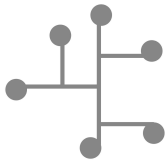


딥 러닝 기반 분류 모델을 이용한 악성코드 제작자 그룹 분류

Malware Author Group Classification using Deep Learning Classifier

홍석진(Suk-Jin Hong)[1] 홍지원(Jiwon Hong)[2] 김상욱(Sang-Wook Kim)[3]
김동필(Dongphil Kim)[4] 김원호(Wonho Kim)[5]

요 약

컴퓨터가 실생활에서 많이 사용됨에 따라, 악성코드(malware)를 만들어 악의적인 목적으로 다른 사람의 컴퓨터를 공격하려는 시도가 기하급수적으로 증가하고 있다. 악성코드들은 해당 악성코드를 제작한 제작자 그룹을 기준으로 분류될 수 있으며, 악성코드 제작자 정보는 디지털 포렌식(digital forensic)에 중요한 정보로 활용될 수 있다. 본 논문에서는 악성코드로부터 정적 특징 정보와 동적 특징 정보를 추출하여 악성코드를 각 특징의 보유 유무로써 표현하였다. 이를 바탕으로 딥 러닝 기법을 활용하여 주어진 악성코드의 제작자 그룹을 분류하는 방안을 제안하였다. 본 논문에서는 다양한 실험을 통해 악성코드 데이터에 맞는 딥 러닝 기법과 하이퍼 파라미터를 찾아 딥 러닝 기반 악성코드 제작자 그룹 분류 모델을 구축하고 평가하였으며, 본 논문에서 제안한 딥 러닝 기반 분류 모델이 기존 분류 모델보다 악성코드 제작자 그룹 분류 문제에서 높은 정확도를 보임을 확인하였다.

주제어: 악성코드, 분류, 딥 러닝

1 한양대학교 컴퓨터·소프트웨어학과, 석사과정.
2 한양대학교 컴퓨터·소프트웨어학과, 박사과정.
3 한양대학교 컴퓨터·소프트웨어학과, 교수, 교신저자.
4 ETRI 부설 연구소, 선임연구원.
5 ETRI 부설 연구소, 실장.
+ 이 논문은 ETRI부설연구소의 “머신러닝을 이용한 악성코드 연관 관계 분석 연구” 연구과제의 지원을 받아 수행된 연구임
+ 이 논문은 2017년도 정부(과학기술정보통신부)의 재원으로 한국연구재단-차세대정보·컴퓨팅기술개발사업의 지원을 받아 수행된 연구임(No. NRF-2017M3C4A7069440).
+ 논문접수: 2018년 07월 22일, 심사완료: 2018년 08월 06일, 게재승인: 2018년 08월 14일.

Abstract

As computers are heavily used in real life, attempts at creating malwares to attack others' computers for malicious purposes are increasing exponentially. Malwares can be categorized based on the group of authors who created the code, and their information is considered to be important for digital forensics. In this paper, we extract the static features and the dynamic features from the malware and use the features to represent the malware by considering the presence or absence of each feature in the malware. Based on the feature information, we proposed a method to classify a group of authors of a given malware by using a deep learning technique. Also, we find a hyperparameter and a deep learning technique that work best on the malware author group classification via extensive experiments. Using these, we construct and evaluate a deep learning based malware author group classification model. We confirmed that the classification accuracy of the proposed model is higher than those of the existing malware author group classification models.

Keywords: malware, classification, deep learning

1. 서 론

최근 컴퓨터 기술이 발달함에 따라 실생활에서 컴퓨터가 많이 사용되고 있다. 많은 정보들이 컴퓨터에 저장되고 있으며, 대부분의 금융거래도 컴퓨터와 컴퓨터 네트워크를 기반으로 이루어지고 있다. 또한 IoT(Internet of Things) 기술의 발달에 따라 실생활에서 사용되는 냉장고, TV 등에 크고 작은 컴퓨터가 탑재되고 있으며, 모두 네트워크를 기반으로 연결되고 있다.

위와 같이 사람들의 컴퓨터에 대한 의존도가 커지고, 각 컴퓨터들이 네트워크를 통해 모두 연결됨에 따라 컴퓨터를 악의적인 목적으로 사용하려는 시도가 나타나기 시작했다. 이에 다른 사람들에게 금전적 피해 및 정보 탈취 등과 같은 악의적인 목적을 달성하기 위해 작성된 실행 가능한 코드(악성코드)가 발생하기 시작했고, 컴퓨터 네트워크를 통해 빠르게 전파되고 있다. 또한 최근 들어 새롭게 발생하는 악성코드의 수가 기하급수적으로 증가하고 있다. [1, 2, 3]

새롭게 등장하는 악성코드의 대부분은 기존 악성코드를 일부 변형하여 생성된 기존 악성코드의 변형인 경우가 많다[1, 2, 4, 5]. 악성코드의 제작자 그룹들은 단시간에 효율적으로 목적을 달성하기 위해 위와 같은 방식을 택하고 있다.

