악성코드 작성자 그룹 분류를 위한 특징 선택

Malware Feature Selection for Author Group Classification

홍지원(Jiwon Hong)[1] 박상현(Sanghyun Park)[2] 김상욱(Sang-Wook Kim)[3]

요 약

현대의 컴퓨터 보안의 가장 큰 위협 중 하나는 악성코드(malware)의 존재이다. 악성코드의 제작자들은 법망을 교묘히 피하며 지속적으로 악성코드를 제작하고 있다. 악성코드를 저자 그룹에 따라 분류하는 기술은 디지털 포렌식(digital forensic)에 유용한 정보를 제공할 수 있다. 본 논문에서는 악성코드로부터 추출한 다양한 특징 정보들로부터 저자 그룹 판별에 더 큰 도움이 되는 특징들을 선별해 내는 방안을 제안한다. 이러한 선별 작업의 결과 특징 추출 시간 및 분류 모델 학습과 분류에 소요되는 시간을 단축시킬 수 있었음을 실험을 통해 확인하였다. 또한 이에 따른 정확도 감소 역시 최소화됨을 증명하였다.

주제어: 악성코드, 분류, 특징 선택

1 한양대학교 컴퓨터·소프트웨어학과, 박사과정.
2 한양대학교 컴퓨터·소프트웨어학과, 석사과정.
3 한양대학교 컴퓨터·소프트웨어학과, 교수, 교신저자.
+ 이 논문은 2017년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원 (No. NRF-2017R1A2B3004581) 및 한국연구재단-차세대정보·컴퓨팅기술개발사업의 지원 (No. NRF-2017M3C4A7083678)을 받아 수행된 연구임.
+ 논문접수: 2018년 03월 16일, 심사완료: 2018년 04월 03일, 게재승인: 2018년 04월 11일.

Abstract

One of the greatest threats to modern cybersecurity is the existence of malware. The authors of malware are avoiding law enforcement and continuously producing new malware. Classification of malware by author groups can provide useful information for digital forensics. In this paper, we propose feature selection methods to identify more useful features for author group classification among a great number of features extracted from malware. As a result of the feature selection process, we confirm that the feature extraction time, classification model learning time, and classification time were significantly shortened. We also verify that decrease in accuracy is also minimized.

Keywords: Malware, classification, feature selection

1. 서 론

악성코드(malware)는 악의적인 목적을 가지고 제작된 컴퓨터 소프트웨어를 뜻한다. 악성코드들은 사용자의 정보를 탈취하여 금전적 피해를 끼치거나 컴퓨터 자원을 악의적인 용도로 사용하려는 목적을 지닌다. 컴퓨터 기술과 인터넷의 발달, IoT(Internet of Things)의 활성화로 네트워크에 연결된 컴퓨터의 수가 무수히 늘어나고 있어 이러한 컴퓨터들을 공격하기 위해 방대한 양의 악성코드가 매일 새롭게 등장하고 있다[1, 2, 3].

이와 같이 새롭게 등장하는 악성코드의 대부분은 기존 악성코드를 일부 수정한 변종이다[1, 2, 4, 5]. 공격자들은 단기간에 효율적으로 다수의 새로운 악성코드를 작성하기 위해 기존에 그들이 작성했던 코드나 공개된 코드를 일부만 수정하는 방식을 택한다. 이에 따라 같은 공격자 집단에 의해 작성된 악성코드들은 코드의 상당 부분에서 유사성을 보인다[1, 6, 7]. 이러한 유사성은 악성코드의 저자 그룹을 추정할 수 있는 근거가 될 수 있다. 이와 같은 저자 그룹 정보는 악성코드 사건에 대한 디지털 포렌식(digital forensic)에 큰 도움을 줄 수 있다.

