JKSCI

한국컴퓨터정보학회논문지
Journal of The Korea Society of Computer and Information
Vol. 26 No. 11, pp. 41-49, November 2021
https://doi.org/10.9708/jksci.2021.26.11.041

Dimensionality Reduction of Feature Set for API Call based Android Malware Classification

Hee-Jin Hwang*, Soojin Lee*

*Student, Dept. of Computer Science and Engineering, Korea National Defense University, Nonsan, Korea
*Professor, Dept. of Computer Science and Engineering, Korea National Defense University, Nonsan, Korea

[Abstract]

All application programs, including malware, call the Application Programming Interface (API) upon execution. Recently, using those characteristics, attempts to detect and classify malware based on API Call information have been actively studied. However, datasets containing API Call information require a large amount of computational cost and processing time. In addition, information that does not significantly affect the classification of malware may affect the classification accuracy of the learning model. Therefore, in this paper, we propose a method of extracting a essential feature set after reducing the dimensionality of API Call information by applying various feature selection methods. We used CICAndMal2020, a recently announced Android malware dataset, for the experiment. After extracting the essential feature set through various feature selection methods, Android malware classification was conducted using CNN (Convolutional Neural Network) and the results were analyzed. The results showed that the selected feature set or weight priority varies according to the feature selection methods. And, in the case of binary classification, malware was classified with 97% accuracy even if the feature set was reduced to 15% of the total size. In the case of multiclass classification, an average accuracy of 83% was achieved while reducing the feature set to 8% of the total size.

▸ **Key words**: API-Call, Feature Selection, Dimensionality Reduction, Malware Classification, CNN

[요 약]

악성코드를 포함한 모든 응용프로그램은 실행 시 API(Application Programming Interface)를 호출한다. 최근에는 이러한 특성을 활용하여 API Call 정보를 기반으로 악성코드를 탐지하고 분류하는 접근방법이 많은 관심을 받고 있다. 그러나 API Call 정보를 포함하는 데이터세트는 그 양이 방대하여 많은 계산 비용과 처리시간이 필요하다. 또한, 악성코드 분류에 큰 영향을 미치지 않는 정보들이 학습모델의 분류 정확도에 영향을 미칠 수도 있다. 이에 본 논문에서는 다양한 특성 선택(feature selection) 방법을 적용하여 API Call 정보에 대한 차원을 축소시킨 후, 핵심 특성 집합을 추출하는 방안을 제시한다. 실험은 최근 발표된 안드로이드 악성코드 데이터세트인 CICAndMal2020을 이용하였다. 다양한 특성 선택 방법으로 핵심 특성 집합을 추출한 후 CNN(Convolutional Neural Network)을 이용하여 안드로이드 악성코드 분류를 시도하고 결과를 분석하였다. 그 결과 특성 선택 알고리즘에 따라 선택되는 특성 집합이나 가중치 우선순위가 달라짐을 확인하였다. 그리고 이진분류의 경우 특성 집합을 전체 크기의 15% 크기로 줄이더라도 97% 수준의 정확도로 악성코드를 분류하였다. 다중분류의 경우에는 최대 8% 이하의 크기로 특성 집합을 줄이면서도 평균 83%의 정확도를 달성하였다.

▸ **주제어**: API-Call, 특성 선택, 차원 축소, 악성코드 분류, CNN

• First Author: Hee-Jin Hwang, Corresponding Author: Soojin Lee
*Hee-Jin Hwang (judgment216@naver.com), Dept. of Computer Science and Engineering, Korea National Defense University
*Soojin Lee (cyberkma@gmail.com), Dept. of Computer Science and Engineering, Korea National Defense University
• Received: 2021. 10. 25, Revised: 2021. 11. 11, Accepted: 2021. 11. 12.

http://www.ksci.re.kr pISSN:1598-849X | eISSN:2383-9945

I. Introduction

전 세계 웹 트래픽을 분석하는 Statcounter에 따르면 2021년 9월 기준으로 전 세계 모바일 운영체제 시장에서 안드로이드가 차지하는 비율은 72.44%이며, 한국에서는 72.2%를 차지하고 있다[1-2]. 이러한 현상이 나타나는 주요인은 무엇보다 안드로이드 특유의 개방성이 여러 사람들의 접근을 용이하게 해주기 때문이다. 그러나 공격자의 입장에서는 악성코드를 유포하기에 최적의 운영체제이기 때문에 안드로이드를 통한 모바일 악성코드 유포도 계속 증가하고 있다.

