Vol.2017-CSEC-79 No.9
2017/12/5

CNN を用いた PE 内関数の類似性によるマルウェア検知手法

田村 壮世† 橋本 正樹†

概要：本研究では，実行ファイルの関数を特徴として利用する畳込みニューラルネットワークを用いたディープラーニングによるマルウェア検知手法を提案する．提案手法は，Windows の実行ファイルである PE 形式の構造に着目するものであり，はじめに，Entry Point の含まれるセクションから複数の関数を抽出し，関数毎に悪性，良性を分類する．その後，その結果から総合的にそのファイルが悪性か良性か判定するものである．評価実験では，提案手法は，98.1％の判定精度で良性と悪性の実行ファイルを判別することができた．今後は，今回使用したデータセット以外の実行ファイルにおいてどの程度の汎化性能を持っているかの検証と，判定箇所の関数のアドレスについて，マルウェア特有の関数を捉えられているか確認していく．

キーワード：マルウェア検知，機械学習，畳込みニューラルネットワーク

A Malware Detection Method by Function Similarity in PE Files using Convolution Neural Network

Moriyo TAMURA† Masaki HASHIMOTO†

Abstract: In this research, we propose a malware detection method based on deep machine learning using convolution neural network (CNN), which uses functions in the PE file as features. In the proposed method, we focus on the structure of the PE format which is the Windows executable file. First, we extract multiple functions from the section containing Entry Point and classify malignant and benign by function. After that, it comprehensively judges whether the file is malignant or benign from the result. As a evaluation of our method, some experiments show that the proposed method is able to distinguish between benign and malicious executable files with a determination accuracy of 98.1%. In the future, we will verify the extent of generalization performance in the executable file other than the dataset we used and check whether malware-specific functions are captured about the function address of the judgment part.

Keywords: Malware Detection, Machine Learning, Convolution Neural Network

1. はじめに

近年，開発ツールや，パッキング，難読化技術の利用によりマルウェアが爆発的に増加を続けている．シマンテックによると，２０１５年には１年間に４億３千万もの新しいマルウェアが検知されており[1]，時間にすると平均して１秒に１３個のマルウェアが作成されていることになる．このため，マルウェアの増加速度に対し，従来のシグネイチャを利用した検知手法ではシグネイチャ作成が間に合わずマルウェアの検知が困難になってきている．また，特定の組織や個人を狙う APT 攻撃では，既存マルウェアの亜種や新たに作成したマルウェアが使用される等の理由により，従来のセキュリティ対策製品を導入していても検知ができず，情報流出等の被害が継続的に発生している[2].

このような状況に対し，マルウェアの検知手法も様々な研究が行われており，近年では機械学習を用いた検知手法が活発に提案されている．学習モデルや特徴点等に関する様々な検討がなされているが，本研究では，実行ファイルの関数を特徴として利用する畳込みニューラルネットワークを用いたディープラーニングによるマルウェア検知手法を提案する．提案手法は，Windows の実行ファイルである PE 形式の構造に着目するものであり，はじめに，Entry Point の含まれるセクションから複数の関数を抽出し，関数毎に悪性，良性を分類する．その後，その結果から総合的にそのファイルが悪性か良性か判定するものである．

以下に，本稿の構成を示す．はじめに，第２章では，マルウェアとその検知手法について概観し，機械学習を用いたマルウェア検知を扱う先行研究を紹介する．次に，第３章では，提案手法の概要と特徴抽出，学習アルゴリズムについて説明する．第４章では，提案手法に対する評価実験について説明し，その結果と考察を示す．最後に第５章で本稿をまとめる．

2. 研究の背景

2.1 マルウェアと検知手法

マルウェア（Malware）とは，Malicious Software の合成

† 情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

語であり，不正かつ有害な動作を行う意図で作成された悪意あるソフトウェアや悪質なコードの総称として用いられている．マルウェアは，感染形態や機能，目的などによってマルウェア，ワーム，トロイの木馬，バックドアなど数多くの種類が存在する．

マルウェアの検知手法には，主にパターンマッチング法，ビヘイビア法，ヒューリスティック法がある[3]．パターンマッチング法は，マルウェアのファイル内に特定のパターンがないかを検査する手法で，ビヘイビア法は，マルウェアの動作が特定の動作パターンに一致しないかを検査する．ヒューリスティック法は，マルウェア動作やファイル内容など様々な特徴を使い類似点がないかを評価し判定する手法で誤検知が多かったが，近年になり機械学習が判定に使われるようになり精度が上がってきている．