악성코드의 저자 그룹 분류(classification)를 위해서 사용되는 분류 알고리즘들은 악성코드에서 추출한 다양한 특징(feature)들을 바탕으로 분류 모델을 학습하고 새로운 악성코드의 그룹을 예상해 낸다[6, 7]. 이 과정에서 사용되는 특징은 악성코드의 바이너리(binary)에서 추출하거나, 가상 환경에서 악성코드를 실제로 실행하여 그 행동으로부터 얻을 수 있다[1, 4, 6]. 더 정확한 분류를 위해 학계에서는 악성코드로부터 분류에 도움이 될 수 있는 더 많은 특징들을 추출하는 연구에 주력해왔다[1, 2, 3, 4].

대부분의 분류 알고리즘은 분류 대상이 가지는 특징들의 수에 그 학습 속도 및 예측 속도가 좌우된다. 특징이 많이 주어질수록 정확도가 향상될 가능성이 높지만, 악성코드로부터의 특징 추출, 분류 모델 학습, 분류 모델을 이용한 예측에 더 많은 시간이 소요된다. 반대로 아주 작은 수의 특징만을 이용한다면 특징 추출, 학습, 예측은 빠르게 수행되겠지만 정확도가 낮아질 수 있다. 즉, 사용되는 특징의 수에 대해 분류의 정확도와 속도는 일종의 트레이드-오프(trade-off) 관계에 있다고 할 수 있다. 조금이라도 더 정확한 분류를 위해 무수히 많은 특징들을 사용할 수 있지만 이 중 상당수는 분류 정확도에 크게 기여하지 않는 특징들일 수 있다. 특히 이미 어느 정도 높은 정확도에 도달해 있다면 추가적인 특징들은 정확도에 더 이상 기여하지 않을 가능성이 높다[6, 9].

본 논문에서는 악성코드 그룹 분류를 위해 사용될 수 있는 수많은 특징들 중 그룹 분류에 실제로 유용한 특징들을 추출하고, 이들만을 이용하여 모든 특징을 다 사용한 경우에 비해 크게 모자라지 않은 정확도를 보이는 분류 모델을 구축하여 분류 정확도와 속도 사이에 균형 잡힌 특징들을 선별(selection)하고자 한다. 이를 위해 악성코드로부터 추출된 분류에 유용한 특징들을 다양한 기존 특징 선택 방안을 활용하여 분석한 후 일부를 선별한다. 이후 실제 분류 실험을 통해 이들이 보이는 정확도를 확인한다.

본 논문의 2장에서는 특징 선택 기법을 이용한 분류 방안에 대해 설명한다. 3장에서는 악성코드 데이터로부터 추출하는 특징 정보들에 대해 서술한다. 4장은 본 연구에서 적용한 특징 선택 기법들에 대하여 설명한다. 5장은 실험에 사용하는 악성코드 데이터 집합과 그 클래스 정의에 대해 서술하고, 특징 선택 실험과 그 결과에 대해 논한다. 6장에서 본 논문을 매듭짓는다.

2. 특징 선택 기법을 이용한 분류

기존의 악성코드 작성자 그룹 분류 연구에서는 정확한 분류를 위해 최대한 다양한 특징들을 추출하여

사용하였다[6, 7]. 그러나 사용하는 특징의 수가 많아지면 분류 작업을 수행하기 위해 소요되는 시간이 증가하는 문제점이 존재한다. 이 시간은 크게 악성코드로부터 해당 특징들을 추출하는 데에 소요되는 시간, 분류 모델을 학습하는 데에 소요되는 시간, 그리고 분류 모델을 이용하여 예측하는 데에 소요되는 시간으로 나뉜다. 기존 연구에서 주로 높은 정확도를 보이는 분류 기법은 SVM(Support Vector Machine)으로, 해당 기법에서는 예측 시간은 특징들의 수에 크게 영향을 받지 않으나, 학습에 소요되는 시간이 큰 부분을 차지한다.

이러한 문제를 해결하기 위해서는 수많은 특징들 중 분류 정확도에 영향을 미치는 주요 특징들만을 선별하여 분류 시 사용되는 특징의 수를 줄이는 것이 중요하다. 본 논문에서는 악성코드의 작성자 그룹 분류를 다음과 같은 단계로 수행할 것을 제안한다. (1) 수집된 악성코드로부터 작성자 그룹 분류에 이용될 수 있는 특징들을 최대한 다양하게 추출한다. (2) 추출된 특징들에 대해 특징 선택 기법을 적용하여 일부를 선별한다. (3) 특징 선택 기법을 통해 선별된 특징들을 이용하여 분류 모델을 학습시킨다. (4) 새로운 악성코드가 발견되었을 때, 학습된 모델을 이용하여 해당 악성코드의 저자 그룹을 예측한다.

