「目 grep」は機械学習で実現可能か？

三村 守[1,2]　大坪 雄平[3,2]　田中 秀磨[1]　後藤 厚宏[2]

概要：情報セキュリティ分野における人材は慢性的に不足しており，特に非常に高度なスキルを有する達人あるいはホワイトハッカーが不足している．彼らは「目 grep」のような論理的に説明できないスキルを有しており，そのスキルの自動化は困難であると考えられている．「目 grep」とは，バイナリファイルから人間の目で文字列を検索する GREP コマンドをエミュレートするスキルである．本稿では，畳み込みニューラルネットワークを用いて「目 grep」を再現し，未知の悪性文書ファイルを検知するいくつかの手法を提案する．畳み込みニューラルネットワークは，画像認識の分野において革新的であり，従来のモデルよりも顕著な成果を挙げている．さらに，実際の悪性文書ファイルからデータセットを作成し，Precision，Recall および F 値を算出して提案手法を評価した．その結果，悪性文書ファイルから「目 grep」によってシェルコードを発見できる可能性があることを確認した．

キーワード：マルウェア，目 grep，画像処理，機械学習，畳み込みニューラルネットワーク

Is Emulating “Binary Grep in Eyes” Possible with Machine Learning?

Mamoru Mimura[1,2]　Yuhei Otsubo[3,2]　Hidema Tanaka[1]　Atsuhiro Goto[2]

Abstract: Talented people who work in the fields of cybersecurity are greatly lacking. Especially, there are insufficient experts or white hackers. They have incredible skills such as “binary grep in eyes” which nobody cannot explain logically. In general, it is difficult to automate their skills. “Binary grep in eyes” is a skill to emulate executing GREP command in binary files with human eyes. This paper proposes some methods to emulate “binary grep in eyes” to detect unseen malicious document files with Convolutional Neural Network (CNN). CNN is commonly linked with innovations in the fields of image recognition and achieves superior results over several prior existing models. Then this paper created the dataset from actual malicious document files in the wild, and calculated the Precision, the Recall and the F-measure to evaluate our method. As the result, there is a possibility that our method can emulate “binary grep in eyes” and detect the shellcode in unseen malicious document files.

Keywords: Malware, Binary Grep in Eyes, Image Processing, Machine Learning, Convolutional Neural Network

1. はじめに

インターネット技術の爆発的な普及に伴い，サイバー攻撃の脅威が増大している．サイバー攻撃やその攻撃に利用される脆弱性に関する情報は毎日のように報告され，多くの組織においてセキュリティに携わる人材が必要とされるようになった．しかしながら，セキュリティの知識を有する人材は不足しており，人材育成が急務と言われている．経済産業省が 2016 年に実施した調査 [1] によると，現在のセキュリティ人材は約 13 万人不足しているとされており，人材不足はさらに拡大することが予想されている．慢性的なセキュリティ人材の不足は，今後さらに増大するサイバー攻撃の脅威をより深刻な事態に発展させる可能性がある．とりわけ，非常に高度なスキルを有する達人あるい

[1] 防衛大学校
National Defense Academy
[2] 情報セキュリティ大学院大学
Institute of Information Security
[3] 警察庁
National Police Agency

はホワイトハッカーは不足している．人材不足を解消する手段としては，機械学習を中心とする人工知能を活用し，業務を自動化することが挙げられる．機械学習はサイバーセキュリティの分野においても積極的に活用されており，未知のマルウェアや攻撃通信の検知などにおいて成果をあげている．しかしながら，達人やホワイトハッカーは論理的に説明できないスキルを有しており，そのスキルの自動化は困難であると考えられている．

論理的に説明できない常人を逸脱したスキルの一つとして，「目 grep」が挙げられる．「目 grep」は，人間の目で文字列を検索する GREP コマンドをエミュレートするスキルである．通常はバイナリエディタに搭載されているビットマップビューと呼ばれるバイナリを視覚化する機能を活用し，鍛えられた眼力や研ぎ澄まされた野生の勘によって目標を発見する．「目 grep」は，鍛錬の度合いによって個人差が大きく生じるスキルであり，未知のマルウェアのシェルコードや脆弱性の発見にも役立っている．したがって，「目 grep」を自動化することができれば，達人あるいはホワイトハッカーのスキルを代用できる可能性につながるものと考えられる．

本稿では，機械学習により「目 grep」を自動化できる可能性について検討する．具体的には，「目 grep」による悪性文書ファイルからのシェルコードの発見を題材とする．「目 grep」において，達人やホワイトハッカーがバイナリを視覚的にとらえている点や鍛錬の度合いにより個人差が大きく生じる点に着目し，画像認識において顕著な成果を挙げている畳み込みニューラルネットワークの活用を試みる．