따라서 악성코드 사이의 유사성을 이용하여, 새롭게 발생하는 악성코드들을 기존에 있는 악성코드 제작자 그룹으로 분류하는 것이 가능하다. 악성코드들의 제작자 그룹 정보는 악성코드와 관련된 디지털 포렌식(digital forensic)에 자료로 활용 될 수 있다.

악성코드의 제작자 그룹 분류는 기존에 사용되고 있는 다양한 분류 모델을 통해 이루어 질 수 있다. 악성코드의 제작자 그룹 분류를 위해서는 분류에 사용될 특징 정보들이 필요하다.

악성코드의 특징 정보는 악성코드의 이진(binary) 파일에서 직접 추출하거나, 가상 환경에서 악성코드를 실행하고 악성코드의 행동을 관찰하여 얻을 수 있다. [1, 4, 6] 악성코드 제작자 그룹 분류의 정확성을 높이기 위해 다양한 특징 추출 기법들이 연구되고 있다. [1, 2, 3, 4]

이처럼 추출된 악성코드 특징 정보를 이용하여, 악성코드 제작자 그룹 분류 문제를 해결하는 방법들이 연구되어 오고 있다. [6]에서는 decision tree, SVM, random forest, naive Bayesian classifier를 이용하여 악성코드를 제작자 그룹으로 분류했다.

최근 가장 각광받고, 많이 연구되는 분류 모델로 딥 러닝 기반 분류 모델이 있다. 딥 러닝 기반 분류 모델은 특징 정보를 여러 비선형 변환 기법의 조합을 통해 추상화하며, 추상화된 정보를 바탕으로 그룹으로 분류하는 모델이다. 현재 많은 분야에서 기존 분류 모델보다 정확도가 높은 분류 모델로 알려져 있다.

하지만 기존 악성코드 제작자 그룹 분류 문제를 딥 러닝 기반 분류 모델을 이용하여 해결하고자 하는 시도가 없었으며, 이에 따라 본 논문에서는 다양한 딥 러닝 기법들과 파라미터 튜닝을 통해 악성코드 제작자 그룹 분류를 정확하게 수행할 수 있는 딥 러닝 기반 분류 모델을 연구하고자 한다. 또한 딥 러닝 기반 분류 모델이 기존 분류 모델의 정확도를 비교하고자 한다,

본 논문의 2장에서는 악성코드 데이터에 대해 설명한다. 3장에서는 딥 러닝 기반 분류 모델에 대해 서술하고, 4장에서는 다양한 딥 러닝 기법에 대해 설명한다. 5장에서는 파라미터 튜닝 및 딥 러닝 기법 적용 결과를 실험을 통해 보이며, 기존 분류 모델과 비교평가 한다. 6장에서는 결론으로 본 논문을 끝맺음한다.

2. 악성코드 데이터

악성코드를 분류 모델을 이용하여 각 제작자 그룹으로 분류하기 위해서는 분류를 위한 악성코드의 특징 정보가 필요하다. 본 논문에서는 위 문제를 해결하기 위해 악성코드 제작자 그룹이 기존 코드를 재사용한다는 점을 이용하였다.

악성코드 제작자 그룹이 코드를 재사용함에 따라 나타나는 특징 정보들은 크게 정적(static) 특징 정보와 동적(dynamic) 특징 정보로 분류된다.

2.1 정적 특징 정보

악성코드의 실행 가능한 이진(binary) 형태의 파일에서 직접 얻을 수 있는 특징 정보를 의미한다. 정적 특징 정보는 해당 이진 파일을 역어셈블(disassemble)하거나 이진 파일 내에서 찾을 수 있는 식별 가능한 문자열 등을 탐색하면서 얻을 수 있다. 본 논문에서 사용한 정적 특징 정보에 해당되는 세부 특징 정보들은 다음과 같다.

2.1.1 문자열 (STIRINGS)

악성코드의 이진 파일에서 찾을 수 있는 식별 가능한 문자열을 의미한다. 이 가운데 대부분의 문자열 정보는 의미가 없다. 하지만 동일한 제작자가 습관적으로 사용한 문자열이나 상징성을 띄는 특정한 문자열의 경우 악성코드 제작자 그룹 분류 문제에서 유용한 특징 정보로 사용될 수 있다.

