Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.29, NO.5, Oct. 2019 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2019.29.5.1019

OP Code 특징 기반의 텍스트와 이미지 데이터셋 연구를 통한 인공지능 백신 개발*

최 효 경,[1†] 이 세 은,[1] 이 주 현,[1] 홍 래 영,[1] 최 원 혁,[2] 김 형 종[1‡]
[1]서울여자대학교, [2]누리랩

Development of Vaccine with Artificial Intelligence: By Analyzing OP Code Features Based on Text and Image Dataset*

Hyo-Kyung Choi,[1†] Se-Eun Lee,[1] Ju-Hyun Lee,[1] Rae-Young Hong,[1]
Won-Hyok Choi,[2] Hyung-Jong Kim[1‡]
[1]Seoul Women's University, [2]Nurilab

요 약

지속적으로 새롭게 등장하는 악성 파일(malware)탐지의 어려움으로 인해 머신러닝 기반 인공지능 백신 개발의 중요성이 크게 대두되고 있다. 하지만 현존하는 인공지능 백신은 파일의 일부 영역만을 검사하기 때문에 탐지율이 떨어진다는 단점이 존재한다. 이에 본 논문에서는 독자적인 로직을 기반으로 개발한 인공지능 백신에 근거하여, 파일 내 전체 데이터를 검사하는 방법을 제안한다. 그 중 정상 파일과 비교했을 때 악성 파일에만 존재하는 unique한 함수에서 추출한 OP Code 특징을 학습 데이터셋으로 한 진단법 강화 방안을 제시한다. 해당 강화법의 성능을 Random Forest 알고리즘을 기반으로 한 CSV 데이터셋 학습과 Inception V3 모델을 기반으로 한 이미지 데이터셋 학습으로 나누어 테스트해본 결과, 약 80%의 탐지율을 도출하는 것을 확인할 수 있었다.

ABSTRACT

Due to limitations of existing methods for detecting newly introduced malware, the importance of the development of artificial intelligence vaccines arises. Existing artificial intelligence vaccines have a disadvantage that the accuracy of the detection rate is low because those vaccines do not scan all parts of the file. In this paper, we suggest an enhanced method for detecting malware which is composed of unique OP Code features in the malware files. Specifically, we tested the method with text datasets trained on Random Forest algorithm and with image datasets trained on the Inception V3 model. As a result, the highest accuracy of the detection rate was about 80%.

Keywords: AI Vaccine, Intelligent Vaccine, OP Code feature, Text based, Image based

I. 서 론

악성 파일을 탐지하는 백신의 기존 진단법은 악성코드가 PE(Portable Executable) 형식일 경우에만 탐지가 가능하며, 악성코드로 판단을 내릴 수 있는 특정 형식의 시그니처(signature)를 가지고 있어야 한다[1]. 이러한 진단법은 악성코드의 파일 자체에서 몇 백 바이트만 바뀌어도 진단이 되지 않는

Received(07. 19. 2019), Modified(09. 04. 2019), Accepted(09. 16. 2019)
* 이 연구는 2017년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(NRF-2017R1D1A1B03034644).
† 주저자, gyrud821@gmail.com
‡ 교신저자, hkim@swu.ac.kr(Corresponding author)

미탐(false positive)이 발생하기 때문에, 기존에 알려진 악성코드 외에 새로운 형태의 알려지지 않은 악성코드에 대해서는 대응을 할 수 없다는 단점을 가진다[2]. 이와 같은 한계 극복을 위해 확장 진단법(Generic detection, Heuristic detection, 파일 외형 검사 등)을 사용하여 악성 파일을 탐지하기 시작하였으나[3], 무한한 악성코드의 개체 수 증가 속도는 사람이 해당 진단법들을 사용하여 처리하는 속도를 능가한지 오래이다[4].

악성 파일 탐지에 대한 근본적인 문제들을 해결하기 위하여 최근 머신러닝과 딥러닝을 이용하여 악성코드를 탐지하고자 하는 기술적 추진이 행해지고 있다[5]. 기존 머신러닝 기술을 기반으로 한 악성코드는 탐지 시 악성코드가 발생한 이벤트, 프로세스 행위 기록 등으로 모델 학습을 진행하고 학습한 결과물을 기반으로 기존의 악성코드와 얼마나 유사한지 기준을 산정하여 정상 및 악성 여부를 분류한다[6]. 현재까지 제안된 인공지능 백신들은 이러한 유사성 기반의 방식을 차용하고 있다[7]. 하지만 대다수의 인공지능 백신에서는 정적 분석 시 수집한 파일들 중 압축이 되어 있는 경우에 압축파일을 풀지 않고 특징을 추출하기 때문에 파일의 전체 영역이 아닌 일부 영역만을 검사하여 탐지율이 낮아진다는 단점이 존재한다[8].