한편 안드로이드 운영체제는 어떤 응용프로그램을 동작시키기 위해 미리 정의된 API(Application Programming Interface)를 호출하는데, 이러한 동작을 API 호출(API Call)이라고 한다. 악성코드 또한 API를 호출하기 때문에, API 호출을 통해 악성코드가 안드로이드에서 어떻게 실행되어 동작하는지 결정할 수 있다. 즉 API Call을 활용한 분석은 악성코드 탐지에 있어 결정적인 역할을 할 수 있다.

악성코드 분석가들은 다양한 정보를 수집하여 악성코드 분석을 수행한다. 일반적으로 알려진 악성코드 분석 방법에는 정적 분석과 동적 분석이 있다. 정적 분석은 프로그램의 종류, 크기, 헤더 정보, 내부 문자열, 등록 정보, 디지털 인증서 등을 분석하며, 이러한 분석을 위해 프로그램을 실행하지 않아도 되는 장점을 가진다. 반면 동적 분석은 프로그램을 직접 실행시켜 동작을 관찰하면서 내부 메모리 등의 변화를 통해 악성코드를 분석한다. 정적 분석에 비해 정확한 분석이 가능하나 시스템 감염 등 악성코드에 의한 피해가 발생할 수 있다.

악성코드 분석 방법이 이와 같이 두 가지 형태로 구분되기는 하지만, 일반적으로 어느 한 가지 방법만을 활용해 분석하는 방법보다는 두 가지 방법을 적절하게 조합하여 분석하는 것이 더 좋은 것으로 알려져 있다. 그러나 API Call과 같은 정보가 대량으로 포함된 데이터를 분석해야 하는 정적 분석의 경우 분석 작업에 장시간이 소요되며, 동적 분석의 경우에는 실제 시스템 감염의 피해가 이어질 수 있다는 점에 유의해야 한다.

악성코드 분석을 위해 수집된 데이터가 많은 특성을 가지고 있다는 점은 악성코드에 대한 정확한 분석이 가능할 수 있음을 의미한다. 그러나 대부분의 값이 '0'이거나 동일한 값을 다수 포함하고 있으면 적절한 분석이 제한된다. 예를 들어, 본 연구에서 사용한 CICAndMal2020의 경우 총 9,503개의 특성이 존재하며, 대부분의 값이 '0' 또는 '1'로만 표시되어 있다. 단일 악성코드를 선택하여 확인해 보면 일반적인 악성코드가 가지는 특성과 해당 악성코드가 호출하는 일부 API 관련 정보를 제외한 9,400여 개 이상의 정보가 '0'으로 표시되어 있다. 따라서 전체 특성을 이용하여 분석을 수행할 경우 효율성이 현저하게 떨어지며, 일반적인 분석 환경에서는 시간과 자원의 소모가 획기적으로 증가하여 신속한 분석이 제한될 수밖에 없다.

독일 보안업체 GData의 통계자료[3]에 따르면 2018년 새롭게 등장한 안드로이드 악성코드의 수는 약 300만 개에 달하며, 이는 약 10초마다 새로운 악성코드가 1개씩 등장하였음을 의미한다. 이러한 상황에서 안드로이드 악성코드의 분석에 소요되는 시간과 자원을 줄이지 못할 경우 나날이 등장하고 있는 악성코드에 대한 신속한 대응은 불가능해질 것이다.

이상과 같은 필요성에 따라 본 논문에서는 다양한 특성 선택(feature selection) 방법을 적용하여 API Call 정보에 대한 차원을 축소시킨 후, 핵심 특성 집합(feature set)을 추출하는 방안을 제시한다. 최근 발표된 안드로이드 악성코드 데이터세트인 CICAndMal2020을 대상으로 핵심 특성 집합을 추출한 후 CNN(Convolutional Neural Network)을 이용하여 안드로이드 악성코드 분류를 시도하고 결과를 분석한다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 API Call 정보를 이용하여 안드로이드 악성코드 분류를 시도했던 기존 연구들을 살펴본다. Ⅲ장에서는 제안하는 접근방법과 이용한 데이터세트에 대해 설명하고, Ⅳ장에서는 학습 데이터세트의 구성을 설명한 후 실험 결과를 분석한다. 마지막으로 Ⅴ장에서 연구 결과를 요약하고, 결론을 맺는다.