2.1.2 함수 (FUNCTIONS)

악성코드의 이진 파일을 역어셈블 하여 얻을 수 있는 명령어들 가운데, 함수 단위에 있는 opcode 리스트의 해싱값을 의미한다. 앞서 밝힌 바와 같이 악성코드 제작자들은 기존 코드를 재사용하는 경우가 많다. 이 경우 동일한 패턴을 갖는 함수 단위의 opcode 리스트들이 여러 코드에서 나타나게 되므로, 악성코드 제작자 그룹 분류 문제에서 유용한 특징 정보로 사용될 수 있다. opcode 리스트의 길이가 가변적이고 용량이 큰 경우가 자주 있어, opcode 리스트를 해시 함수(hash function)를 이용하여 변환한 뒤, 이를 특징 정보로 사용한다.

2.1.3 익스포트 (EXPORTS)

악성코드 파일이 다른 악성코드 파일에 의해 사용되는 API명을 의미한다. 악성코드는 일반적으로 온전한 하나의 파일을 갖는 것이 아닌 여러 개의 파일을 가지고 있으며, 다른 악성코드 파일을 임포트 혹은 익스포트 하면서 동작한다. 코드 재사용이 일어나는 경우 동일한 익스포트 패턴이 나타나게 되므로, 익스포트 정보가 제작자 그룹 분류 문제를 위한 효과적인 특징 정보가 될 수 있다.

2.2 동적 특징 정보

악성코드가 실제 동작할 때 나타나는 행동들을 통해 얻을 수 있는 특징 정보를 의미한다. 일반적으로 동적 특징 정보는 악성코드를 가상 환경에서 실행하고, 악성코드의 행동을 관찰하여 수집된다. 본 논문에서 사용한 동적 특징 정보에 해당되는 세부 특징 정보는 다음과 같다.

2.2.1 파일 (FILES)

악성코드가 실행되는 과정에서 읽기, 쓰기, 삭제 등이 수행된 모든 파일의 경로를 의미한다. 몇몇 악성코드들은 자신의 존재를 숨기거나 컴퓨터를 직접 공격하기 위해, 침투한 컴퓨터에 존재하는 파일들을 읽기 · 쓰기 · 삭제하는 행동을 보인다. 일반적으로 악성코드 제작자는 악성코드가 생성한 파일을 일반

사용자가 중요한 시스템 파일로 착각하도록 시스템 파일이 위치하는 경로에 저장하고, 다른 시스템 파일과 유사하게 이름 짓는 경우가 많다. 이 때 지어지는 이름 및 저장되는 경로는 악성코드 제작자에 의해 결정되므로, 파일 경로 정보가 악성코드 제작자 그룹 분류 문제에서 유의미하게 사용될 수 있다.

2.2.2 네트워크 (NETWORKS)

악성코드가 실행되면서 사용되는 DNS, URL, IP 주소를 의미한다. 악성코드가 요청한 네트워크 패킷(packet)을 분석하여 얻는다. 동일한 제작자에 의해 만들어진 악성코드들은 유사한 목적을 띄는 경우가 많다. 따라서 악성코드가 실행되면서 사용되는 네트워크 주소가 동일하게 사용되는 경우가 많아 효과적인 특징 정보로 사용될 수 있다.

2.2.3 뮤텍스 (MUTEXES)

악성코드의 제작자들이 유사한 악성코드가 침투한 하나의 컴퓨터에서 중복 실행되는 것을 방지하기 위해 사용하는 상호 배재 뮤텍스의 이름을 의미한다. 같은 제작자 그룹을 갖는 악성코드는 서로 중복 실행되는 것을 방지하기 위해, 동일 그룹 내에서 뮤텍스 이름을 공유한다. 따라서 뮤텍스 특징 정보가 제작자 그룹 분류를 위한 효과적인 특징 정보가 될 수 있다.

2.2.4 드롭 (DROPS)

악성코드가 악성코드 파일 내에 저장되어 있는 파일을 추출하여 저장하거나, 네트워크를 통해 다운받은 파일을 저장하는 경로, 원격지 주소, 파일 해시 값을 의미한다. 동일 제작자 그룹에 의해 제작된 악성코드들은 동일한 경로, 원격지 주소, 파일을 사용하는 경우가 많다. 따라서 드롭 정보도 효과적인 특징 정보로 사용될 수 있다.