以下，第 2 節ではニューラルネットワークを用いた関連研究と本研究の違いを示す．第 3 節では，「目 grep」の特徴について説明する．第 4 節では，畳み込みニューラルネットワークを用いて「目 grep」を再現し，悪性文書ファイルを検知するいくつかの手法を提案する．第 5 節では提案手法を実際の悪性文書ファイルに適用し，交差検証により精度を確認する．第 6 節では実験結果を考察し，最後にまとめと今後の課題を示す．

2. 関連研究

本研究では，機械学習の一分野であるニューラルネットワークを，マルウェアの検体に適用して悪性か良性かの分類を実施する．ニューラルネットワークをマルウェアに適用した例としては，以下に示す関連研究が挙げられる．

文献 [2] では，実行ファイルのバイナリデータおよび ASCII 文字列から作成したヒストグラム，PE ヘッダから抽出した情報等から，ニューラルネットワークを用いてマルウェアを分類する手法を提案している．この手法では，実行ファイルを分類の対象としており，分類のために 4 層のニューラルネットワークを用いている．本研究では，文書ファイルタイプのマルウェアを対象としており，検体そのものの分類や検知よりも，むしろ内部のシェルコードの検知に焦点をあてている．また，バイナリファイルから作成したヒストグラムではなく，直接バイナリファイルを視覚化したビットマップイメージを用い，画像認識に優れる畳み込みニューラルネットワークを用いている．

文献 [3] では，Windows および Linux のバイナリファイルから，RNN (Recurrent Neural Networks) を用いて関数を識別する手法を提案している．RNN は，文脈を考慮したニューラルネットワークのモデルの 1 つであり，時系列のデータを取り扱うことが可能である．この手法では，直接的にバイナリファイルを取り扱っており，実行ファイルの各オフセットの値の文脈に着目している．また，RNN を適用する対象は Windows および Linux の実行ファイルである．本研究では，文書ファイルタイプのマルウェアを対象としている．また，バイナリファイルを直接的に取り扱っている点は共通しているが，対象の文脈に焦点をあてている点が異なっている．本研究では，対象の文脈には着目せず，視覚的にバイナリファイルを識別するアプローチを試みる．

文献 [4] では，RNN を用いてマルウェアの実行結果を集約し，多層パーセプトロンおよびロジステック回帰モデルを用いて分類する手法を提案している．この手法では，マルウェアの実行結果を得るために動的解析を必要とする．本研究では，学習の対象はマルウェアの実行結果ではなく，バイナリファイルそのものであるため，動的解析を必要としない．また，分類のために畳み込みニューラルネットワークを用いている．

文献 [5] では，マルウェアの実行ファイルの Opcode を視覚化し，3 層の畳み込みニューラルネットワークを用いて亜種を検知する手法を提案している．本研究とは，マルウェアを視覚化して畳み込みニューラルネットワークを利用する点は共通しているが，その視覚化の手法は異なっている．また，本研究ではマルウェアの実行ファイルではなく，文書ファイルタイプのマルウェアを対象としている点も異なっている．

文献 [6] では，RNN を用いてマルウェアの実行結果から得られる特徴を集約し，畳み込みニューラルネットワークを用いて分類する手法を提案している．本研究とは，畳み込みニューラルネットワークを利用する点は共通しているが，その分類の対象が異なっている．この手法では，マルウェアの実行結果を得るために動的解析を必要とする．本研究では，学習の対象はマルウェアの実行結果ではなく，バイナリファイルそのものであるため，動的解析を必要としない．

文献 [7] では，Auto Encoder モデルを用いて実行ファイルから抽出した API コールを学習し，マルウェアを検知する手法を提案している．この研究では，マルウェアの検

知に教師なし学習モデルである Auto Encoder モデルを用いている．本研究では，分類のために畳み込みニューラルネットワークを用いている．また，学習の対象は API コールではなく，バイナリファイルそのものとなっている．

文献 [8] では，実行ファイルから抽出した Opcode の N-gram から特徴ベクトルを作成し，Deep Belief Network (DBN) を用いて分類する手法を提案している．この手法では，DBM を Auto Encoder モデルとして用いている．本研究では，分類のために畳み込みニューラルネットワークを用いている．また，学習の対象は Opcode ではなく，バイナリファイルそのものとなっている．