II. Preliminaries

Abir Rahali 등[4]은 본 연구의 실험에서 사용하는 CICAndMal2020 데이터세트를 직접 생성하였다. 그리고 Extra-Tree Classifier를 이용하여 API Call을 포함한 9,503개의 특성을 2,200여 개로 줄인 후 CNN을 적용해 악성코드 탐지성능을 분석하였다. 악성코드와 정상 파일을 분류하는 이진 분류는 실시하지 않았으며, 악성코드 카테고리를 분류하는 다중분류의 정확도는 82.22%로 나타났다. 그러나 제시된 방법론에 대한 구체적인 설명이 부족하며, 일반적인 실험환경이 아닌 고성능의 환경(50개의 CPU와 512GBytes의 메모리가 장착된 고성능 컴퓨팅 환경)에서 이뤄진 실험이기에 현실에서의 적용이 힘들다. 또한, 다른 특성 선택 방법들과의 비교가 제시되지 않았다.

Naser Peiravian 등[5]은 안드로이드 Manifest에서 추출한 권한 특성과 클래스 파일에서 추출한 API Call 특성을 결합하여 Bagging Classifier를 통한 이진 분류를 시도하였으며, 정확도는 96.39%로 확인되었다.

Lucky Onwuzurike 등[6]은 API Call 순서를 마르코프 체인을 기반으로 모델링하여 안드로이드 악성코드를 탐지할 수 있는 MaMaDroid를 제시하였다. 최초 F1-Score는 0.99였으며, 학습된 지 1년이 지난 후에도 F1-Score는 0.86을 유지하는 것으로 확인되었다.

정적 분석과 더불어 동적 분석에 대한 연구도 진행되고 있다. AndreaDe Lorenzo 등[7]은 안드로이드 응용프로그램에서 동작하는 악성 프로그램에 대한 동적 분석 결과를 색상으로 시각화하여 나타내는 VizMal이란 도구를 제시하였다. David Sean Keyes 등[8]은 CICAndMal2020 데이터세트를 가상환경에서 실행시킨 후 동적 분석을 통해 141개의 특성을 추출하고, Decision Tree Classifier를 이용하여 악성코드를 분류한 결과 Precision은 0.984, Recall은 0.983을 달성하였다.

정적 분석과 동적 분석을 결합하여 악성코드 분류를 시도하는 연구도 진행되었다. Suleiman Y. Yerima 등[9]은 정적 API 시퀀스와 동적 API 시퀀스 간 관계를 조사하고 상호 관계와 융합을 통해 악성코드를 분류하는 MalDA 프레임워크를 제안하였으며, 분류 정확도는 94.39%로 나타났다. Laya Taheri 등[10]은 CICAndMal2017 데이터세트의 초기 버전에 API Call 정보를 포함시킨 데이터세트를 제시하였다. 그리고 해당 데이터세트를 대상으로 정적 분석 기반 이진 분류와 동적 분석 기반 카테고리 및 패밀리 분류를 시도하여 각각 95.3%, 83.3%, 59.7%의 정확도를 달성하였다. 이상에서 살펴본 API Call 정보를 기반으로 한 안드로이드 악성코드 분류 연구들의 핵심 접근방법과 성능지표를 요약한 결과는 Table 1에서 보는 바와 같다.

Table 1. Summary of Previous Studies

Analysis	Ref.	Main Approach	Classification	Performance
Static	[4]	Feature selection by Extra-Tree Classifier	Multiclass	Accuracy : 82.22%
	[5]	Classification by Premission+API Call	Binary	Accuracy : 96.39%
	[6]	Classification by Markov Chain	Binary	F1-Score : 0.99
Dynamic	[7]	Visualize information with colors	·	·
	[8]	Dynamic Extraction and Classification by Decision-Tree	Binary	Precision : 0.984 Recall : 0.983
Static+ Dynamic	[9]	Correlate static API sequences and dynamic API sequences	Binary	Accuracy : 94.39%
	[10]	Binary classification by static analysis Multiclass classification by dynamic analysis	Binary	Accuracy : 95.3%
			Multiclass(C)	Accuracy : 83.3%
			Multiclass(F)	Accuracy : 59.7%

III. The Proposed Scheme

본 연구에서는 제안하는 접근방법은 Fig. 1에서 보는 바와 같다.