2.2.5 레지스트리 (REGISTRY)

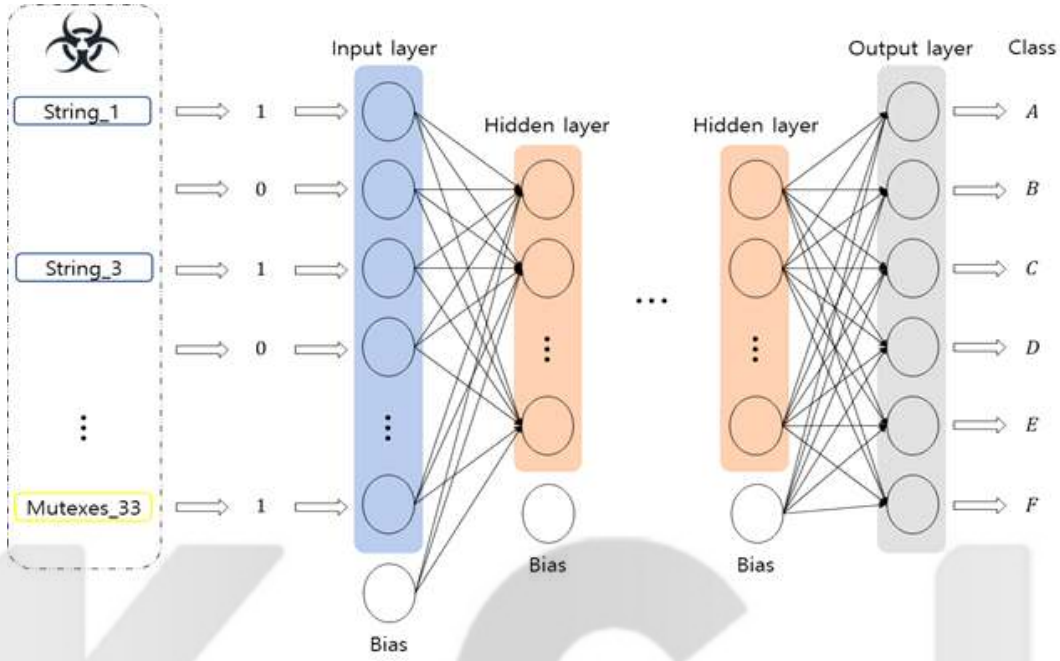
악성코드가 접근 및 수정하는 레지스트리를 의미한다. 마이크로소프트사의 윈도우 운영체제를 사용하는 컴퓨터에는 레지스트리 정보가 저장되어 있다. 레지스트리는 윈도우 운영체제나 응용프로그램에 대한 설정 정보를 가지고 있다, 동일 제작자 그룹에 의해 만들어진 악성코드는 동일 레지스트리에 접근 및 수정하는 경우가 많아, 효과적인 특징 정보로 활용될 수 있다.

다수의 악성코드로 부터 앞서 설명한 정적 특징 정보 · 동적 특징 정보를 추출하면 각 악성코드들을 특정 특징 정보의 보유 유무로써 표현할 수 있다.

3. 딥 러닝 기반 분류 모델

딥 러닝 기반 분류 모델은 선형 맞춤과 비선형 변환 기법의 조합을 통해 복잡한 데이터의 핵심적인 부분을 요약하고, 요약된 내용을 기반으로 분류를 진행하는 모델을 의미한다. 현재 가장 활발하게 연구되고 있는 분류 모델이며, 많은 분야에서 기존 분류 모델보다 정확도가 높은 것으로 알려져 있다.

모델의 구조 관점에서 딥 러닝 기반 분류 모델은 크게 완전 연결(fully connected) 신경망 분류 모델, 합성곱 신경망(CNN) 기반 분류 모델, 순환 신경망(RNN) 기반 분류 모델로 구분된다. 완전 연결 신경망 분류 모델은 대부분의 데이터에서 사용되는 분류 모델이며, 합성곱 신경망 기반 분류 모델은 이미지 데이터에서, 순환 신경망 기반 분류 모델은 순차 데이터나 시계열 데이터에서 주로 사용된다.



[그림 1] 악성코드를 위한 완전 연결 신경망 기반 분류 모델

본 과제에서는 완전 연결 신경망 기반 분류 모델을 사용하였다.

완전 연결 신경망 기반 분류 모델은 각 층의 정점들과 각 층의 앞 층의 정점들이 완전하게 연결되어 있는 구조를 가진 분류 모델이다. 위 [그림 1]은 악성코드를 위한 완전 연결 신경망 기반 분류 모델을 나타낸다.