이에 본 논문에서는 머신러닝 기반 탐지에 근거하되 기존 악성 파일 진단 문제를 개선 및 해결하기 위해 차별화된 악성 파일 진단 강화 로직을 적용한 인공지능 백신 개발에 대한 연구 결과를 제시한다.

2장에서는 unique 한 함수 내 OP Code 정보를 N-gram 구조로 변경하여 악성 특징을 추출하는 기법을 구조화하여 설명한다. 3장에서는 파일의 전체 영역을 포괄하기 위해 Raw Data 분석을 진행하여 구축한 unique 한 데이터베이스에 대해 이야기한다. 4장에서는 머신러닝 알고리즘 중 Random Forest와 CNN 알고리즘 중 Inception V3 모델을 사용하여 추출한 특징을 기반으로 학습하는 백신 알고리즘과 실험 결과에 대해 기술한다. 마지막으로 5장에서 결론과 향후 연구에 대해 제시한다.

II. 특징 추출 기법

머신러닝 기반의 인공지능 백신 개발의 중요성은 신종 악성코드에 대한 즉각적인 탐지 여부에서 발현된다[9]. 또한 인공지능 백신에서는 학습을 시킬 데이터셋이 어떠한 특징을 포함하느냐에 따라 성능이 크게 좌우된다[10]. 본 연구에서는 독자적인 악성코드 특징 추출을 위해 악성 혹은 정상 파일들에 대한 정적 및 동적 분석을 진행 후 결과를 재분석하였다. 분석 결과 각 파일에 존재하는 여러 함수 내부에서 동작되는 OP Code 값의 유일성을 파악하였고, 이에 해당 값을 기반으로 정상 파일의 함수와 비교했을 때 나타나는 악성 파일의 고유한 특징을 신종 악성 파일 판단을 위한 학습 데이터로 사용할 수 있도록 구조화하였다.

구조화 과정을 통해 백신은 텍스트 버전과 이미지 버전으로 구분된다. 버전의 구분은 학습의 입력 형태를 다양하게 하여 입력 형태에 따른 성능 비교를 위해서 이루어졌다. 이에 텍스트 버전은 TXT Version, 이미지 버전을 Image Version이라 칭한다.

(TXT Version) TXT Version에서는 구분자를 통해 데이터를 취급하기에 용이한 text 형태의 CSV (Comma Separated Value) 확장자 파일을 사용한다. CSV 파일로 데이터셋을 사용하는 TXT Version의 특징 추출 프로세스의 전체 구조도는 Fig.1.과 같다. 데이터베이스에서 black과 white를 검색 keyword로 하여 악성 혹은 정상 파일 (vir 파일) 내에 존재하는 모든 함수에서 얻은

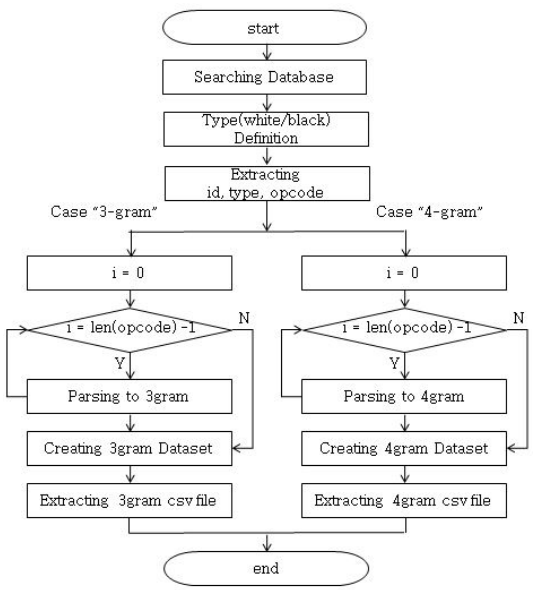

Fig. 1. Flow chart of the feature extracting process

함수 별 OP Code 정보를 3-gram과 4-gram의 형태로 다시 구조화하는 과정을 거친다. N-gram 구조화란, OP Code 데이터를 Nbit만큼 끊어낸 후 문자열을 저장하고 다시 오른쪽으로 1bit 이동하여 이 과정을 반복하는 데이터 구축 방법을 말한다 [11]. 이에 따라 각 함수 마다 추출한 OP Code 길이만큼 파싱을 반복하면서 구조화를 진행하고 전처리 과정으로 text 형식(TXT 형태)의 파일에 저장한다.