3. 특징 추출

악성코드의 분류를 위해서는 각 클래스를 구분해 줄 수 있는 유용한 특징들이 필요하다. 본 연구에서는 악성코드 제작자 그룹 분류를 위해서 제작자들이 악성코드를 작성할 시 본인들의 기존 코드를 재사용함에 주목하였다. 이러한 재사용이 드러나는 특징들을 각 악성코드 샘플에 대한 정적(static) 및 동적(dynamic) 분석[6, 7, 8]을 통해 추출하였다.

악성코드의 핵심 파일들은 일반적으로 실행 가능한 바이너리 형태로 저장된다. 이러한 바이너리에 대해 정적 및 동적 분석을 수행할 수 있다. 정적 분석은 해당 바이너리 파일을 직접적으로 분석한다. 해당 파일을 역어셈블(disassemble)하거나 바이너리 내의 인식 가능한 문자열(string)을 탐색하는 등의 분석이 포함된다. 동적 분석은 해당 바이너리 파일을 가상 머신이나 외부와의 연결이 차단된 컴퓨터에서 실제로 실행하여 실행 중 나타나는 악성코드의 행동을 분석한다. 네트워크 사용 내역이나 파일 사용 내역 등이 이에 해당한다. 본 논문에서는 정적 및 동적 분석의 결과 다음과 같이 재사용과 연관이 있을 수 있는 특징들을 추출하였다.

3.1 정적 분석

3.1.1. 문자열(STRINGS)

악성코드의 바이너리가 포함하는 모든 출력 가능한 문자열들을 이용한다. 이들 중 상당수는 의미 없는 문자열이지만, 동일한 저자 그룹에서 습관적으로 반복 사용한 문자열이 존재하거나, 상징적으로 이용하는 특정 문자열이 포함되는 경우가 많아 재사용 탐지에 유용하다.

3.1.2. 임포트(IMPORTS) 및 익스포트(EXPORTS)

대부분의 악성코드는 하나의 온전한 실행 파일만을 가지지 않는다. 이들은 외부 라이브러리를 호출하거나 해당 파일이 가진 함수가 악성코드의 다른 부분에서 사용될 수 있도록 하는 함수 인터페이스를 지닌다. 이와 같이 외부로 노출된 정보를 재사용 탐지를 위한 특징으로 이용할 수 있다.

3.1.3. 함수(FUNCTIONS)

악성코드를 역어셈블하여 얻은 코드로부터 각 함수가 가지는 OP코드(opcode)들을 특징으로 사용할 수 있다. 순서가 보존된 각 함수의 OP코드 리스트는

해당 함수가 새로운 악성코드에서 재사용되었을 경우에 동일한 값을 가지게 될 것이다. 이러한 함수의 OP코드 리스트는 길이가 가변적이고 용량이 크기 때문에 해시 함수(hash function)를 이용하여 비교적 짧은 고정 길이 값으로 변환하여 사용한다.

3.1.4. 함수 블록(FUNCTION BLOCKS)

함수 특징의 경우 함수 전체가 재사용되어야만 동일한 값을 갖는 악성코드가 등장한다. 함수의 일부분이 변형되었을 때에도 재사용을 탐지할 수 있도록 하기 위해 함수의 OP코드들 중 실행 순서를 조작하는 코드(예를 들어, JUMP 등)를 구분자로 하여 함수들을 더 작은 단위로 쪼갠 함수 블록 특징을 이용한다. 이 역시 함수 특징과 동일하게 해시 값(hash value)만을 취하여 사용한다.