文献 [9] では，Recursive Neural Network を用いてマルウェアの時系列の通信内容を分類し，動的解析を継続するか否かを判断する手法を提案している．この手法では，分類に Recursive Neural Network を用いており，その対象はマルウェアの通信内容である．本研究では，分類のために畳み込みニューラルネットワークを用いている．また，学習の対象はマルウェアの通信内容ではなく，バイナリファイルそのものとなっている．

3. 「目 grep」の特徴

3.1 「目 grep」の概要

「目 grep」とは，人間の目で文字列を検索する GREP コマンドをエミュレートするスキルである．GREP は UNIX 系の OS におけるコマンドであり，ファイル内から正規表現等に合致した行を検索して出力する．「目 grep」はもともとプログラマの間で使用されていた俗語であり，普段とは異なる環境の現場でのデバッグにおいて，この GREP コマンドをプログラマ自身がエミュレートする知る人ぞ知る高等テクニックであった．「目 grep」は柔軟な非正規表現を用いた検索が実行できるとされているが，稀にマッチングに失敗する可能性も指摘されている．このように，「目 grep」は主にプログラムのデバッグの用途に用いられてきた．

2010 年頃になると，「目 grep」はバイナリファイルのリバースエンジニアリングやデジタルフォレンジックの用途で用いられるようになってきた．これらの用途では，解析対象とするマルウェア等の未知のアーティファクトをバイナリエディタで開き，解析者自身が目でバイナリを調査することがある．通常はビットマップビューと呼ばれるバイナリを視覚化する機能を活用し，経験によって鍛えられた眼力や研ぎ澄まされた野生の勘を最大限に活用して目標を発見する．ここで言う目標とは，マルウェアのエクスプロイトコード，シェルコード，あるいは特定のコード等，その目的によって様々である．「目 grep」は，熟練度によって個人差が大きく生じるスキルであり，未知のマルウェアのシェルコードや脆弱性の発見にも役立っている．特に，「目 grep」を得意とする達人はバイナリアンと呼ばれており，リバースエンジニアリングやデジタルフォレンジックでは重宝されている．

本稿では，「目 grep」をリバースエンジニアリングやデジタルフォレンジックの用途において，バイナリファイルを視覚的に調査するスキルと解釈する．

3.2 「目 grep」の手法

「目 grep」の手法は，バイナリファイルの中からマジックナンバーあるいはシグネチャを探すのが基本である．マジックナンバーはそのファイルの種類を示すコードであり，ファイルの先頭に記録されていることが多い．たとえば，Windows の EXE の拡張子を持つ実行ファイルであれば，MZ ヘッダの 0x5A4D や PE ヘッダの 0x00004550 がシグネチャであり，これらのシグネチャからそのファイルの種類を判断することができる．このような固定のシグネチャであれば，機械学習を用いなくとも自動的に検知することは容易である．

より実践的な「目 grep」の手法としては，ビットマップビューでバイナリファイル全体を視覚的に概観する手法が挙げられる．バイナリファイル全体は，10 進数の 0〜255 の値の連続で表現することが可能である．したがって，その各値を画素の値と解釈すれば，ビットマップイメージとしてバイナリファイル全体を視覚化することが可能である．**図 1** にビットマップビュー機能の使用例を示す．

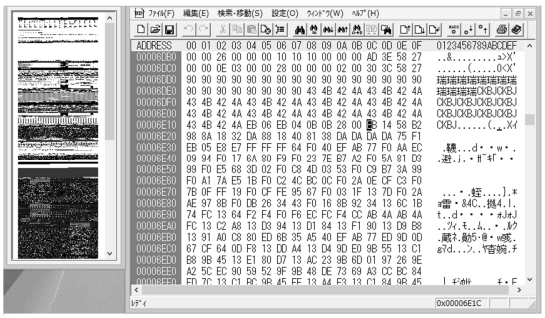

図 1 ビットマップビュー機能の使用例
Fig. 1 A usage example for a Bitmap View function.