그 다음, 특정 문자열이 각 함수에 얼마나 포함되어있는지에 대해 counting 과정을 수행한다. Counting 이란 3-gram의 데이터의 경우 000에서 FFF까지 4096개의 레이블에 대한 값이 OP Code 안에 얼마나 존재하는지를 계산하는 것으로, 4-gram의 경우는 0000에서 FFFF까지 65536개의 레이블 값에 대한 계산을 진행한다. 이 과정을 통해 탐지하려는 파일이 기존의 악성 또는 정상 파일과 비교 시 어떤 유사점을 갖는지 확인할 수 있는 지표를 생산한다. N-gram과 counting의 예시는 Fig.2.와 같다.

3fd9gIldspf23d4	000 001 ··· 3fd··· FFF
→ 3fd fd9 d9g ··· 3d4	0 0 1 0
< 3-gram parsing>	< Counting>

Fig. 2. Example of the 3-gram parsing and the counting process

Counting 된 값을 포함해 추출한 TXT 형태의 파일은 인공지능 학습을 위해 CSV 확장자로 변경한다. 이에 최종 추출된 데이터셋은 vir 파일 내부의 각 함수별로 Fig.3.과 같이 레이블을 가진 CSV 파일 형태로 저장된다. 이후 counting 과정의 결과물인 최종 특징 데이터를 attribute 값으로 하여 숫자 형태를 기반으로 학습과 판단을 하는 Random Forest 알고리즘에서 학습을 진행한다.

origin	type	hash	opcode
0e458b95fbcaa668bd7d71bcaceeba39.vir_(	black	ad00da2c4709c42e7b7ce	646a68506481e8
0e458b95fbcaa668bd7d71bcaceeba39.vir_(	black	08a7348256fd8e2d268d1	565768e8e86ae8
0e458b95fbcaa668bd7d71bcaceeba39.vir_(	black	309f7ae3daeac3337f504a	55568b8b8d50e8
0e458b95fbcaa668bd7d71bcaceeba39.vir_(	black	8f1193e819f92068f7c377	1815355568b578b
0e458b95fbcaa668bd7d71bcaceeba39.vir_(	black	0097abb5032c8e3860132	568b6ae883e8e8
0e458b95fbcaa668bd7d71bcaceeba39.vir_(	black	7381499f9ed45b3273148	558b56578b6ae8
0e458b95fbcaa668bd7d71bcaceeba39.vir_(	black	9e65fe01ee788c123a4782	56578b8575bf8b
0e458b95fbcaa668bd7d71bcaceeba39.vir_(	black	45c7b9de132972a91b809	8153555633578b
0e458b95fbcaa668bd7d71bcaceeba39.vir_(	black	e12da73a7dc0bde3d8f0f	81535556578bc7
0e458b95fbcaa668bd7d71bcaceeba39.vir_(	black	582b843c082451356dc2d	a1835355563357
0e458b95fbcaa668bd7d71bcaceeba39.vir_(	black	d62d61e18f49d6ffd3867e	8b8575b8506868

Fig. 3. Results of the CSV file

(Image Version) Image Version에서는 사용되는 이미지의 공간 정보를 유지한 상태로 학습이 가능한 CNN (Convolutional Neural Network) 알고리즘을 이용해 이미지 학습을 진행하기 위하여 PNG 확장자 파일을 데이터셋으로 사용한다. 여기서는 TXT Version의 과정에서 최종 추출한 counting 값을 이미지 파일로 변환하는 구조화 과정을 거친다.