모델은 여러 층(layer)으로 구성되어 있다. 악성코드 데이터가 들어가는 부분의 층이 입력층(input layer)이며, 입력층과 반대 방향으로 모델의 최종 결과 값이 출력되는 층이 출력층(output layer)이다. 그리고 입력층과 출력층 가운데 있는 층들이 은닉층(hidden layer)이 된다. 각 층들에는 정점이 존재하며, 각 정점들은 자신의 앞뒤층의 모든 정점들과 간선으로 연결되어 있다. 각 간선들은 특정 가중치 값을 가지고 있다. 입력층과 출력층을 제외한 각 층의 정점의 앞부분에는 비선형 활성화 함수가 있다.

모델을 학습할 때, 하나의 악성코드가 입력층으로 들어오게 되면 모델의 구조에 따라 입력층에서 출력층 방향으로 선형결합과 비선형변환이 반복해서 이루어지면서 계산이 진행된다. 최종적으로 악성코드는 출력층에서 각 그룹별 실수 값으로 나타나게 되며 해당 실수 값과 손실함수, 실제 악성코드 데이터의 제작자 그룹 정보를 이용하여 손실함수의 현재 그래디언트(gradient)를 계산하고, 손실함수 값을 줄이는 방향으로 간선의 가중치를 출력층 방향에서 입력층 방향으로 단계별로 업데이트한다.

모델을 이용하여 분류할 때는 학습과정과 동일하게 하나의 악성코드를 모델의 입력층에서 출력층 방향으로 간선의 가중치를 기반으로 계산한다. 출력층에서 최종 계산된 각 제작자 그룹별 실수 값 가운데 가장 큰 값을 해당 악성코드 데이터의 샘플로 분류한다.

4. 다양한 딥 러닝 기법

일반적으로 딥 러닝 기반 분류 모델들에서 나타나는 문제점으로 과적합 문제와, 그래디언트 소실 및 폭주 문제가 있다.

과적합 문제는 딥 러닝 기반 분류 모델이 훈련 데이터에 과도하게 적합되어 평가 데이터나 실제 데이터에서 정확도가 떨어지는 현상을 말한다. 과적합 문제를 해결하기 위해 다양한 규제 기법이 제안되었으며, 그 가운데 대표적인 기법으로 dropout이 있다. [7]

그래디언트 소실 및 폭주 문제는 출력층에서 입력층 방향으로 갈수록 그래디언트가 비정상적으로 작아지거나 커지는 현상을 말한다. 그래디언트 소실 및 폭주 문제를 해결하기 위한 다양한 기법이 제안되었으며, 그 가운데 대표적인 기법으로 가중치 초기화 기법 · 비선형 활성화 함수, batch normalization이 있다. [8]

가중치 초기화 기법과 비선형 활성화 함수는 각각 분류 모델의 가중치를 초기화 하는 방법과 모델에서 사용하는 비선형 활성화 함수를 의미한다. 현재 대표적으로 사용되는 가중치 초기화 방법으로 Xavier와 He가 있다. [9, 10], 또한 대표적 비활성화 함수로 S자 형태를 띄는 sigmoid 함수와 hyperbolic tangent 함수가 있고, 좌표평면의 원점에서 꺾이는 형태를 띄는 ReLU 함수와 ELU 함수가 있다. [11, 12]

5. 실험

본 논문에서는 정확한 딥 러닝 기반 분류 모델을 구축하기 위해 딥 러닝 기반 분류 모델의 파라미터를 튜닝하고, 다양한 딥 러닝 기법들을 적용하는 실험을 진행하였다. 각 파라미터 튜닝 실험은 가장 정확도가 높은 모델의 파라미터를 기준으로, 실험 대상 파라미터를 변경하는 방식으로 진행하였다. 이후 기존 연구에서 가장 정확한 분류모델로 밝혀진 SVM 모델과 본 연구에서 구축한 딥 러닝 기반 분류 모델을 비교 평가하였다. [6]

5.1 실험 환경

실험을 위해 보안 전문가에 의해 6개의 제작자 그룹으로 분류된 1,944 건의 악성코드 샘플을 수집하였다. 수집된 데이터의 각 제작자 그룹 별 샘플 수는 균등하지 않다. 가장 많은 악성코드 샘플을 갖는 제작자 그룹은 1,220개의 샘플을 가지고 있고, 가장 적은 악성코드 샘플을 갖는 그룹은 3개의 샘플을 가지고 있다.

수집한 악성코드 샘플들에 대해 2장에서 설명한 특징들을 추출하였다. 추출된 특징은 총 111,144개이며, 추출된 특징들을 이용하여 악성코드 샘플들을 각 특징의 보유 유무로써 표현하였다.