2.2 先行研究

Bojan ら[4]は，マルウェアのシステムコールの呼び出し順序を機械学習することでマルウェアを分類する手法を提案している．使用するマルウェアは，VirusShare[5]，Maltrieve[6]，独自のソースの３つから取得し，Cuckoo サンドボックス[7]を用いてシステムコールを取得した．学習に使用するマルウェアの名称は VirusTotal[8]を利用し，VAMO[9]というシステムを使に１０個のマルウェアファミリーに分類した．

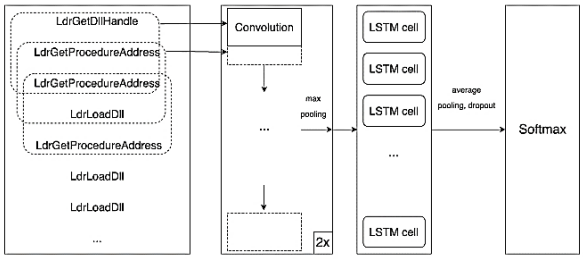

図１ Deep Neural Network architecture
Figure 1 Deep Neural Network architecture.

機械学習には，図 1 に示す 2 つの畳み込み層と 1 つの再帰層を持った Deep Neural Network（DNN）を使用し，システムコールの呼び出し順序の 3-gram を特徴とした．畳み込み層は入力されたデータから特徴を抽出する処理を行うもので，再帰層は時系列データを扱うためのものである．

機械学習の結果，隠れマルコフモデル（HMM）[10]やＳＶＭを利用した手法よりも高い精度でマルウェアを分類することができた．ただし，マルウェアの動作が酷似しているファミリーにおいては分類が困難であった．

Joshua ら[11]は，マルウェアの静的な特徴をニューラルネットワーク使用し学習することで，ヒューリスティック手法の問題である誤検知率を低く抑えつつ検知精度を上げるための手法について検証した．この研究では，悪性 35 万，良性 8 万の実行ファイルから，バイト分布のヒストグラム，インポートアドレステーブル，バイナリに含まれる文字列，ヘッダの値を特徴として抽出し，二つの隠れ層を持つニューラルネットワークを使用し判別を行った．

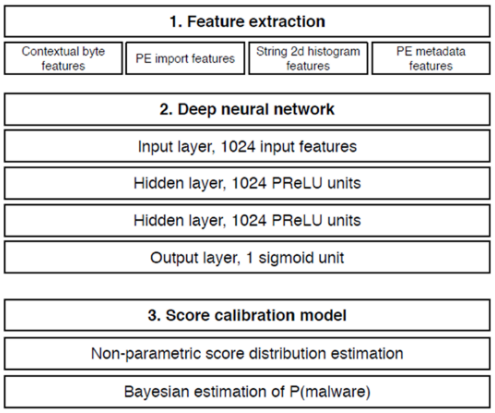

図２ Malware classification framework
Figure 2 Malware classification framework.

その結果，0.1%の誤検知率で 95%の検知率であったことが報告されている．また，この研究では 2001 年１月１日～2014 年７月 31 日までのマルウェアを学習し，それ以降に作成されたマルウェアが検知できるか検証したところ，0.1%の誤検知率で 67%のマルウェアが検知できたと報告されている．

Nataraj ら[12]は，実行ファイルをグレースケース画像として，画像処理技術を用いて分類する手法を提案した．この研究では GIST という画像の特徴を抽出するエンジンを使用し，実行ファイルから 4×4 の画像を２０個（３２０次元）の特徴を抽出し，ユークリッド距離法を用いた K 近傍法を使用し，2000 個のマルウェアを８クラスに分類した．その結果，98%の精度で分類することができたが，この手法では，プログラムの再配置や冗長データを付加する攻撃に弱いことが報告されている．

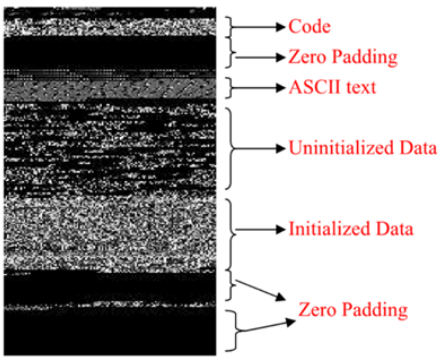

図３ 画像化したバイナリファイル
Figure 3 Imaged binary file.