3.2. 동적 분석

3.2.1. 뮤텍스(MUTEXES)

악성 코드의 저자들은 자신들이 배포한 비슷한 역할의 악성코드들이 하나의 컴퓨터 내에서 중복 실행되는 것을 막는 등의 목적을 위해 상호 배재 뮤텍스를 사용할 수 있다. 즉, 같은 저자 그룹 내에서 유사한 악성코드들 간에 동일한 뮤텍스 이름을 공유할 가능성이 있다. 이를 특징으로 사용한다.

3.2.2. 네트워크 활동(NETWORKS)

주어진 악성코드가 요청한 네트워크 요청 패킷(packet)들을 분석하여 DNS, IP 주소 등을 특징으로 사용한다. 동일한 저자가 유사한 목적으로 제작한 악성코드들은 이러한 네트워크 주소들을 공유할 가능성이 높다. 이러한 정보 또한 특징으로 사용한다.

3.2.3. 파일(FILES)

일부 악성코드는 침투한 컴퓨터의 파일시스템(filesystem)에 존재하는 파일을 읽거나 변형, 혹은 새롭게 작성하는 행동을 보인다. 이는 악성코드가 자신을 숨기려는 시도이거나 컴퓨터를 직접 공격하는 행위일 수 있다. 이 때 대상이 되는 파일시스템 상의 주소는 기존에 시스템 파일이 존재하는 곳을 가리키거나 기존 시스템 파일과 유사한 주소인 경우가 많다. 이는 악성 코드가 생성한 파일을 사용자가 중요한 시스템 파일로 오해하기를 바라는 악성코드 제작자의 테크닉이다. 해당 주소는 시스템 파일의 이름과 유사하도록 제작자가 직접 지정하기 때문에 같은 저자 그룹에 의해 개발된 악성코드는 이러한 주소 역시 공유할 가능성이 높다.

3.2.4. 드롭(DROPS)

드롭은 악성코드가 실행 가능한 바이너리 내의 임의 영역에 적재된 파일을 추출하여 파일시스템에 저장하거나, 인터넷을 통해 파일을 다운받는 등의 행위를 뜻한다. 본 연구에서는 해당 파일의 저장 경로, 원격지 주소, 파일 해시 값 등을 특징으로 활용한다.

3.2.5. 레지스트리 키(KEYS)

마이크로소프트 윈도우(Microsoft Windows)는 운영 체제 기능이나 응용 프로그램들의 설정을 저장하는 데에 레지스트리(registry)라는 일종의 데이터베이스를 활용한다. 많은 수의 악성코드들이 레지스트리를 악용하여 공격을 수행한다. 본 연구에서는 주어진 악성코드가 접근하는 모든 레지스트리 키에 대한 정보를 특징으로 활용한다.

3.2.6. API 호출(API CALLS)

악성코드를 구성하는 코드 중 상당 부분은 대상 운영 체제(operating system)의 시스템 API 호출로 이루어져 있다. 본 연구에서는 각 악성코드에서 호출한 시스템 API들의 목록을 특징으로 활용한다.

4. 특징 선택

분류를 위해서는 분류의 대상이 되는 객체를 나타내는 특징들을 추출한 후, 그 특징들의 나열로 해당 객체를 표현할 필요가 있다. 일반적으로 특징의 개수가 많을수록 높은 분류 정확도를 보이게 된다. 그러나 원하는 분류의 목적에 따라서는 분류 정확도를 향상시키는 데에 효용성이 크지 않거나 아예 의미가 없는 특징들이 존재할 수 있다. 실생활의 예를 들자면 고등학교 학생들을 성적에 따른 클래스로 분류하고자 할 때 키나 성별이 큰 의미를 가지지 않는 것과 같다. 또한, 사용되는 특징의 수가 많아질수록 특징 추출, 분류 모델의 학습 및 분류에 들어가는 시간이 커지게 된다.