구조화 과정에서는 TXT 형태로 저장되어 있는 CSV 파일의 특징 데이터 구조를 데이터의 가로, 세로 길이로 조정하여 정사각형 형태의 이미지로 변환한다. 이때 재생성하는 파일 각각에 존재하는 특징 데이터(counting 값)를 나누지 않은 값, %256의 결과, 그리고 %16의 결과로 나누어 저장한다. 특징 데이터 처리 종류를 이와 같이 나누는 이유는 텍스트 형태를 이미지로 변환하기 위해서 binary 또는 hex 값이 필요하기 때문이다. 따라서 2bit를 한 블록으로 사용하는 16진수를 채택하여 2bit 16진수의 최대 범위인 FF($(255)_{10}$)까지 사용할 수 있도록 counting 값을 16의 제곱수인 16과 256으로 나누어 나머지 값을 255 이하로 제한한다. 또한 데이터를 나누지 않은 값의 경우도 이미지화 하여 여러 경우에서의 탐지 결과를 관찰한다.

Fig.4.와 같이 3-gram 데이터셋에서는 %256 또는 %16을 할 때의 나머지 값을 16진수로 변환 시 2bit씩 counting 값이 재조정되므로 4096개의 데이터를 64개씩 나열하여 가로 길이를 64*2bit, 즉 128bit로 한다. 4-gram 데이터셋에서도 마찬가지로 2bit씩 counting 값이 재조정되므로 65536개의 데이터를 256개씩 나열하여 가로 길이를 256*2bit, 즉 512bit로 한다.

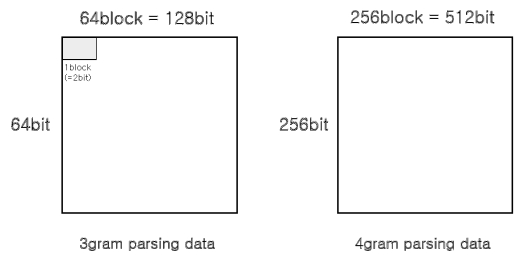

Fig. 4. Data transformation of the Image Version

또한 이미지로의 변환에는 텍스트 데이터를 이미

지로 변경하는 Python의 PIL 라이브러리를 사용한다. 이 때 텍스트 내에 써진 재조정된 숫자의 크기별 색상의 차이를 주기 위해 숫자 데이터의 byte 값에 따라 RGBA 색을 설정하여 이미지화를 진행하였다. 또한 색을 극대화하기 위해 전처리로 생성했던 PNG 파일을 다시 로드하여 RGB 값을 각각 R=255-원본, G=255-원본, B=255-원본으로 재구성하여 색상을 강조하였다.

Fig.5.와 같이 이미지 변환을 통해 생성된 정상 및 악성의 고유 특징을 N-gram화 한 TXT 데이터에 대한 이미지 데이터셋이 생성된다. 이후 컨볼루션 계층과 풀링 계층으로 이루어져 2차원 평면 행렬에서 지정한 영역의 값들을 하나의 값으로 압축하는 CNN 알고리즘[12] 중 이미지 악성코드 학습에 사용할 Inception V3 모델에서 해당 정사각형 모양의 이미지를 학습 데이터로 사용하였다.

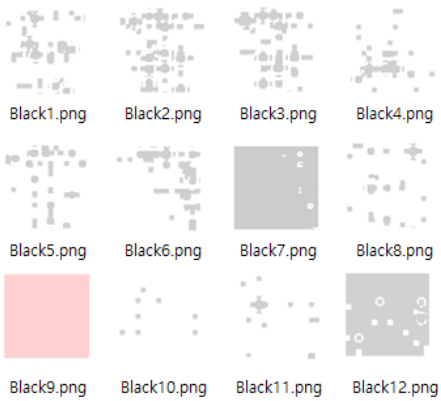

Fig. 5. Results of the transformation

III. 데이터베이스 구축

TXT Version과 Image Version 백신 모두에서 필요한 악성 혹은 정상 파일의 특징 데이터 사용을 위해 서버에서는 데이터베이스를 구축한다. 악성 혹은 정상 파일에 존재하는 정보 추출은 IDA debugger를 이용하였으며, 데이터베이스는 Elasticsearch를 사용하였다.