先行研究においては，マルウェアの動作や，構造の類似性など様々な特徴を機械学習することでマルウェアを検出する手法について提案している．しかしながら，動的な手法では，実際にマルウェアを動作させる必要があり特徴の取得に時間を要してしまう．また，近年ではサンドボックスを検知し動作を変えるマルウェアが存在するなど，特徴取得が困難となってきている．また，静的な手法では，実行ファイルの類似性等を特徴とするが，実際にどの特徴をどのように判定しているのかが不明瞭となり，判定内容の検証が困難である．さらには，単純なダミーデータの付加やバイナリの改変などにより検出が難しくなる問題があった．動的，静的のどちらの特徴を用いる場合でも一長一短があり，目的や環境により使い分けるべきである．本研究においては，マルウェア動作前の検知を目的とするため静的な特徴を用いた手法について提案する．

3. 提案手法

3.1 概要

本研究では，Windows の実行ファイルである PE（Portable Executable）形式の構造に着目し，EP（Entry Point）の含まれるセクションから複数の関数を抽出し，関数毎に悪性，良性を分類する．その結果から総合的にそのファイルが悪性か良性か判定する手法を提案する．この手法では，既存のソースコードや作成ツールを利用したマルウェアを検出することができ，判定箇所の RVA（Relative Virtual Address）が分かるためデバッガ等を使用することで実行ファイルの判定箇所を確認することができる．ただし，パッキングされたファイルの場合，本来の関数が見えないため判定が難しい．

3.2 特徴抽出

本研究では，EP を含むセクションから関数をそれぞれ取得する．

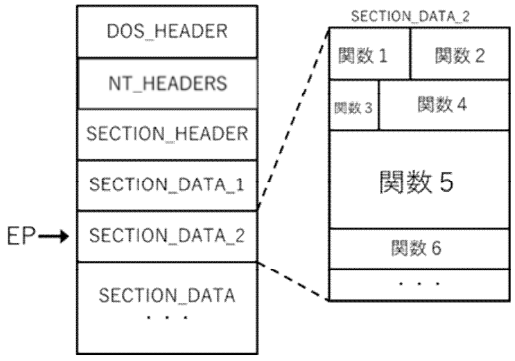

図 4　特徴抽出箇所

Figure 4　About feature extraction part.

3.3 学習アルゴリズム

本研究では，学習アルゴリズムに CNN（Convolutional Neural Network）[13]を用いたディープラーニングを使用する．

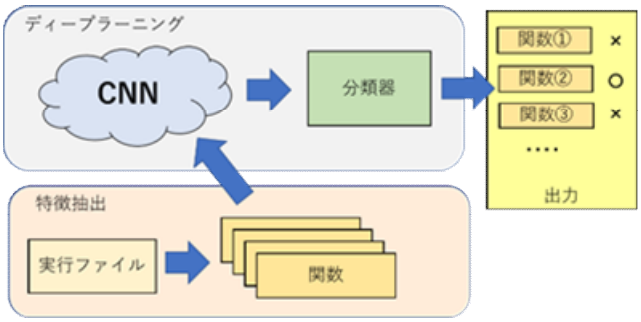

図 5　提案モデル

Figure 5　Proposed model.

CNN は，脳の視覚野の構造を参考にしたニューラルネットワークで，物体認識に秀でたアルゴリズムであるが近年ではテキスト分類やセンチメント分析など物体認識以外のタスクでも使用されている．CNN を使用するメリットとしては，プログラムコードの命令順序を入れ替えたとしても同じものとして扱えることが挙げられる．この学習モデルにより，最終的に一つの実行ファイルに含まれる関数それぞれに対して悪性，無害といった結果が出力される．

4. 評価実験

4.1 データセット

評価実験では，良性の実行ファイル 8,000 個，悪性の実行ファイル 58,068 個からなるデータセットを使用する．良性の実行ファイルは，Windows10，Windows7，Office2016，Visual Studio2015，Adobe 製品等や Vector[14]から取得したもので，悪性の実行ファイルは，VirusShare[5]にて 2014 年 7 月～2016 年 12 月に公開された検体を取得し，Kaspersky アンチウイルスにて 28 種類のマルウェアファミリーに分類したものである（表 1）．

表 1　取得したマルウェアの一覧

Table 1　List of obtained malware.