図中の左側は視覚化したビットマップイメージであり，右側はその一部分の値を 16 進数および ASCII 文字で表示したものである．この手法では，全体の構成，エントロピー，視覚的な模様等から，ファイルの形式を判断したり，不審な部分を抽出したりする．たとえば，JPEG 等の圧縮されたファイルであればエントロピーが高くなるため，ぐちゃぐちゃに見える傾向がある．さらに，抽出した不審な部分については直接バイナリの確認を実施する．この手法では，使用頻度が高い文字コード，繰り返しのパターン，何らかの規則性等に基づいて調査する．たとえば，UTF-16 の ASCII 文字の場合には，一文字毎に 0x00 が入るため，

0x00 がの使用頻度が高いという特徴がある．このような特定の文字コードが多いという特徴は，ビットマップビューにおいても特定の色で表現されることがある．実践的な「目 grep」の判断基準は分析者によって様々であり，曖昧である点に特徴がある．これらの実践的な手法は，シグネチャを用いることができない場合にも有効である．たとえば，未知のマルウェアやアーティファクトにおいては，シグネチャがそのまま残されているとは限らず，ヘッダが偽造されている場合もある．また，デジタルフォレンジックにおいては，一部のデータが欠如している場合も珍しくない．そのため，視覚的な模様等を認識する能力や曖昧な判断基準が有益となる．

バイナリファイルを視覚的に認識する手法としては，画像処理技術の活用が考えられる．特に，畳み込みニューラルネットワーク (CNN) は画像認識において目覚しい成果をあげている．また，回数を重ねるごとに精度が向上する特徴は，熟練度によって個人差が大きく生じるという特徴に合致している．そこで本稿では，未知のマルウェアやアーティファクトの解析を想定し，人間が得意とする視覚的な模様等を認識する能力や曖昧な判断基準を，CNN を用いて再現する手法を検討する．

3.3 悪性文書ファイルにおける「目 grep」の必要性

本稿では，マルウェアの一種である悪性文書ファイルを対象として，「目 grep」を自動化できる可能性を検討する．悪性文書ファイルとは，Microsoft Office あるいは PDF (Portable Document Format) 形式のマルウェアであり，標的型攻撃においてメールで送付されることが多い．悪性文書ファイルは，脆弱性を利用するタイプと，脆弱性を利用しないタイプに分類される．脆弱性を使用しないタイプの悪性文書ファイルとしては，悪性のマクロや不正サイトへのリンクを含むものが挙げられる．本稿では，脆弱性を使用するタイプの悪性文書ファイルを想定している．一般的な脆弱性を使用するタイプの悪性文書ファイルの構造を図 1 に示す．

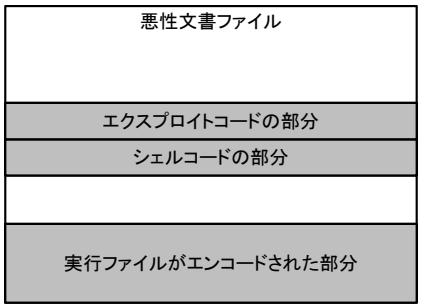

図 2 悪性文書ファイルの構造
Fig. 2 Structure of a malicious document file.

表 1 入力画像の作成手法
Table 1 The methods to generate input image data of a fixed size.

手法		概要
1	全体を縮小する手法	ファイル全体のビットマップイメージを縮小して固定サイズにする．
2	終端部分を抽出する手法	ファイルの終端部分を抽出し，固定サイズのビットマップイメージを作成する．
3	シェルコード部分を抽出する手法	シェルコードの部分を抽出し，固定サイズのビットマップイメージを作成する．
4	エンコード部分を抽出する手法	実行ファイルがエンコードされた部分を抽出し，固定サイズのビットマップイメージを作成する．

悪性文書ファイルには，脆弱性を突くエクスプロイトコード，実行ファイルを展開するシェルコードが含まれており，実行ファイルがエンコードされて埋め込まれている場合もある．この実行ファイルが埋め込まれているタイプのマルウェアは，ドロッパーと呼ばれている．実行ファイルが埋め込まれておらず，インターネットからダウンロードするタイプのマルウェアは，ダウンローダと呼ばれている．未知の悪性文書ファイルを解析する場合には，動的解析や静的解析などの様々な手法が駆使される．しかしながら，高度なマルウェアには対解析技術が施されており，動的解析や静的解析も困難である場合も珍しくない．また，エクスプロイトが動作する環境が判明せず，動的解析に失敗する場合もある．このような場合，バイナリエディタ等でエクスプロイトコード，シェルコード，実行ファイル等の不審な部分の有無を確認し，そのファイルが悪性であるか良性であるかを判断することになる．

4. 提案手法

4.1 入力画像の作成手法

畳み込みニューラルネットワーク (CNN) を活用するためには，その入力として固定のサイズの入力画像を準備する必要がある．「目 grep」の特徴や悪性文書ファイルの構造を考慮し，悪性文書ファイルから入力画像を作成する手法を 4 つ考案した．入力画像の作成手法を**表 1** に示す．