따라서 원하는 분류 목적에 적합하여 분류 정확도를 향상시키는 데에 도움이 되는 특징들만을 선별하여 특징의 수를 적당한 수준으로 유지할 필요가 있다. 이를 위해 사용되는 기법이 특징 선택(feature selection)이다. 본 연구에서는 기존 악성코드 그룹 분류 연구에서 수많은 특징들이 사용됨에 따라 분류와 관련된 작업들을 수행하는 데에 시간이 많이 들거나 고차원의 특징 벡터를 저장하는 데에 있어 메모리 소요량이 큰 문제를 해결하기 위해 특징 선택 기법을 활용한다. 특징 선택 기술은 주어진 특징 집합에서 가장 분류와 관련성이 높은 특징 집합을 찾아내는 기술이다.

특징 선택 기술은 분류와 관련되지 않은 특징이나 유사하거나 중복된 의미를 갖는 특징들이 특징 집합에 선택되는 것을 최소화하고 분류와 깊은 관련을 지닌 특징들을 최대화하여 특징 집합을 찾아낸다. 이를 통해 분류 정확도에 영향을 크게 주는 특징들만 선택한다면 특징 추출 및 분류모델 생성시간을 경감하는 효과를 기대할 수 있다.

특징 선택을 위한 기존 알고리즘들은 크게 필터(filter)와 래퍼(wrapper) 방식으로 나뉜다[9] 필터 방식은 각각의 특징들을 평가하는 척도를 도입하여 각 특징들을 평가하고, 해당 척도 순으로 특징들을 정렬한 후 상위 k개의 특징들만을 선택한다. 특징 평가의 척도로는 일반적으로 information gain이 많이 사용된다[9].

래퍼 방식은 실제로 분류에 사용할 분류 알고리즘을 이용하여 특징 집합을 평가하는 방식이다[9]. 래퍼 방식의 평가를 위해서는 최초에 주어진 전체 특징 집합으로부터 더 작은 부분집합들을 생성할 필요가 있다. 예를 들어 모든 가능한 부분집합을 각각 평가하는 완전 탐색(exhaustive search) 방식을 사용할 수 있다. 그러나 완전 탐색을 위해서는 $2n$개의 부분집합을 평가해야 하기 때문에 매우 많은 시간이 걸린다. 이를 완화하기 위해 일반적으로 best-first 탐색, hill climbing 탐색 등의 방법으로 부분집합들을 top-down 혹은 bottom-up 방식으로 순차적으로 생성하여 정해진 개수의 부분집합을 평가한 후 그 중에서 가장 높은 정확도를 보이는 방법[7]을 이용한다. 래퍼 방식으로 각 부분집합을 평가할 때마다 해당 특징 부분집합을 이용하여 분류 모델을 학습하기 때문에 평가하는 부분집합의 개수가 늘어남에 따라 전체적인 특징 선택 과정에 매우 긴 시간이 소요될 수 있다. 그러나 실제로 분류에 사용할 분류 알고리즘을 이용하여 특징 집합을 평가하기 때문에 높은 정확도를 보이는 특징이 있다.

본 논문에서는 필터 방법은 물론 래퍼 방법도 함께 사용하여 특징 조합을 평가하고 각각이 악성코드의 저자 그룹 분류에 어떠한 영향을 끼치는지 분석한다.

5. 실험

본 논문에서는 기존의 악성코드 저자 그룹 분류 연구[6, 7]에서 더 나아가 특징 선택 기법을 적용하여 분류와 관련된 시간을 줄이거나 분류 정확도를

향상시킬 수 있는지 판단하기 위해 다양한 실험을 수행하였다.

5.1. 클래스 정의 및 데이터 집합

본 절에서는 실험에서 사용된 데이터 집합과 그 클래스 레이블(class label), 그리고 그 특징들을 설명한다.

클래스의 정의는 분류의 목적에 따라 달라질 수 있다. 본 연구에서는 악성코드를 제작한 공격자 그룹의 유사성을 바탕으로 해당 분야의 전문가들이 직접 정의한 악성 코드 제작자 클래스 집합을 사용한다. 또한, 주어진 각 악성코드 객체의 클래스 레이블 역시 동일한 전문가들에 의해 매겨진다.