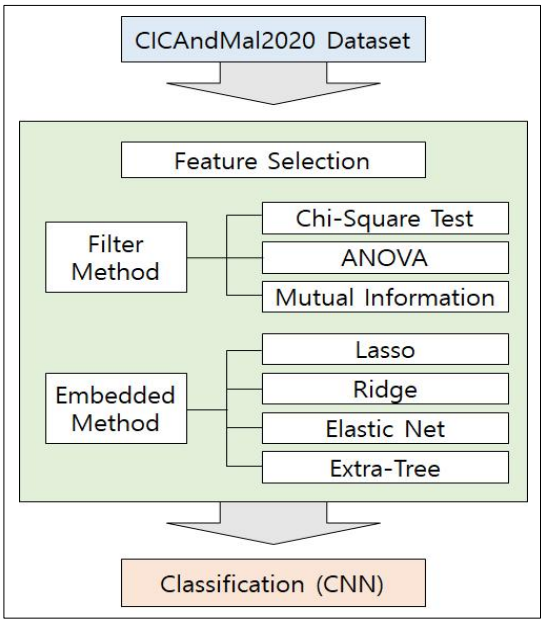

Fig. 1. Overview of Proposed Approach

전술한 바와 같이 API Call 정보를 포함하고 있는 CICAndMal2020 데이터세트는 특성 정보의 차원이 크기 때문에, 일차적으로 기존에 사용되어 왔던 특성 선택 방법들을 사용하여 차원을 줄이면서 핵심 특성 집합만을 추출한다. 이어서 추출된 특성 집합을 이미지로 변환하고 CNN을 기반으로 악성코드 분류를 시도한다.

1. Dataset

CICAndMal2020은 캐나다 사이버보안센터와 사이버보안연구소가 함께 제작한 데이터세트로 안드로이드 악성

코드를 포함하는 앱을 이용하여 총 195,623개의 악성코드를 수집하였다. 세부 구성은 Table 2에서 보는 바와 같다.

Table 2. Number of Malware in CICAndMal2020 Dataset

Class	Number of samples
Adware	47,210
Backdoor	1,538
FileInfector	669
No_Category	2,295
PUA	2,051
Ransomware	6,202
Riskware	97,349
Scareware	1,556
Trojan	13,559
Trojan_Banker	887
Trojan_Dropper	2,302
Trojan_SMS	3,125
Trojan_Spy	3,540
Zero_day	13,340

Malware는 총 14개의 카테고리로 구분되어 있다. 그 중 No_Category는 다른 13개의 카테고리에 비해 특성을 분류하기 어렵기에 별도의 카테고리로 지정되었으며, Zero_day에 속하는 악성코드는 하위 패밀리가 너무 다양하여 Zero_day만의 고유 특성 파악이 어렵다. 따라서, 본 연구에서는 이 2개의 카테고리를 제외한 12개의 카테고리만을 활용하여 실험을 진행하였다.

정상 파일(Benign)의 경우에는 Malware의 데이터 수와 균형을 맞추기 위해 Androzoo(https://androzoo.uni.lu) 데이터 162,901개를 수집하였으며, 세부 정보는 Table 3에서 보는 바와 같다. 5개의 서브 데이터세트로 구분되어 있기는 하지만 수집 시기에 차이가 있을 뿐 분류 자체가 큰 의미를 가지는 것은 아니다.

Table 3. Number of Benign Dataset

Class	Number of samples
Ben0	32,804
Ben1	47,861
Ben2	42,635
Ben3	7,847
Ben4	31,754

2. Feature Selection

본 연구에서 사용된 CICandMal2020 데이터세트는 총 9,503개의 특성 정보를 가지고 있으며, 그 중 API Call과 관련된 대부분의 정보는 '0'의 값을 가진다. 따라서 정보의 양에 비해 실제 악성코드 분류에 활용되는 특성 정보는 많지 않다. 또한 9,503개의 특성 정보를 모두 분석하고자 시도할 경우 기존 연구에서처럼 50개의 CPU와 500GByte의 RAM을 갖춘 컴퓨팅 환경을 구축하지 않는 한 실험의 정상적인 진행이 불가능하다.