手法 1 は，バイナリファイル全体からビットマップイメージを作成し，その画像を固定のサイズに縮小する手法である．この手法は，達人がバイナリファイル全体のビットマップイメージを概観するスキルを想定している．手法 2 は，単純にバイナリファイルの終端から固定サイズを抽出し，そのビットマップイメージを作成する手法である．この手法は，達人のスキルを想定していない．他の手法との比較検討のために用いる．手法 3 は，シェルコードの部分から固定サイズを抽出し，そのビットマップイメージを作成する手法である．この手法は，達人がバイナリファイルの 16 進数の値を眺め，通常の文書ファイルには含まれ

ていない不審な部分を発見するスキルを想定している．手法 4 は，実行ファイルがエンコードされた部分から固定サイズを抽出し，そのビットマップイメージを作成する手法である．この手法も手法 3 と同様に，達人がバイナリファイルの 16 進数の値を眺め，通常の文書ファイルには含まれていない不審な部分を発見するスキルを想定している．

4.2 畳み込みニューラルネットワーク (CNN) の構成

手法 1〜4 で作成した固定サイズの入力画像を，畳み込みニューラルネットワーク (CNN) に入力して画像認識を試みる．今回の実験で使用する CNN の構造を**図 3** に示す．

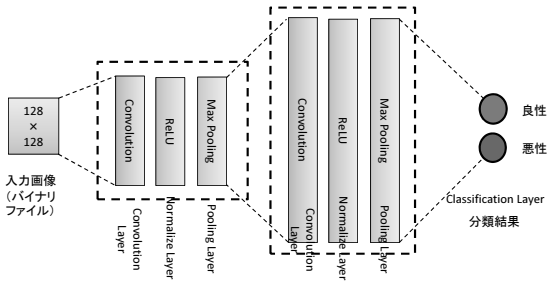

図 3 畳み込みニューラルネットワークの構造
Fig. 3 The structure of the convolutional neural networks.

この CNN は 128 × 128 のバイナリファイルから作成した画像を入力とし，２層の畳み込み層，活性化関数およびプーリング層を経由し，全結合層を経て２値のラベルを出力する．畳み込み層におけるフィルタのサイズは５×５とし，第１層の出力サイズを 32，第２層の出力サイズを 64 とした．１〜２層の活性化関数は ReLU (Rectified Linear Unit), プーリング層でのダウンサイジングには Max Pooling を用いた．損失関数には以下の交差エントロピー誤差を用いた．

$$E = -\sum_{n}^{N}\sum_{i}^{D} t_{ni} \log y_i$$

式中の N はデータの数，D は出力数のユニット数，y は出力された値，t はラベルの値を示す．ミニバッチ勾配降下法のバッチサイズは 100 とし，最適化アルゴリズムは Adam (Adaptive Moment Estimation) を用いた．

提案手法で入力画像を作成し，CNN に入力して訓練および識別を実施する試験プログラムを Python-2.7 を用いて実装した．CNN の実装には，Chainer-1.23[10]，CUDA 8.0 および cuDNN-8.0 を用いた．

5. 検証実験

5.1 データセットの作成

作成した試験プログラムを用い，悪性文書ファイルと良性文書ファイルを区別することができるか検証する．検証実験に使用する検体の概要を**表 2** に示す．

表 2 検体の概要
Table 2 The outline of the specimen.

種類	拡張子	悪性	良性
CFB	doc	63	63
	xls	8	9
	ppt	2	2
RTF	rtf/doc	69	69
OOXML	docx	1	1
	xlsx	2	2
	pptx	7	7
PDF	pdf	86	86
Total		238	239

表 3 データセットの概要
Table 3 The outline of the dataset.

	手法	悪性画像数	良性画像数
1	全体を縮小する手法	238	239
2	終端部分を抽出する手法	238	239
3	シェルコード部分を抽出する手法	259	239
4	エンコード部分を抽出する手法	174	239

検体は，Virus Total[11] に 2013〜2014 年に登録された doc，xls，ppt，rtf，docx，xlsx，pptx および pdf の拡張子のファイルとした．悪性の検体は，脆弱性を示す CVE タグが付与されている検体とした．良性の検体については，2015 年の時点でいずれの対策ソフトでも検知されない検体とした．

これらの検体から，提案する 4 つの手法を用いて入力画像のデータセットを作成し，悪性あるいは良性のラベルを付与した．作成したデータセットの概要を**表 3** に示す．