連番	ファミリー名	個数
1	Trojan	16, 772
2	Downloader	10, 683
3	Trojan-PSW	5, 411
4	P2P-Worm	5, 266
5	Worm	4, 100
6	Trojan-Downloader	3, 919

7	Trojan-GameThief	2,545
8	Backdoor	2,498
9	Email-Worm	1,489
10	Trojan-Dropper	1,215
11	Packed	840
12	Trojan-Spy	807
13	Trojan-Banker	542
14	Trojan-Ransom	377
15	Hoax	308
16	Trojan-Proxy	264
17	Trojan-FakeAV	220
18	Rootkit	202
19	Virus	195
20	Trojan-Clicker	162
21	Net-Worm	133
22	Porn-Downloader	54
23	IM-Worm	36
24	Trojan-DDoS	15
25	Trojan-Mailfinder	11
26	Trojan-Notifier	2
27	IRC-Worm	1
28	Trojan-IM	1
	総数	58,068

表１のデータのうち，個数の差による学習の偏りを避けるため，悪性はそれぞれのファミリーから 1,000 個を上限として 14,169 個のデータを実験に使用する．また，表１の分類では個体数が非常に少ないファミリーもあるため，表２のとおりに分類したものを学習ラベルとして使用する．

表２　データセット

Table 2　List of obtained malware.

分類名	個数	含まれるファミリー名
Benign	8,000	良性ファイル
Backdoor	1,000	Backdoor
Downloader	2,054	Downloader, Trojan-Downloader, Porn-Downloader
Dropper	1,000	Trojan-Dropper
GameThief	1,000	Trojan-GameThief
Packed	840	Packed
Ransom	377	Trojan-Ransom
Rootkit	202	Rootkit
Trojan	4,331	Trojan, Trojan-Banker, Trojan-Clicker, Trojan-DDoS, Trojan-FakeAV, Trojan-IM, Trojan-PSW, Trojan-Spy, Trojan-Proxy, Trojan-Mailfinder, Trojan-Notifier, Hoax
Virus	195	Virus
Worm	3,170	P2P-Worm, Worm, Net-Worm, Email-Worm, IM-Worm, IRC-Worm

機械学習を行うにあたって，表２のデータセットの 90%のデータを訓練データとし，残りの 10%のデータをテストデータとして分類精度の算出のために使用する．

4.2　実験環境

本研究の実験環境を構成する機材は，表３の通りである．

表３　実験環境

Table 3　Experiment environment.

OS	Windows7 64bit
CPU	XEON E5-1620v3 3.5GHz
RAM	48GB
GPU	Geforce GTX1070 RAM 8GB
ツール等	Python 3.6.3, Chainer 2.1.0

4.3　学習モデル

評価実験では，CNN によりプログラムコードから特徴を抽出して分類を行う．CNN に使用したモデルは ResNet[15]と呼ばれるネットワークで，今回のタスクに合わせて縮小させたものを使用する．ResNet は残差ブロックと呼ばれる構造を持ち，前の層の出力を足すことで特徴量の消失を防ぎつつニューラルネットワークの階層を深めることで精度を高めたモデルである．

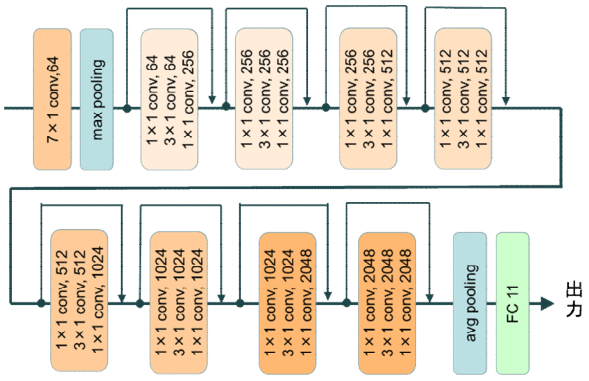

図６　ＣＮＮのネットワーク構成

Figure 6　Network configuration of CNN

4.4 特徴抽出

評価実験では，ファイル毎に EP の含まれるセクションから，正規表現により関数の抽出を行う．具体的には図 7 のように，スタックのベースポインタを変更する命令からリターンコードまでを一つの関数として扱う．分岐命令により複数のリターンがある場合は図 8 のように，より遠いアドレスにあるリターンを関数の終端位置とする．

```
E9 49 F4 FF FF      jmp virusshare_00
55                  push ebp
8B EC               mov ebp,esp
83 C4 F0            add esp,FFFFFFF0
B8 BC AB 41 00      mov eax,virusshar
E8 E1 FB FF FF      call virusshare_0
11 48 00            adc dword ptr ds:
8B C0               mov eax,eax
FF 25 70 11 48 00   jmp dword ptr ds:
8B C0               mov eax,eax
```