분류 모델을 평가하기 위해 stratified 5-fold cross-validation 방법을 사용하였고, 평가 척도로 정확도(accuracy), 정밀도(precision), 검출율(recall), F-measure를 사용하였다.

5.2 은닉층 수 및 정점 수 튜닝 실험

본 실험은 본 논문에서 사용한 악성코드 데이터에 맞는 은닉층 수 및 정점 수를 찾기 위한 실험이다. 본 실험에서는 은닉층 수는 2~4 사이의 값, 정점 수는 10~1000개 사이에서 값에 대해 튜닝을 진행했으며, 다음 [표1]은 본 실험의 대표적인 결과를 보여준다.

[표1] 은닉층 수 및 정점 수 튜닝 실험

	(10,10)	(1000, 100)	(100,100, 100)	(1000,1000, 1000,1000)
Accuracy	93.06%	94.96%	94.60%	93.93%
Precision	93.04%	94.88%	94.44%	93.96%
Recall	93.06%	94.96%	94.60%	93.93%
F-Measure	92.56%	94.82%	94.43%	93.85%

[표1]의 가로는 튜플 형태로 표현된 은닉층 수와 각 은닉층의 정점 수를 의미하며, 세로는 각 평가 척도를 의미한다. 본 실험의 환경과 본 논문에서 사용한 악성코드 데이터에서는 사용되는 은닉층 수와 각 은닉층의 정점수에 관계없이 높은 정확도를 보였으며, 은닉층 수가 2개, 각 은닉층의 정점 수가 1000 · 100개인 경우 평가 결과가 가장 좋음을 알 수 있다.

5.3 학습률 튜닝 실험

본 실험은 본 논문에서 사용한 악성코드 데이터에 맞는 학습률을 찾기 위한 실험이다. 본 실험에서는 학습률에 대해 0.1 ~ 0.0001 사이의 값으로 튜닝을 진행했으며, 다음 [표2]는 본 실험의 대표적인 실험 결과를 보여준다.

[표2] 학습률 튜닝 실험

	0.1	0.01	0.005	0.001	0.0001
Accuracy	93.37%	94.55%	94.96%	94.65%	94.70%
Precision	94.82%	94.51%	94.88%	94.61%	94.63%
Recall	93.37%	94.55%	94.96%	94.65%	94.70%
F-Measure	93.94%	94.43%	94.82%	94.54%	94.54%

[표2]의 가로는 학습률을 의미하며, 세로는 각 평가 척도를 의미한다. 본 실험의 환경과 본 논문에서 사용한 악성코드 데이터에서는 대부분의 파라미터에서 높은 정확도를 보였으며, 학습률이 0.005일 때 모델의 평가 결과가 가장 좋음을 확인할 수 있다.

5.4 가중치 초기화 및 활성화 함수 실험

본 실험은 본 논문에서 사용한 악성코드 데이터에 맞는 학습률을 찾기 위한 실험이다. S자 형태의 활성화 함수는 Xavier 초기화 방법과 같이 사용했을 때 효과적이며 좌표평면의 원점에서 꺾이는 형태의 활성화 함수는 He 초기화 방법과 같이 사용했을 때 효과적이다. 따라서 본 실험에서는 S자 형태의 활성화 함수인 Sigmoid 함수와 Xavier 초기화 방법을 사용한 경우, 좌표평면의 원점에서 꺾이는 형태의 활성화 함수인 ReLU · ELU와 He 초기화 방법을 사용한 경우에 대해 실험을 진행하였다. 다음 [표 3]은 가중치 초기화 및 활성화 함수에 따른 평가 결과를 보여준다.

[표3] 가중치 초기화 및 활성화 함수 실험

	Xavier+sig	He+ReLU	He+ELU
Accuracy	94.45%	94.60%	94.96%
Precision	94.33%	94.55%	94.88%
Recall	94.45%	94.60%	94.96%
F-Measure	94.27%	94.47%	94.82%

[표3]의 가로는 가중치 초기화 방법과 활성화 함수를 의미하며, 세로는 각 평가 척도를 의미한다. 본 실험의 환경과 본 논문에서 사용한 악성코드 데이터에서는 가중치 초기화 방법과 활성화 함수의 조합에 관계없이 대부분 높은 정확도를 보였으며, He 초기화 방법과 ELU 활성화 함수를 같이 사용한 경우가 평가 결과가 좋은 것을 알 수 있다. 대부분 높은 정확도를 보인 점은 본 논문에서 실험한 조합이 일반적으로 정확한 딥 러닝 기반 분류 모델을 구축하는데 사용되는 조합인 점과 본 논문에서 사용한 악성

코드 데이터에 기인한 결과이다.