手法 1 および手法 2 については，１つのファイルから１つの入力画像を作成する手法である．そのため，良性の画像は悪性の画像と同じ手法で作成した．手法 3 および手法 4 については，１つのファイルから複数の入力画像を作成する方法である．そのため，シェルコードや実行ファイルがエンコードされた部分が発見されないファイルからは，入力画像を作成することができない．手法 3 については，あらかじめ OfficeMalScanner[12] を実行し，シグネチャを検知したすべてのオフセットから入力画像を作成した．OfficeMalScanner は，文書ファイルからシェルコード等のシグネチャに合致する部分を検知し，そのオフセットを表示することが可能である．手法 4 については，あらかじめ Handy Scissors[13], [14] を実行し，実行ファイルを検知したオフセット以降から入力画像を作成した．Handy Scissor は，文書ファイルからエンコードされた実行ファイルを抽出するツールであり，エンコードされた実行ファイルをオフセットを求めることが可能である．手法 3 および

手法 4 の良性ファイルについては，各ファイルからランダムなオフセットを選択して作成した．

5.2 評価指標

本実験では，評価指標として Accuracy, Precision, Recall および F 値 (F-measure) を用いる．各評価指標の定義は以下の数式のとおりである．

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

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

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

$$F - measure = \frac{2Recall \times Precision}{Recall + Precision}$$

式中の TP (True Positive)，FP (False Positive)，TN (True Negative) および FN (False Negative) の関係は**表 4** に示すとおりである．

表 4 クラス分類の結果
Table 4 Confusion matrix for two possible outcomes.

		実際の値	
		正	負
予測結果	正	TP	FP
	負	FN	TN

5.3 実験環境

試験プログラムが動作する実験環境は**表 5** のとおりである．

表 5 実験環境

CPU	Core i7-5820K 6core 3.3GHz
Memory	DDR4 SDRAM 24GB
GPU	GeForce GTX980/4G
OS	Windows-8.1

5.4 実験結果

作成したデータセットを訓練データとテストデータに 9 : 1 の割合で分割し，100 回の訓練を実施した．各回毎の Accuracy の推移を**図 4**，**図 5**，**図 6** および**図 7** に示す．

図中の縦軸は Accuracy の値，横軸は訓練の回数を示す．各手法ともに，概ね 40 回時以降は徐々に安定した値となっている．次に，100 回時のモデルを用い，10 分割交差検証により Precision，Recall および F-measure を求めた．10 分割交差検証の結果を**表 6** に示す．

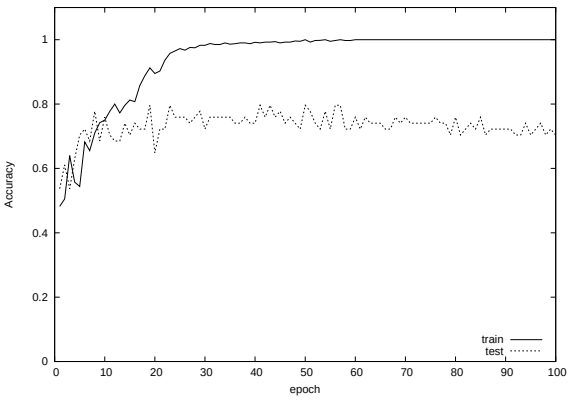

図 4 手法 1 の各回毎の Accuracy
Fig. 4 The accuracy at each epoch of the method 1.

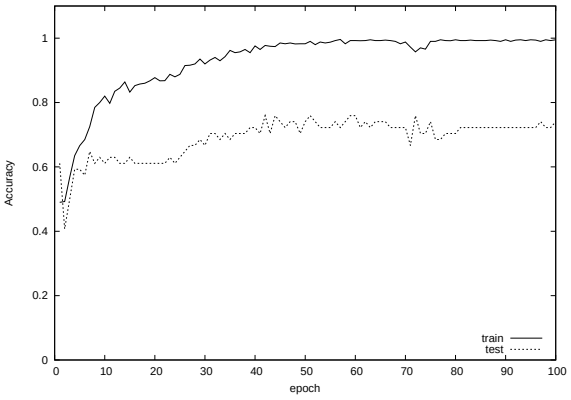

図 5 手法 2 の各回毎の Accuracy
Fig. 5 The accuracy at each epoch of the method 2.

