠損はアンチウイルスによる同定には致命的な悪影響を与え同定成功率をほぼ 0%まで押し下げたが，機械学習を用いた本手法においてはさほど重要ではなく，検証データセット A とほぼ同様の同定精度となった．データ分割を行わない場合は，先頭クラスタが欠損していたとしても，機械学習による同定により 97.2%の精度でマルウェア名の同定に成功している．データ分割を行った場合でも，TN により 49.1%の精度でマルウェア名の同定に成功している．

5. 考察

結果から，TN，CNN ともに欠損マルウェアを高精度に同定できており，アンチウイルスを使用した同定 [3] よりも欠損の影響を低く抑えられていることがわかる．シナリオ 2 の条件であっても，10 クラスタ程度の欠損までであれば 171 クラス分類問題を約 42%超の精度で解けており，これはあるマルウェア名を持つ無欠損マルウェアを訓練データとして保持していれば，同名かつハッシュが異なるマルウェアが 10 クラスタ欠損した場合でも元のマルウェア名を 42%前後の精度で推定できる可能性を示す．

シナリオ 1 およびシナリオ 2 ともに，データ欠損箇所が増えるにつれて同定精度が低下していることが見て取れる．欠損クラスタ数が増えると同定に利用できる情報が減るため精度が低下し，さらに検体サイズによっては情報が完全に消えるためであると考えられる．**図 8** に無欠損検体のクラスタ数を示す．サイズが 50 クラスタ未満である検体も多く全体の約 3 割を占め，今回の実験条件ではデータ欠損によりこれら検体の情報は全損しうる．サイズの小さい検体は少量の上書きでデータの全損が起こりやすいため，欠損が起きた場合同定が難しい検体であるといえる．10，20，30，40，50 クラスタの欠損が生じると，全検体はそれぞれ平均して 76.9%，64.2%，55.2%，48.0%，41.8%のデータが残存する．

機械学習による同定手法が欠損に対してある程度の耐性を得た原因を探るために，検体のどの部分に着目してクラス推定が行われたかを調査した．Grad-CAM [13] を用いることで，訓練済みモデルに対して画像を入力しクラス推定を行った際，推定に強く寄与した画像の領域をハイライトすることができる．シナリオ 1 の条件で訓練を行った CNN に対して，Grad-CAM を用いて推定根拠の可視化を行った．結果の一例として，無欠損マルウェア検体 Trojan:Win32/Dorv.A をシナリオ 1 の CNN を用いてクラス推定を行った際の推定根拠を**図 9** に示す．主にファイル先頭の PE ヘッダ領域，.text セクション，.rsrc セクションを推定根拠としている．推定根拠をファイルの一部分に依存しすぎず，ファイル全体の各所を手がかりとしているため，局所的な欠損に耐性があると思われる．また，検体を画像化する際にゼロパディングが行われた部分も，推定根拠として使用されていることがわかる．ゼロパディング領域の形状は検体のサイズに依存するため，実質的に検体サイズを推定根拠として使用していると考えられる．検体サイズは攻撃者やマルウェア自身によって容易に変更可能であるため，この性質は好ましくない．

今回の実験では，多くの実験条件において CNN の精度が TN の精度を上回った．TN は訓練データの各クラスに属するサンプル数が少なくても各クラスを精度良く分離で