図 7　関数の開始位置の判定

Figure 7　Determining the starting position of a function.

```
E8 9D FF FF FF      call virusshare_0
85 C0               test eax,eax
75 05               jne virusshare_00
33 C0               xor eax,eax
5E                  pop esi
5B                  pop ebx
C3                  ret
8B 16               mov edx,dword ptr
89 50 08            mov dword ptr ds:
8B 56 04            mov edx,dword ptr
89 50 0C            mov dword ptr ds:
8B 13               mov edx,dword ptr
89 10               mov dword ptr ds:
89 58 04            mov dword ptr ds:
89 42 04            mov dword ptr ds:
89 03               mov dword ptr ds:
B0 01               mov al,1
5E                  pop esi
5B                  pop ebx
C3                  ret
8B 50 04            mov edx,dword ptr
```

図 8　関数の終端位置の判定

Figure 8　Determining of end position of a function.

抽出する関数は実行ファイルにより数個のものや千個近くもの関数を持つがあり，関数のサイズも非常に短いものから数キロバイトもの長さをもつものが存在する．それらを全て扱うのは困難であることから，今回は関数の長さが 100 バイトから 1,024 バイトの範囲のもので，かつ長さの大きい順から 20 個を用いて機械学習を行う．なお，実行ファイルには JAVA や C#のように中間言語にコンパイルされているものがあるが今回はネイティブコードにコンパイルされているものを対象とする．

4.5 判定方法

評価実験では，学習データを訓練データとテストデータに分け，訓練データを学習に使用する．その後，テストデータを分類できるか評価し，結果を確認する．訓練データには良性ファイルを 7,200 個，悪性ファイルを 12,758 個使用した．テストデータには良性ファイルを 800 個，悪性ファイルを 1,411 個使用する．

テストデータの判定には，分類器による出力結果により，抽出した関数の 30%以上が悪性と分類されたファイルをマルウェアとして判定する．

4.6 評価実験結果と考察

評価実験の結果を表 4，表 5 に示す．

表 4 は判定結果の分布を示したもので，表 5 は正しいクラスへ分類されたかを示す分類精度と良性か悪性かの判定の正しさを示す判定精度を示したものである．

結果から，全てのファミリーにおいて正しいクラスに分類されたファイルが一番多かったが，それに次いでほとんどのファミリーでは Tjojan と誤った分類をされものが多く，分類精度は良い結果を得られなかった．

しかしながら，良性か悪性かの分類の正しさを示す判定精度では，ほとんどのファミリーで 90%後半と高い精度を示し，全体で 98.1%の判定精度であった．

表 4　分類結果の分布

Table 4　Distribution of classification results

番号	ファミリー名	0	1	2	3	4	5	6	7	8	9	10
0	Benign	783	0	3	0	2	0	0	0	12	0	0
1	Backdoor	1	69	5	7	1	0	0	0	16	0	1
2	Downloader	2	6	162	2	1	2	0	1	29	0	0
3	Dropper	3	8	4	66	3	0	0	0	16	0	0
4	GameThief	1	2	2	3	73	1	0	0	17	0	1
5	Packed	3	8	2	0	1	59	0	0	11	0	0
6	Ransom	2	2	2	1	0	3	17	0	10	0	0
7	Rootkit	0	0	0	0	1	0	0	12	7	0	0
8	Trojan	10	25	14	11	10	18	1	0	339	0	2
9	Virus	0	4	1	0	0	1	0	0	5	8	0
10	Worm	3	81	1	0	0	1	0	0	36	1	193

表 5　良性・悪性の分類精度

Table 5　Classification accuracy of benign / malware

ファミリー名	分類精度	判定精度
Benign	0.978	0.978
Backdoor	0.690	0.990
Downloader	0.779	0.990
Dropper	0.660	0.970
GameThief	0.730	0.990
Packed	0.702	0.964
Ransom	0.444	0.946
Rootkit	0.600	1.000
Trojan	0.788	0.977
Virus	0.389	1.000
Worm	0.611	0.991