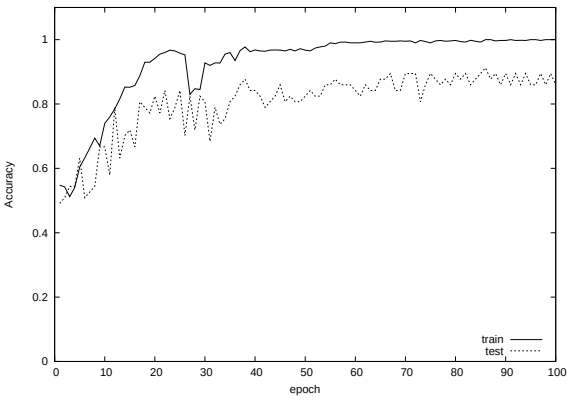

図 6 手法 3 の各回毎の Accuracy
Fig. 6 The accuracy at each epoch of the method 3.

手法 1 および手法 2 については，全般的に Precision も Recall もいまひとつの結果となった．手法 3 については，Precision，Recall ともに最もよい結果となった．手法 4 については，悪性の Recall についてはやや良い結果となったものの，良性についてはもっとも悪い結果となった．この

表 6　10 分割交差検証の結果
Table 6　The result of the 10-fold cross-validation.

	手法	Precision		Recall		F-measure	
		悪性	良性	悪性	良性	悪性	良性
1	全体を縮小する手法	0.75	0.71	0.68	0.78	0.71	0.74
2	終端部分を抽出する手法	0.73	0.75	0.78	0.72	0.75	0.73
3	シェルコード部分を抽出する手法	0.87	0.91	0.92	0.85	0.90	0.88
4	エンコード部分を抽出する手法	0.76	0.71	0.86	0.55	0.81	0.62

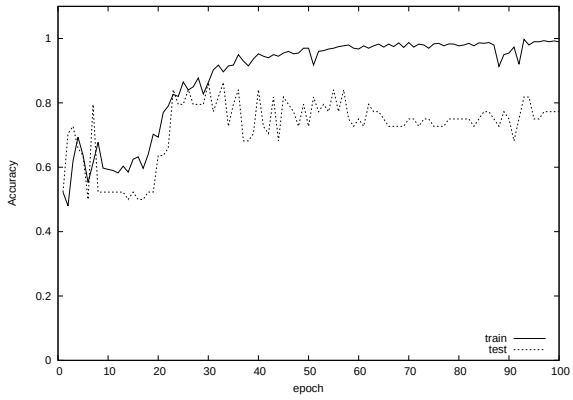

図 7　手法 4 の各回毎の Accuracy
Fig. 7　The accuracy at each epoch of the method 4.

原因は，良性のファイルにおいて，エンコードされた部分と視覚的に類似する箇所があったためであると考えられる．

6. 考察

6.1 「目 grep」の可能性

手法 1 は達人がバイナリファイル全体のビットマップイメージを概観するスキルを想定した手法であり，手法 2 は達人のスキルを想定しない手法であった．今回の実験では，手法 1 および手法 2 において精度に大きな差は現れなかった．この結果から，バイナリファイル全体のビットマップイメージを概観するだけでは，その文書ファイルが悪性か良性かを判断することは困難であると考えられる．

手法 3 および手法 4 は，達人が直接バイナリファイルの 16 進数の値を眺め，通常の文書ファイルには含まれていない不審な部分を発見するスキルを想定した手法であった．手法 3 では，Precision，Recall ともに最もよい結果となった．この結果から，シェルコードの部分は他の部分と比較すると視覚的に特徴があり，「目 grep」による発見がある程度は可能であると考えられる．手法 4 では，悪性の Recall についてはやや良い結果となったものの，良性についてはもっとも悪い結果となった．この結果から，悪性文書ファイルのエンコード部分は他の部分と比較すると視覚的にやや特徴があるものの，良性の文書ファイルにも視覚的に似ている部分があるものと考えられる．また，ダウンローダの場合には，そもそもエンコード部分は存在しない．

以上の考察から，悪性文書ファイルから「目 grep」によってシェルコードを発見することができる可能性があり，未知の文書ファイルを悪性か良性か判断する一指標になるものと考えられる．

6.2 提案手法の用途

今回の実験では，未知のマルウェアやアーティファクトの解析において，動的解析や静的解析も困難であり，バイナリエディタ等でエクスプロイトコード，シェルコード，実行ファイル等の不審な部分の有無を確認し，そのファイルが悪性であるか良性であるかを判断する場合を想定した．

提案手法のその他の用途としては，デジタルフォレンジックにおいて，ハードディスクのイメージからアーティファクトを復元する用途が考えられる．断片化したイメージにはファイルの一部しか残されておらず，パターンマッチング等の手法が困難である場合も珍しくない．また，ファイルを残さないマルウェアについては，メインメモリのイメージを解析して検体を復元する必要がある場合もある．このような場合には，提案する機械学習による視覚的な認識や曖昧な判断基準による「目 grep」が有効であると考えられる．