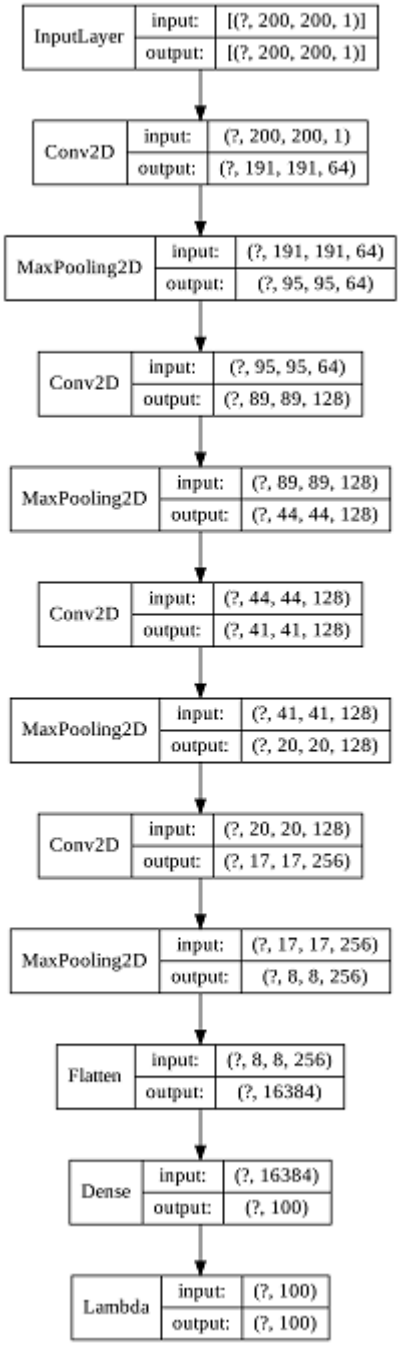

図 4 ベースネットワークの構造
カッコ内の数字は，順にバッチサイズ（常に任意長），マップ高さ，マップ幅，チャネル数を表す．

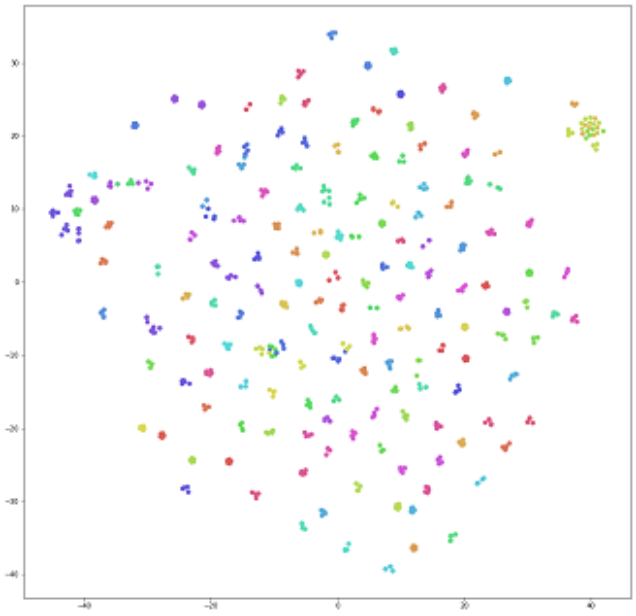

図 5 特徴量の分布

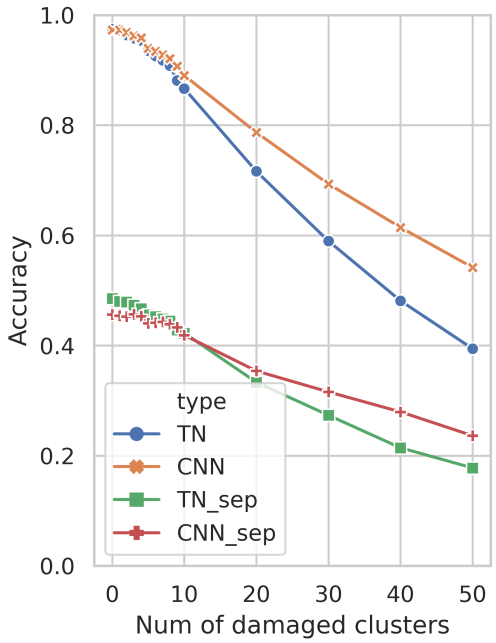

図 6 全箇所をランダムに欠損させた場合の同定精度

きる手法として知られている．訓練データにおける無欠損マルウェアの各クラスあたりのサンプル数はシナリオ 1 で 5 検体，シナリオ 2 で 3 検体と少数であったが，ランダム箇所欠損により各クラスに属する欠損マルウェアサンプル数は膨大になるため，TN の優位性が活きなかったと考えられる．

6. まとめと今後の課題

本論文では，ファイルが消去されたことによりデータが欠損したマルウェアに着目し，機械学習を用いてこのような欠損マルウェアから元のマルウェア名を同定する手法について実験を行った．10 クラスタ程度の欠損であれば 95%前後（シナリオ 1：無欠損時に同ハッシュである検体を訓練データに含めた場合），もしくは 45%前後（シナリオ 2：無欠損時に同名である検体を訓練データに含めた場合）の精度で元のマルウェア名を同定できることがわかった．先頭クラスタの欠損はアンチウイルスによるマルウェア名同定の精度をほぼ 0%まで押し下げるが，機械学習による同定手法の場合はほぼ影響は無かった．