실험을 위해 보안 전문가에 의해 9개의 공격자 그룹으로 분류된 약 2,300 건의 악성코드 샘플을 수집하였다. 각 그룹은 평균적으로 약 250 건의 악성코드로 이루어져 있으나 그 분포는 많게는 약 1,500 건, 적게는 5건 내외로 큰 차이를 보인다. 기존의 유사한 연구에서도 실제 발생 사례의 비율과 유사한 비율로 샘플이 수집된 바 있다[6, 7].

5.2. 실험 환경

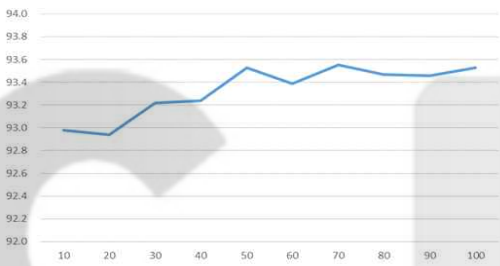
수집한 데이터 집합으로부터 3장에서 설명한 특징들을 추출하였다. 전체 데이터에서 추출한 특징의 수는 약 30만 개이다. 특징 선택 기법 중 필터 방식의 특징 평가 척도로는 information gain을 사용한다. 래퍼 방식에서는 hill climbing 탐색을 이용하여 bottom-up으로 부분집합을 생성한다. 실제 분류 및 래퍼 방식에서의 평가를 위한 분류 모델을 생성하기 위해 기존 연구에서 가장 높은 정확도를 보이는 SVM 알고리즘을 이용한다[1]. 분류 모델을 평가하기 위해서 5-fold cross-validation [9] 방법을 사용하며, 분류 정확도는 F_1-score[9]로 나타낸다.

실험을 위해 사용된 컴퓨터는 Intel Core i7 CPU를 탑재하였으며 32GB의 가용 메모리를 가졌다.

5.3. 필터 방법을 이용한 특징 선택

본 실험에서는 필터 방식의 특징 선택 기법을 적용하여 악성 코드 그룹을 분류하고 그 정확도를 평가하였다. 특징들은 개별적으로 information gain으로 평가된 후 점수가 높은 순서로 정렬하고, 상위 일부 부분집합에 대해 SVM으로 분류 모델을 생성하여 그 정확도 및 모델 생성 시간과 특징 선택에 들어간 시간을 평가하였다.

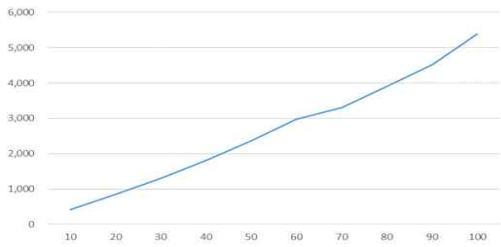
그림 1은 선택된 상위 일부 특징이 사용되었을 때의 분류 정확도를 나타내는 그래프이다. 세로축은



[그림 1] 필터 방식 특징 선택의 분류 정확도

분류 결과의 정확도인 F_1-score를 나타내며, 가로축은 특징의 개수이다. 그래프의 각 점은 info-rmation gain이 높은 순으로 정렬된 특징 목록에서 최상위로부터 전체 특징 개수의 10%씩 증가시키며 선택된 특징 집합을 나타낸다. 그림 1에 따르면 전체 특징들 중 information gain 기준 상위 10%의 특징들만 선택해도 F_1-score는 약 0.55%만 감소함을 알 수 있었다. 특히, 상위 50%가 선택된 경우에는 전체 특징들을 사용했을 때와 거의 동일한 정확도를 보인다. 상위 70%의 특징들이 사용되었을 때에는 정확도가 0.02% 증가한 것을 알 수 있다. 이는 정확한 분류를 혼란시키는 일부 특징들이 제거됨에 따라 특징의 수가 감소되었음에도 분류 정확도는 오히려 증가한 것으로 추정된다.