5.5 Dropout 실험

본 실험은 본 논문에서 사용한 악성코드 데이터에 적합한 dropout 기법 사용 여부 및 강도를 찾기 위한 실험이다. Dropout을 사용하지 않은 경우 (1), dropout 사용한 경우의 사용 정도 (0.8 ~ 0.2)에 대해 실험을 진행하였으며 실험 결과는 아래 [표4]와 같다.

[표4] Dropout 실험

	1(off)	0.8	0.6	0.4	0.2
Accuracy	94.50%	94.65%	94.96%	94.70%	94.65%
Precision	94.48%	94.53%	94.88%	94.73%	94.57%
Recall	94.50%	94.65%	94.96%	94.70%	94.65%
F-Measure	94.39%	94.49%	94.82%	94.62%	94.52%

[표4]의 가로는 dropout 사용 여부 및 강도를 의미하며, 세로는 각 평가 척도를 의미한다. 본 실험의 환경과 본 논문에서 사용한 악성코드 데이터에서는 dropout 정도에 관계없이 대부분 높은 정확도를 보였으며, dropout정도를 0.6을 사용한 경우가 평가 결과가 가장 좋음을 알 수 있다.

5.6 Batch normalization 실험

본 실험은 본 논문에서 사용한 악성코드 데이터에 batch normalization 기법 사용 여부에 따른 모델 평가 결과를 확인하기 위한 실험이다. Batch normalization 사용 여부에 따라 모델을 평가한 결과는 [표5]와 같다.

[표5] Batch normalization 실험

	Batch_none	Batch
Accuracy	94.96%	94.45%
Precision	94.88%	94.50%
Recall	94.96%	94.45%
F-Measure	94.82%	94.35%

[표5]의 가로는 batch normalization 사용 여부를 의미하며, 세로는 각 평가 척도를 의미한다. 본 실험의 환경과 본 논문에서 사용한 악성코드 데이터에서는 batch normalization을 사용한 경우, 오히려 정확도가 떨어지는 것을 확인 할 수 있다. 이는 본 실험의 환경에서 사용된 은닉층의 수가 2개로 batch normalization을 통한 효과를 보기 어려운 점과 본 논문에 사용된 악성코드 데이터에 기인한 결과로 볼 수 있다.

5.7 분류 모델 비교 실험

본 실험은 본 논문에서 사용한 딥 러닝 기반 분류 모델, 기존에 악성코드 제작자 그룹 분류에서 가장 효과적인 것으로 나타난 SVM 모델을 비교하는 실험이다. 딥 러닝 기반 분류 모델의 파라미터로 은닉층 수는 2, 각 은닉층의 정점 수는 1000 · 100, 학습률은 0.005, 가중치 초기화 방법으로 He, 활성화 함수로 ELU, dropout을 0.6의 정도로 사용하였으며, batch normalization은 사용하지 않았다. SVM 모델의 파라미터로 사용한 커널은 RBF이며, c는 100, gamma는 0.01을 사용하였다. 이는 파라미터 튜닝을 통해 평가 결과가 가장 좋은 모델을 선정한 결과이다. 두 모델의 평가 결과는 아래 [표6]과 같다.

[표6] 분류 모델 비교 실험

	Deep	SVM
Accuracy	94.96%	93.42%
Precision	94.88%	93.32%
Recall	94.96%	93.42%
F-Measure	94.82%	93.12%

위 [표6]에 따라, 모든 평가 척도에서 딥 러닝 기반 분류 모델이 SVM 모델보다 1.54% 높은 정확도를 보임을 알 수 있다. 또한 앞선 파라미터 실험들과 비교했을 때, 대부분의 파라미터에서 SVM보다 딥 러닝을 이용한 분류 모델의 정확도가 높았다. 이는 악성코드 제작자 그룹 분류 문제에 딥 러닝 기반 분류 모델을 이용하면 기존 분류 모델들보다 정확도 향상을 기대할 수 있음을 의미한다.