데이터베이스에 저장하는 정보는 파일에 존재하는 모든 함수에 대해 각각 추출한 '5가지 데이터: 함수명, 함수 해시값, OP Code, type(black or white), 원본 파일명'이며, 이를 Kibana에서 사용할 수 있는 SQL 명령어를 통해 Elasticsearch DB에 삽입한다. 이와 관련한 서버 PC 상의 Kibana 결과는 Fig.6.과 같다.


```
"hits" : {
    "total" : 8375,
    "max_score" : 0.44005156,
    "hits" : [
        {
        "_index" : "reverse",
        "_type" : "func",
        "_id" :
"ad00da2c4709c42e7b7cea3c6cfa760cf33d2a97088ed2776a9
cac3dd21ad212",
        "_score" : 0.44005156,
        "_source" : {
            "origin" :
"0e458b95fbcaa668bd7d71bcaceeba39.vir_cc4739e9b93a57
0237547ae5b10bd46ceac83f51dbe7e542b6c4340fb824e23e",
            "hash" :
"ad00da2c4709c42e7b7cea3c6cfa760cf33d2a97088ed2776a9
cac3dd21ad212"
```


Fig. 6. Results of Kibana

정보 추출 과정은 다음과 같다.

1) IDA를 통한 .vir 파일을 분석 - 먼저 디버깅을 통해 각각의 .vir 파일 안의 OP Code에 포함된 함수의 특징을 분석한다.

2) Entry point로 접근하여 각 함수 관련 정보 수집 - 정보 추출 시 함수의 OP Code는 OP Code의 길이가 14bit 미만인 짧은 경우는 unique 한 함수가 아닐 가능성이 높기 때문에 제외한다.

정보 추출 후 사용하는 type 정보는 .vir 파일이 black인지 white인지를 구분하는 식별자 정보이다. 결과 CSV 파일에서 레이블 데이터를 추출하여 0(정상, white) 또는 1(악성, black)로 저장하였다. DB 구축 시 type 정보를 활용하는 이유는 정상 파일을 모두 DB에 저장한 후 악성 파일을 저장하기 위함이다. 이와 같이 악성 파일을 이후에 저장하는 이유는 중복을 검사하며 정상 구축 후 악성을 저장할 경우 정상과 유사하지 않은 독특한 악성 함수가 black으로 구분될 수 있기 때문이다.

정보 추출 과정 후 DB 구축 시 진행하는 중복 검사는 현재 삽입 데이터 시점에서 바로 이전에 구축된 DB를 기준으로 하여 진행한다. 현재 삽입하려는 함수의 해시값을 search 하여 이미 존재할 경우와 search 한 해시 값에서 추출한 OP Code 결과와 현재 삽입하려는 OP Code 값의 유사도가 80% 이상일 경우에 데이터 삽입을 중지한다. 이때 함수의 해시값은 각각의 악성 및 정상 파일에 존재하는 모든 함수의 고유한 sha-256 해시값을 말한다. 이에 따

라 정상 함수와 비교되는 악성 파일만의 unique 한 데이터를 DB에 구축할 수 있다.

IV. 학습 및 판단

본 연구에서는 KISA의 연구용 악성 및 정상 파일(vir 확장자) 데이터셋을 포함한 약 25000개의 악성 및 정상 파일을 사용하였으며, 위에서 제시한 특징 추출 과정을 통해 학습과 판단에서 사용할 정상(white)함수에 따른 특징 869146개, 악성(black) 함수에 따른 특징 625955개, 총 1495101개의 데이터를 Elasticsearch 데이터베이스에 구축하였다. 구축한 데이터셋 중 텍스트 기반 데이터셋의 경우 지도학습 알고리즘인 Random Forest 알고리즘으로, 이미지 기반 데이터셋은 CNN 알고리즘 모델인 Inception V3 모델을 통해 학습시켰으며, 학습 및 판단 과정에서 사용된 개발 환경은 Table 1.과 같다.

Table 1. Developpment Environments

	Environments
Operating System	Windows 10 Pro (64bit) Intel® Core™ i5-7200 CPU, 8.00 GB
Programming Language	Python 2.7, Python 3.5 Pycharm Community Edition 2018.3
Debugger	IDA Pro 7.0
Database	Elasticsearch 5.4.1, Kibana 5.4.1
Machine Learning	Anaconda2 Sikit-learn library, Tensorflow library
Testing Tool	RapidMiner 0.1.1.0

4.1 Random Forest

TXT Version의 백신의 경우 5개의 알고리즘(Decision Tree, Random Forest, Gradient Boosting, AdaBoost, GNB)을 약 10만개의 랜덤한 데이터셋으로 사전 테스트 하였다. 테스트 결과 약 60~70%의 정확도를 내는 4개의 알고리즘과 대비해 Random Forest 알고리즘의 경우 약 80~90%의 정확도를 내기 때문에 가장 적합한 결과물을 도출한다고 판단하여 Random Forest 알고리즘을 최종적으로 채택하였다.