6.3 提案手法の制約

提案手法は，達人のスキルを機械学習で再現することを試みたものである．この手法は，人間が得意とする視覚的な認識や曖昧な判断基準を基本としている．そのため，IDS (Intrusion Detection System) やウイルス対策ソフトのような厳格な判定を必要とする用途には適しているとは言いがたい．また，その精度についても実運用での検知の用途に耐えられるほど高くはない．そのため，あくまでも未知のマルウェアやアーティファクトの解析や，デジタルフォレンジックにおける補助的な手段として位置づけられるものであると考えられる．

7. おわりに

本稿では，機械学習による「目 grep」の自動化の可能性について検討し，畳み込みニューラルネットワークを用いて「目 grep」を自動化し，悪性文書ファイルを検知するいくつかの手法を提案した．さらに，提案手法を実際の悪性

文書ファイルに適用し，交差検証により精度を確認するとともに，その結果や用途について考察した．その結果，バイナリファイル全体のビットマップイメージを概観するだけでは，その文書ファイルが悪性か良性かを判断することは困難であることを確認した．しかしながら，悪性文書ファイルから「目 grep」によってシェルコードを発見できる可能性があることが確認できた．

今後の課題としては，さらに大きなデータセットを使用しての評価が．今回の実験では，500 体弱の検体を用いてデータセットを作成したが，これは十分な数とは言いがたい．より多くの検体からより多くのデータセットを作成することができれば，提案手法の有効性をより正確に判断することができるものと考えられる．また，デジタルフォレンジック等の他の用途における検証や実用性の評価についても，今後の課題である．

参考文献

[1] 経済産業省: *IT* 人材の最新動向と将来推計に関する調査結果, (2016).

[2] Saxe, J., and Berlin, K.: *Deep Neural Network Based Malware Detection Using Two Dimensional Binary Program Features*, Proc. 10th International Conference on Malicious and Unwanted Software, pp.11-20 (2015).

[3] Chul, E., Shin, R., Song, D., and Moazzezi, R.: *Recognizing Functions in Binaries with Neural Networks*, Proc. 24th USENIX Security Symposium, pp.610-626 (2015).

[4] Pascanu, R., Stokest, J.W., Sanossian, H., Marinescu, M. and Thomas, A.: *Malware Classification with Recurrent Networks*, Proc. 2015 IEEE International Conference on Acoustics, Speech and Signal Processing, pp.1916-1920 (2015).

[5] Zhang, J., Qin, Z., Yin, H., Ou, L., and Hu, Y.: *IRMD: Malware variant Detection using opcode Image Recognition*, Proc. 22nd International Conference on Parallel and Distributed Systems, pp.1175-1180 (2016).

[6] Tobiyama, S., Yamaguchi, Y., Shimada, H., Ikuse, T., and Yagi, T.: *Malware Detection with Deep Neural Network Using Process Behavior*, Proc. 40th Annual Computer Software and Applications Conference, pp.577-582 (2016).

[7] Hardy, W., Chen, L., Hou, S., Ye, Y., and Li, X.: *DL4MD: A Deep Learning Framework for Intelligent Malware Detection*, Proc. 2016 International Conference on Data Mining, pp.61-67 (2016).

[8] Ding, Y., Chen, S., and Xu, J.: *Application of Deep Belief Networks for Opcode Based Malware Detection*, Proc. 2016 International Joint Conference on Neural Networks, pp.3901-3908 (2015).

[9] Shibahara, T., Yagi, T., Akiyama, M., Chiba, D., and Yada, T.: *Efficient Dynamic Malware Analysis Based on Network Behavior Using Deep Learning*, Proc. Global Communications Conference, (2016).

[10] Chainer (online), 入手先 ⟨https://chainer.org/⟩ (2017.07.06).

[11] VirusTotal (online), 入手先 ⟨https://www.virustotal.com/⟩ (2017.07.06).

[12] Boldewin, F.: Analyzing MSOffice malware with OfficeMalScanner(online), 入手先 ⟨http://www.reconstructer.org/⟩ (2017.07.06).

[13] 三村 守，田中英彦 : Handy Scissors: 悪性文書ファイルに埋め込まれた実行ファイルの自動抽出ツール，情報処理学会論文誌, Vol.54, No.3, pp.1211–1219 (2013).

[14] 三村 守，大坪 雄平，田中英彦 : 悪性文書ファイルに埋め込まれた RAT の検知手法，情報処理学会論文誌, Vol.55, No.2, pp.1089-1099 (2014).