今後の課題として，データセットにパック済みマルウェアを含めることが挙げられる．パックはバイナリを画像化

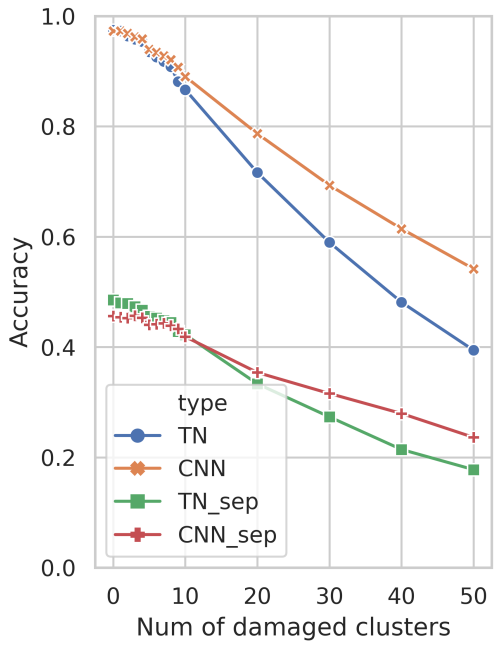

図 7 先頭クラスタを必ず欠損させた場合の同定精度

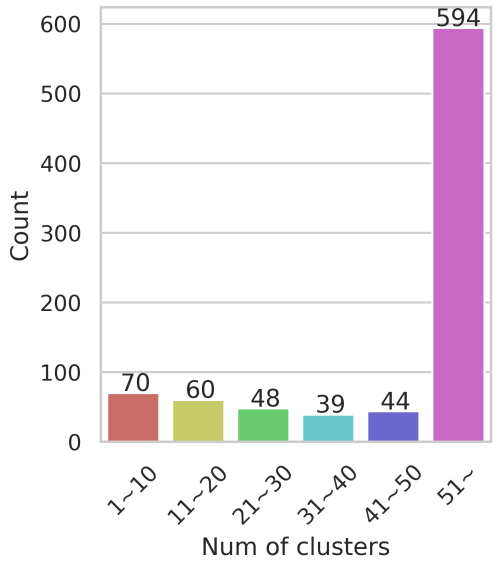

図 8 無欠損検体のクラスタ数

した時の見た目を著しく変えるため，シナリオ 2 の分類精度を著しく押し下げると思われる．

参考文献

[1] Hand, S., Lin, Z., Gu, G. and Thuraisingham, B.: BinCarver: Automatic Recovery of Binary Executable Files, *Proceedings of the 12th Annual Digital Forensics Research Conference (DFRWS'12)* (2012).

[2] 林　健，佐々木良一：時間経過に着目した HDD のデータ復元に関する実験と解析，情報処理学会研究報告，Vol. 2013-CSEC-60, No. 14 (2013).

[3] 小久保博崇, 大山恵弘：マルウェア検体のデータ欠損がマルウェア同定に与える影響の調査, コンピュータセキュリティシンポジウム 2019 論文集, No. 2019, pp. 947–952 (オンライン)，入手先 ⟨https://ci.nii.ac.jp/naid/170000181104/⟩ (2019).

[4] Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., Chen, B. and Wu, Y.: Learning Fine-Grained Image Similarity with Deep Ranking, *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1386–1393 (2014).

[5] Nataraj, L., Karthikeyan, S., GrégoireJacob, Manjunath, B. : Malware Images: Visualization and Automatic Classification, (online), DOI: 10.1145/2016904.2016908 (2011).

[6] Hsiao, S.-C., Kao, D.-Y., Liu, Z.-Y. and Tso, R.: Malware Image Classification Using One-Shot Learning with Siamese Networks, *Procedia Computer Science*, Vol. 159, pp. 1863–1871 (online), DOI: 10.1016/j.procs.2019.09.358 (2019).

[7] Yakura, H., Shinozaki, S., Nishimura, R., Oyama, Y. and Sakuma, J.: Neural malware analysis with attention mechanism, *"Computers & Security"*, Vol. 87, p. 101592 (2019).

[8] Imagehash: Imagehash, `https://github.com/bjlittle/imagehash/`.

[9] ChronicleSecurity: VirusTotal, `https://www.virustotal.com/`.

[10] Microsoft: Malware names, `https://docs.microsoft.com/en-us/windows/security/threat-protection/intelligence/malware-naming`.

[11] Cylance: PyPackerDetect, `https://github.com/cylance/PyPackerDetect`.

[12] van der Maaten, L. and Hinton, G.: Visualizing Data using t-SNE, *Journal of Machine Learning Research*, Vol. 9, pp. 2579–2605 (online), available from ⟨http://www.jmlr.org/papers/v9/vandermaaten08a.html⟩ (2008).

[13] Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D. and Batra, D.: Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization, *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 618–626 (2017).

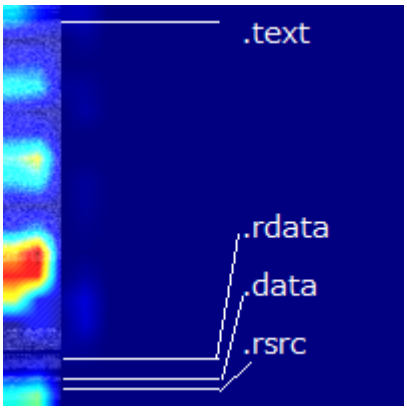

図 9 推定根拠の可視化の一例　(Trojan:Win32/Dorv.A)
モデルが推定時に強く注視した箇所ほど赤く示す．