학습의 정확도를 높이고 판단에서 사용되는 데이터셋을 최소화하기 위해 TXT Version에서 추출한 데이터셋을 8:2의 비율로 랜덤 구분하여 80%의 데이터는 학습용으로 20%의 데이터는 판단용으로 사용하였다. 판단에서는 학습 후 나온 PKL 파일을 사용하였다. 판단 결과는 Fig.7.과 같다. 약 20만 개의 악성과 정상 파일에 대해 80.86%의 탐지율을 보였으며, 해당 결과는 여러 특징 데이터 로직 중 OP Code를 4-gram 형태로 구조화하여 추출한 후 counting 했을 경우이다.

```
(base) bash-3.25 python learning.py



Researching important feature based on 4097 total features

Now testing algorithms
RandomForest : 80.862340 %

Winner algorithm is RandomForest with a 80.862340 % success
Saving algorithm and feature list in classifier directory...
```

Fig. 7. 80.86% detection rate

4.2 Inception V3

TXT Version 백신과의 성능 비교 지표로 사용하기 위해 동일한 데이터셋을 상이한 파일 확장자와 인공지능 알고리즘을 사용하여 악성 혹은 정상을 판별하는 Image Version 학습 및 판단을 진행하였다. Image Version의 성능 측정을 위해서는 3가지의 경우로 학습을 진행하였다. 첫 번째의 경우는 counting 한 값을 그대로 사용하는 경우, 두 번째는 counting 한 값을 256으로 나눈 경우, 마지막으로는 counting 값을 16으로 나눈 경우이다.

학습 알고리즘으로는 Inception V3 모델을 사용하였으며, 마찬가지로 데이터셋을 8:2의 비율로 나누어 80%의 데이터셋의 경우 1000회의 training step을 거쳐 학습을 진행하였고 이에 따른 이미지 특징에 따른 그래프를 도출했다. 결과 그래프를 사용하여 20%의 데이터셋에 대해 판단을 진행한 결과 73.7%의 탐지율을 보였다. 해당 결과는 3-gram 형태로 구조화하여 counting 한 데이터를 특정한 수로 나누지 않고 이미지화한 PNG 파일로 학습 및 판단을 진행한 경우이다. 학습 및 판단에 대한 과정은 1000회의 step을 거쳐 학습 후 데이터셋에 대해

```
2018-11-30 22:04:35.532556: Step 960: Cross entropy = 0.460271
2018-11-30 22:04:35.679371: Step 960: Validation accuracy = 56.0% (N=100)
2018-11-30 22:04:36.918923: Step 970: Train accuracy = 76.0%
2018-11-30 22:04:36.919375: Step 970: Cross entropy = 0.497968
2018-11-30 22:04:37.042382: Step 970: Validation accuracy = 61.0% (N=100)
2018-11-30 22:04:38.355293: Step 980: Train accuracy = 82.0%
2018-11-30 22:04:38.355789: Step 980: Cross entropy = 0.427688
2018-11-30 22:04:38.481276: Step 980: Validation accuracy = 57.0% (N=100)
2018-11-30 22:04:39.832878: Step 990: Train accuracy = 81.0%
2018-11-30 22:04:39.832878: Step 990: Cross entropy = 0.446222
2018-11-30 22:04:40.018383: Step 990: Validation accuracy = 53.0% (N=100)
2018-11-30 22:04:41.136366: Step 999: Train accuracy = 85.0%
2018-11-30 22:04:41.136366: Step 999: Cross entropy = 0.409510
2018-11-30 22:04:41.256880: Step 999: Validation accuracy = 52.0% (N=100)
Final test accuracy = 73.7% (N=38)
```

Fig. 8. Process of the Image Version vaccine

판단한 결과 값인 Fig.8.과 같다. 결과 값 중 N은 캐시 되었던 bottleneck feature 값이다.