6. 결론

본 논문에서는 악성코드 제작자 그룹 분류 문제에 딥 러닝 기반 분류 모델을 적용해 보고, 파라미터 튜닝 및 다양한 딥 러닝 기법을 통해 정확한 모델을 구축하는 진행하였다. 최종적으로 은닉층 수는 2개, 각 은닉층 수의 정점은 1000 · 100개, 학습률은 0.005, 가중치 초기화 방법은 He, 활성화 함수는 ELU, dropout 정도를 0.6으로 사용하였을 때, 가장 정확도가 94.96%로 가장 높았다. 이는 기존 연구에서 악성코드 제작자 그룹 분류 문제에서 가장 효과적이었던 SVM모델보다 1.54% 높은 수치이며, 파라미터 튜닝을 진행한 대부분의 파라미터에서도 SVM모델보다 정확도가 높았다. 따라서 보안 전문가들이 악성코드 제작자 그룹 분류를 위한 분류 모델을 구성할 때, 딥 러닝 기반 분류 모델을 이용하여 구성한다면 타 모델보다 좋은 모델을 구축할 수 있을 것이라 사료된다.

참고 문헌

[1] Islam R, Tian R, Batten LM, Versteeg S, "Review: Classification of malware based on integrated static and dynamic features", Journal of Network and Computer Applications, vol. 36, no. 2, pp. 646-656, 2013.

[2] Pai S, Troia FD, Visaggio CA, Austin TH, Stamp M, "Clustering for malware classification", Journal of Computer Virology and Hacking Techniques, vol. 13.2, pp. 95-107, 2017.

[3] Hassen M, Carvalho M, Chan P, "Malware classification using static analysis based features", IEEE Symposium Series on Computational Intelligence, 2017.

[4] Tian R, Islam R, Batten L, Versteeg S, "Differen -tiating malware from cleanware using behavioural analysis", In Proceedings of the 5th International Conference on Malicious and Unwanted Software, pp. 23-30, 2010.

[5] Alabdulmohsin I, Han Y, Shen Y, Zhang X, "Content-agnostic malware detection in hetero -geneous malicious distribution graph", In Proceed -ings of the 25th ACM International on Conference on Information and Knowledge Management, pp. 2395-2400, 2016

[6] Hong J, Park S, Kim S-W et al., "Classifying malwares for identification of author groups", Concurrency and Computation Practice and Experience, vol. 30, no. 3, 2018.

[7] Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R, “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”, Journal

of Machine Learning Research, vol. 15, pp. 1929-1958, 2014

[8] Loffe S, Szegedy C, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”, In Proceedings of the 32nd International Conference on Machine Learning, pp. 448-456, 2015

[9] Glorot X, Bengio Y, “Understanding the difficulty of training deep feedforward neural networks”, In Proceedings of the 13th International Conference on Artificial Intelligence and Statistics, pp. 249-256, 2010

[10] He K, Zhang X, Ren S, Sun J, “Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification”, In Proceedings of the 15th International Conference on Computer Vision, pp. 1026-1034, 2015

[11] Nair V, Hinton GE, “Rectified linear units improve restricted boltzmann machines”, In Proceedings 27th International Conference on Machine Learning, pp. 807-814, 2010

[12] Clevert D-A, Unterthiner T, Hochreiter S, “Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)”, arXiv preprint arXiv:1511.07289, 2015

홍 석 진

2015년 한국해양대학교 데이터정보학과 학사

2015년~현재 한양대학교 대학원 컴퓨터·소프트웨어학과 석사과정 재학

관심 분야 : 보안, 데이터 마이닝, 딥 러닝

홍 지 원

2009년 한양대학교 정보통신대학 컴퓨터전공 학사

2009년~현재 한양대학교 대학원 컴퓨터·소프트웨어학과 박사과정 재학

관심 분야: 사회연결망 분석, 보안, 데이터 마이닝

김 상 욱

1989년 서울대학교 컴퓨터공학과 학사

1991년 KAIST 전산학과 석사

1994년 KAIST 전산학과 박사

1991년~1991년 미국 Stanford University, Computer Science Department, 방문 연구원

1994년~1995년 KAIST 정보전자 연구소 전문 연구원

1999년~2000년 미국 IBM T.J. Watson Research Center, Post-Doc.

1995년~2003년 강원대학교 정보통신공학과 부교수

2003년~현재 한양대학교 공과대학 컴퓨터공학부 교수

2009년~2010년 미국 Carnegie Mellon University, Visiting Scholar

관심 분야: 데이터베이스 시스템, 데이터 마이닝, 멀티미디어 검색, 사회연결망 분석, 웹 데이터 분석, 딥 러닝

김 동 필

2005년 경북대학교 정보보호학과 석사

2009년 경북대학교 정보보호학과 박사

2009년~2012년 LG전자 MC연구소 선임연구원

2012년~현재 ETRI 부설연구소 선임연구원

관심 분야: 사이버보안, 빅 데이터

김 원 호

2004년 포항공과대학교 컴퓨터공학과 석사

2004년~현재 ETRI 부설연구소 실장

관심 분야: 사이버보안