本提案手法を用いることで，98.1％の判定精度で良性と悪性の実行ファイルを判別することができた．しかし，個別のファミリーへの分類においては，あまり高い精度は得られなかった．

今回の実験では，一つのファイルから関数の長さの大きいものから順に 20 個を取り出し学習に使用した．今回の特徴抽出方法ではマルウェアファミリーを分類するために十分な特徴を取得できていないことが考えられる．このほかにも，アンチウイルスのシグネイチャは実行ファイルのプログラムコードのみではなく，リソースやリソースに埋め込まれた別の実行ファイルを検出している場合もあり，提案手法と検出の部位が異なることから分類は正確には一致しないことが考えられる．

5. まとめ

本研究では，実行ファイル内の関数を特徴として利用する畳込みニューラルネットワークを用いたディープラーニングにより，98.1%の精度で良性の実行ファイルと悪性の実行ファイルを判別することができることを示した．

ディープラーニングでは特徴抽出をニューラルネットワークに行わせるため，入力データのどの箇所を学習し何を判定しているのか不明な場合がある．本手法では，どのアドレスの関数を使って良性・悪性の判定をしたのかが分かるため,デバッガ等で判定箇所を確認することができる．さらに，マルウェア解析等により特定した関数を学習データとして追加することで判定精度を高めていくことが可能であると考えている．

今後は，今回使用したデータセット以外の実行ファイルにおいてどの程度の汎化性能を持っているかの検証と，提案手法のメリットとして考えている判定箇所の関数のアドレスを確認できる点について，マルウェア特有の関数を捉えられているか確認していく必要がある．

参考文献

[1] シマンテック. “インターネットセキュリティ脅威レポート第21号”. https://www.symantec.com/content/ja/jp/enterprise/other_resources/istr-21-exec-summary-jp.pdf, (参照2017-01-14).
[2] 一般社団法人JPCERTコーディネーションセンター. “高度サイバー攻撃（APT）への備えと対応ガイド～企業や組織に薦める一連のプロセスについて”. https://www.jpcert.or.jp/research/20160331-APTguide.pdf, (参照2017-11-04)
[3] 独立行政法人情報処理機構. “未知ウイルス検出技術に関する調査”. https://www.ipa.go.jp/security/fy15/reports/uvd/documents/uvd_report.pdf, (参照2017-11-04)
[4] Bojan Kolosnjaji, Apostolis Zarras, George Webster and Claudia Eckert. Deep Learning for Classification of Malware System Call Sequences. AI 2016: Advances in Artificial Intelligence, pp.137-149, 2016.
[5] Virus Share. https://virusshare.com/. (参照2017-09-04)
[6] Maltrieve. https://github.com/krmaxwell/maltrieve. (参照2017-01-14)
[7] Cuckoo Sandbox. https://cuckoosandbox.org/.(参照2017-01-14)
[8] VirusTotal. http://www.virustotal.com. (参照2017-01-14)
[9] R. Perdisci and M. C. U. VAMO: Towards a Fully Automated Malware Clustering Validity Analysis. In Annual Computer Security Applications Conference. pp.329-338. 2012.
[10] M.Stamp. A Revealing Introduction to Hidden MarkovModels. http://www.cs.sjsu.edu/~stamp/RUA/HMM.pdf,(参照2017-11-04)
[11] Joshua Saxe and Konstantin Berlin. Deep Neural Network Based Malware Detection Using Two Dimensional Binary Program Features. 2015 10th International Conference on Malicious and Unwanted Software. pp.11-20. 2015.
[12] Nataraj, L.Karthikeyan, S.Jacob, G.and Manjunath,B. Malware Images: Visualization and Automatic Clas-sification. Proceedings of the 8th International Symposium on Visualization for Cyber Security, Article No. 4.
[13] LeCun and Bengio. Convolutional Networks for Images, Speech, and Time-Series. The handbook of brain theory and neural networks, pp.255-258, 1995.
[14] Vector. http://www.vector.co.jp/. (参照2017-06-29)
[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun. Deep Residual Learning for Image Recognition. Computer Vision and Pattern Recognition, 2016 IEEE Conference on pp.770-778. 2016.