그림 2는 선택된 상위 일부 특징이 사용되었을 때의 분류 모델 생성 시간을 나타내는 그래프이다. 세



[그림 2] 필터 방식 특징 선택의 모델 구축 시간

로 축은 분류 모델의 생성에 들어간 시간을 나타내며, 가로 축은 특징의 개수이다. 그래프의 각 점은 그림 2와 동일하다. 그림 2에 따르면 모델의 생성 시간은 특징의 개수에 대해 거의 일차 함수적으로 비례한다. 모든 경우 특징 선택에 소요된 시간은 120초로, 모델 생성 시간에 비해 적은 시간만이 소요되었다. 특징 선택으로 특징의 수를 줄였을 때 정확도는 거의 영향을 받지 않은 반면 모델 생성 시간은 크게 달라지므로 특징 선택의 효과가 크다는 것이 입증된다. 예를 들어, 상위 50%의 특징만 선택할 경우 특징 선택 시간을 포함하더라도 모델 생성 시간은 절반 이하로 줄어들지만 정확도는 거의 동일하다. 상위 10%만 선택하더라도 분류 정확도는 0.55%만 감소하지만 생성 시간은 1/10 이하로 감소하는 결과를 보인다. 표 1은 그림 1과 그림 2의 결과를 하나의 표로 정리한 것이다.

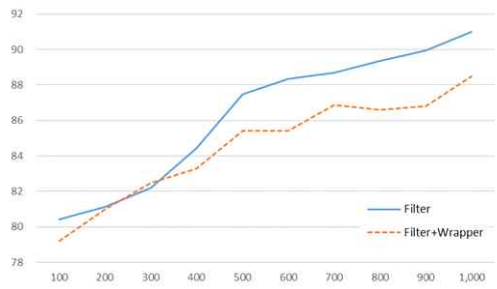
5.4. 래퍼 방법을 이용한 특징 선택

본 실험에서는 래퍼 방법을 이용한 특징 선택의 정확도를 평가하였다. 래퍼 방법은 bottom-up hill-climbing 탐색을 이용하여 매 단계별로 후보 특징 부분집합을 생성하고 SVM을 이용한 분류 모델로 정확도를 평가한 뒤, 각 단계에서 가장 좋은 후보 특징 부분집합을 다음 단계에서 확장하여 후보 특정 부분집합을 생성하는 방식으로 진행하였다. 각 단계에서 생성된 후보 특징 부분집합 중 이전 단계에서의 분류 정확도를 넘는 분류 정확도를 보이는 것이 하나도 없다면 특징 선택 과정은 해당 단계에서 종료한다.

이 때, 매 단계별로 후보 특징 부분집합이 (전체 특징 수 n) − (단계 s) 개 생성되기 때문에, 전체 특징을 이용하여 후보 특징 부분집합을 생성할 경우 SVM을 이용한 분류 모델 구축에 너무 긴 시간이 소요되는 문제가 있었다. 이 문제를 완화하기 위해 본 실험에서는 information gain 기반 필터 방법을 이용하여 상위 k개의 특징들을 먼저 골라낸 후, k개의 특징들에 대해서만 bottom-up hill-climbing 탐색으로 후보 특징 부분집합을 생성하도록 하였다.

[표 1] 필터 방법을 사용한 특징 선택

선택된 특징 수 (%)	F_1-score (%)	모델 학습 시간 (초)	특징 선택 시간 (초)
100	93.53	5,390	0
90	93.46	4,515	120
80	93.47	3,901	120
70	93.55	3,309	120
60	93.39	2,977	120
50	93.53	2,368	120
40	93.24	1,801	120
30	93.22	1,301	120
20	92.94	844	120
10	92.98	415	120



[그림 3] 필터 및 필터+래퍼 방식 특징 선택의 정확도

그림 3은 필터 방식으로 선별한 최상위 k개의 특징들에 대해 래퍼 방식으로 특징 선택을 수행했을 경우의 분류 정확도를 나타내는 그래프이다. 가로축은 선택된 필터 방식으로 선택된 특징들의 수를 나타내며 100씩 증가한다. 세로 축은 F_1-score를 나타낸다. Filter 선은 필터 방식만 적용했을 때의 정확도를 나타내며, Filter+Wrapper 선은 필터 방식을 적용한 뒤 래퍼 방식으로 추가적으로 선별한 특징들만을 이용했을 때의 정확도를 나타낸다. 표 2는 그림 3의 결과를 표로 나타낸 것이다. Filter+Wrapper 방식에서 최종적으로 선별된 특징들의 수는 표 2에서 확인할 수 있다. 단순히 필터 방식의 특징 선택만 이용했을 때에 비해 Filter+Wrapper 방식은 특징 수를 극단적으로 줄이면서도 정확도는 거의 유지할 수 있었다.