V. 결 론

본 논문에서는 기존 악성코드 탐지 과정에서 악성 파일의 증가 비율에 따른 처리 속도의 문제와 변종 미탐에 따른 문제를 해결하기 위한 방법으로 새로운 특징 추출법을 접목시킨 인공지능 기반 백신 개발을 통한 진단 강화법을 기술하였다.

인공지능으로 학습할 특징 추출을 위해서는 악성 및 정상 파일 내 함수 OP Code 데이터를 이용하였으며, 이 OP Code 내 특정 문자열이 레이블 값에 따라 어떤 양상으로 존재하는지 counting 하는 방식을 통해 Fig.9.과 같은 구조에 따라 텍스트 파일과 이미지 파일로 학습 데이터셋을 나누어 학습과 판단을 진행하였다.

판단의 결과는 Fig.10.와 같이 점진적으로 증가하였으며, 결과적으로 TXT Version에서는 약 80%의 정확도, Image Version에서는 약 70%의 정확도로 악성과 정상 샘플 파일을 탐지하는 것을 확인하였다. 특히 텍스트 데이터셋의 경우 4-gram 특징 기반에서, 이미지 데이터셋의 경우 3-gram 특징 기반에서 탐지 효율이 높음을 확인하였다.

본 연구에서 제안하는 악성파일 탐지 모델은 인공지능 기술을 통해 변종 및 신종 악성코드를 탐지할 수 있지만, 다음과 같은 단점을 가진다. 첫째로 TXT 데이터셋 기반의 학습에서는 N-gram 데이터의 N값이 커질수록 탐지 정확도가 상승하나 이에 따른 메모리 과부하 문제가 발생한다. 둘째로 Image 데이터셋 기반의 학습에서는 어떠한 알고리즘에서 이미지 특징이 극대화될 수 있는지에 대한 추가 연구가 필요하다.

이에 따라 추후 연구에서는 고사양의 환경에서 최적화 알고리즘과 N 값을 증가시킨 N-gram 데이터를 활용하여 문제에 대한 개선안을 개발한다. 추후 연구를 통해 악성 탐지율을 높일 경우, 범용적인 악성 파일과 랜섬웨어 혹은 트로이목마 등 특정 악성 파일뿐만 아니라 신종 악성 파일에 대한 탐지 백신으로써 기능하여 사이버 보안 분야에서 발생하는 각종 이슈에 빠르게 대응할 수 있을 것이다.

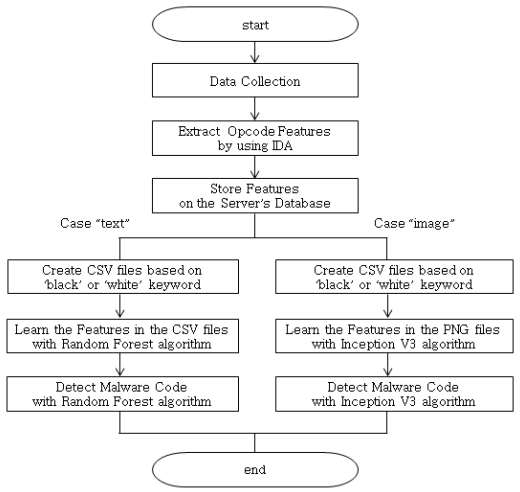

Fig. 9. Flow chart of the overall vaccine structure

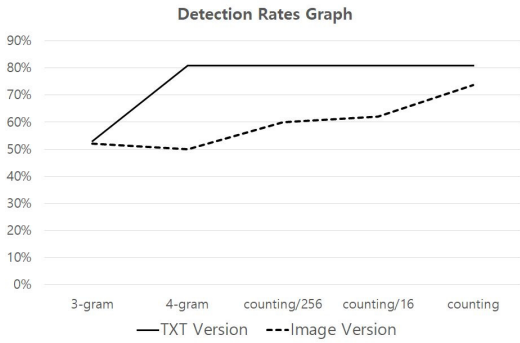

Fig. 10. Detection rates graph

References

[1] N. Idika and A.P. Mathur, "A survey of malware detection techniques," Department of Computer Science, Purdue University, Feb. 2007.