특히, 필터 방식에서 상위 300개의 특징을 사용했을 때와 해당 특징들로부터 추가적으로 래퍼 방식으로 27개의 특징만을 추출하여 사용했을 때는 27개의 특징을 사용한 것이 오히려 300개의 특징을 사용한 것보다 조금 더 높은 정확도를 보였다. 이러한 결과는 다수의 주요 특징들 중에서 특정 조합을 사용했을 경우에 전체 특징을 사용했을 경우에 비해 분류 정확도 역시 향상시킬 수 있음을 의미한다.

6. 결론

본 논문에서는 악성코드에 대한 정적 및 동적 분석을 통해 추출한 다양한 특징 정보들을 바탕으로 악성 코드의 저자 그룹을 판별하는 데에 있어 특징 선택 기법을 적용함으로써 더 빠른 분류 모델 구축 및 클래스 예측이 가능하도록 하는 연구에 대해 논하였다. 실험을 통해 필터 방식의 특징 선택 기법을 적용하면 특징 추출, 분류 모델 학습 및 클래스 예측 과정에 걸리는 시간을 크게 단축하면서도 정확도를 거의 일정하게 유지할 수 있음을 확인하였으며, 래퍼 방식의 특징 선택 기법을 이용하면 더 높은 정확도를 얻을 수 있음 역시 확인하였다.

래퍼 방법을 이용하면 아주 적은 수의 특징만으로도 높은 정확도로 악성코드의 저자 그룹을 분류해 내는 것이 가능하지만 이러한 특징들만을 이용하기 위해 래퍼 방법을 사용하는 것이 항상 전체적인 처리 시간을 단축시키지는 않을 것으로 예상된다. 이는 래퍼 방법으로 특징을 선택하는 데에 드는 시간이 전체 특징을 이용하여 분류 모델을 학습하는 데에 드는 시간보다 길기 때문이다. 그러나 이와 같이 특징 선택 기법을 이용하여 선별된 특징들은 지속적으로 새로운 분류에 활용할 수 있으며, 보안 전문가들이 악성코드를 분석하는 과정에서도 도움이 될 것으로 사료된다.

[표 2] 래퍼 방식을 이용한 특징 선택

Filter	선택된 특징 수 (개)	100	200	300	400	500	600	700	800	900	1,000
	F1-score (%)	80.43	81.14	82.2	84.43	87.49	88.35	88.69	89.35	89.96	90.99
Filter+Wrapper	선택된 특징 수 (개)	7	14	27	17	22	27	41	20	30	28
	F1-score (%)	79.20	81.00	82.50	83.30	85.40	85.40	86.90	86.60	86.80	88.50

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, et al., "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] Hong J, Park S, Kim S-W, "On exploiting static and dynamic features in malware classification", In Proceedings of the 7th EAI International Conference on Big Data Technologies and Applications, pp. 122-129, 2016.

[8] Chae D-K, Ha J, Kim S-W, et al., "Credible, resilient, and scalable detection of software plagiarism using authority histograms", Knowledge-Based System, vol. 95, pp. 114-124, 2016.

[9] Han J, Pei J, Kamber M, Data Mining: Concepts and Techniques, Morgan Kaufmann, 2011.

홍 지 원

2009년 한양대학교 정보통신대학 컴퓨터전공 학사

2009년~현재 한양대학교 대학원 컴퓨터·소프트웨어학과 박사과정 재학

관심 분야: 사회연결망 분석, 보안, 데이터 마이닝

박 상 현

2016년 선문대학교 공과대학 컴퓨터공학 학사

2018년 한양대학교 대학원 컴퓨터·소프트웨어학과 석사

관심 분야: 사회연결망 분석, 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