[2] T. Lee, B. Choi, Y. Shin, and J. Kwak, "Automatic malware mutant

detection and group classification based on the n-gram and clustering coefficient," The Journal of Supercomputing, vol. 74, no. 8, pp. 3489-3503, Aug. 2018.
〔3〕 Z. Cui, F. Xue, X. Cai, Y. Cao, G. Wang, and J. Chen, "Detection of malicious code variants based on deep learning," IEEE Transactions on Industrial Informatics, vol. 14, no. 7, pp. 3187-3196, Jul. 2018.
〔4〕 P.V. Shijo and A. Salim, "Integrated static and dynamic analysis for malware detection," Procedia Computer Science, vol. 46, pp. 804-811, Jan. 2015.
〔5〕 I. Baptista, S. Shiaeles, and N. Kolokotronis, "A novel malware detection system based on machine learning and binary visualization," arXiv preprint arXiv:1904.00859, Apr. 2019.
〔6〕 H. Cui, Y. Zhou, C. Wang, Q. Li, and K. Ren, "Towards privacy-preserving malware detection systems for android," 2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS), pp. 545-552, Dec. 2018.
〔7〕 A. Pfeffer, B. Ruttenberg, L. Kellogg, M. Howard, C. Call, A. O'Connor, G. Takata, S.N. Reilly, T. Patten, J. Taylor, R. Hall, A. Lakhotia, C. Miles, D. Scofield, and J. Frank, "Artificial intelligence based malware analysis," arXiv preprint arXiv: 1704.08716, Apr. 2017.
〔8〕 W. Fleshman, E. Raff, R. Zak, M. McLean, and C. Nicholas, "Static malware detection & subterfuge: quantifying the robustness of machine learning and current anti-virus," 2018 13th International Conference on Malicious and Unwanted Software (MALWARE), pp. 1-10, Oct. 2018.
〔9〕 I. Firdausi, A. Erwin, and A.S. Nugroho, "Analysis of machine learning techniques used in behavior-based malware detection," 2010 second international conference on advances in computing, control, and telecommunication technologies, pp. 201-203, Dec. 2010.
〔10〕 S. Khalid, T. Khalil, and S. Nasreen, "A survey of feature selection and feature extraction techniques in machine learning," 2014 Science and Information Conference, pp. 372-378, Aug. 2014.
〔11〕 T. Abou-Assaleh, N. Cercone, V. Keselj, and R. Sweidan, "N-gram-based detection of new malicious code," Proceedings of the 28th Annual International Computer Software and Applications Conference, vol. 2, pp. 41-42, Sep. 2004.
〔12〕 K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: visualising image classification models and saliency maps," arXiv preprint arXiv: 1312.6034v2, Apr. 2014.

〈저자소개〉

최 효 경 (Hyo-Kyung Choi) 학생회원
2019년 8월: 서울여자대학교 정보보호학과 학사
〈관심분야〉 정보보호, 인공지능, 네트워크, 악성코드

이 세 은 (Se-Eun Lee) 학생회원
2016년 3월~현재: 서울여자대학교 정보보호학과 학사과정
〈관심분야〉 정보보호, 인공지능, 디지털 포렌식

이 주 현 (Ju-Hyun Lee) 학생회원
2019년 8월: 서울여자대학교 정보보호학과 학사
〈관심분야〉 정보보호, 인공지능

홍 래 영 (Rae-Young Hong) 학생회원
2019년 8월: 서울여자대학교 정보보호학과 학사
〈관심분야〉 정보보호, 인공지능

최 원 혁 (Won-Hyok Choi) 정회원
1997년 2월: 경일대학교 컴퓨터공학과 학사
2001년 2월: 동국대학교 정보보호학과 석사
2015년 2월: 동국대학교 정보통신공학과 컴퓨터전공 박사과정 수료
〈관심분야〉 디지털 포렌식, 악성코드 분석, 해킹, 정보 통신, 정보 보호

김 형 종 (Hyung-Jong Kim) 종신회원
1996년: 성균관대학교 정보공학과 공학사
1998년: 성균관대학교 정보공학과 공학석사
2001년: 성균관대학교 전기전자 및 컴퓨터공학과 공학박사
2001~2007년: 한국정보보호진흥원 수석연구원
2004~2006년: 미국 Carnegie Mellon University, CyLab 국제공동연구원
2013~2014년: 미국 Carnegie Mellon University, ECE, Visiting Professor
2007~현재: 서울여자대학교 정보보호학과 정교수
〈관심분야〉 안드로이드 환경의 IoT서비스 개인정보보호, 블록체인 서비스 성능평가, 클라우드 서비스 보안 모델