이러한 문제를 해결하기 위해 본 연구에서는 기존 많은 연구에서 활용되었던 특성 선택 알고리즘을 적용하여 특성 정보의 차원 축소를 최우선으로 수행하였다. 특성 선택이란 데이터를 분석하는 과정에서 소요되는 계산량과 대상이 되는 데이터의 차원을 줄여 예측 성능을 향상시키는 방법이다. 특성 선택의 주목적은 불필요한 변수의 영향을 줄이면서 좋은 예측 성능을 제공할 변수의 하위집합을 선택하는데 것이다.

특성 선택은 크게 Filter Method, Wrapper Method 및 Embedded Method 3가지 방법이 있다[11]. Filter Method는 결과 변수와의 상관관계 같은 통계량이나 척도에 기반하여 모든 특성에 순위를 정하여 선택하는 방법이며, 대표적으로 Chi-Square Test[12], Analysis of Variance (ANOVA)[13], Mutual Information[14]의 3가지 알고리즘이 있다. 본 연구에서는 이 3가지 알고리즘을 모두 사용하여 선택된 특성 수와 CNN 모델의 탐지성능을 비교하였다.

Wrapper Method는 정확도 측면에서 가장 좋은 특성들의 부분집합을 찾는 방법으로, Forward Selection[15]과 Backward Elimination[16] 알고리즘이 있다. 다만, Wrapper Method는 모든 특성을 여러 부분집합으로 만들면서 부분집합에 대한 정확도를 이용해 비교하는 과정을 수행해야 하기에, 특성의 수가 많은 경우에는 다양한 조합을 만들어 정확도를 도출하기까지 시간이 과도하게 소요된다는 단점을 가진다. 따라서 본 연구에서는 Wrapper Method를 제외하고 실험을 진행하여 결과를 비교하였다.

Embedded Method는 모든 특성을 학습함으로써 가장 좋은 특성을 선택하며, 본 연구에서는 Lasso[17], Ridge[18], Elastic Net[19], Extra-Tree Classifier[20]의 4가지 알고리즘을 사용하였다. 그러나 Lasso, Ridge 및 Elastic Net의 경우 회귀 문제에 적합했던 방법들이기 때문에 이진 분류에만 적용할 수 있어 다중분류 실험은 진행하지 않았다.

3. Deep Learning

특성 선택을 통해 선정된 특성 정보는 이미지로 변형된 후 CNN에 입력된다. 실험에서 사용된 CNN 모델은 3개의 컨볼루션 계층(Convolution Layer), 3개의 맥스풀링 계층(Maxpooling Layer), 2개의 완전 연결 계층(Fully-Connected Layer)을 가진다. 출력층의 활성화 함수는 소프트맥스(Softmax), Optimizer는 Adam을 사용하였다. 세부적인 신경망 구조는 Table 4에서 보는 바와 같다.

Table 4. CNN Model Structure

#	Layer	Option	Activation Function
1	Convolution 1	Filter = 32, Size = 3 Dropout = 0.6	ReLu
2	MaxPooling 1	Size = 2	/
3	Convolution 2	Filter = 64, Size = 3 Dropout = 0.6	ReLu
4	MaxPooling 2	Size = 2	/
5	Convolution 3	Filter = 128, Size = 3 Dropout = 0.3	ReLu
6	MaxPooling 3	Size = 2	/
7	Flatten	-	-
8	FC 1	Neurons = 256 Dropout = 0.3	ReLu
9	FC 2	Neurons = The number of class	Softmax

IV. Results

본 섹션에서는 모든 특성 선택 방법을 사용하였을 때, 선택되어지는 특성의 수를 비교하고, 이를 활용한 악성코드 탐지성능을 비교한 실험 결과를 정리한다.

1. Environment

모든 실험은 Windows 10 Home 64bit 운영체제, Intel(R) Core(TM) i7-10700 CPU, RAM 16GB 사양의 PC에서 진행하였다. 개발언어는 Python 3.7.10 버전이며, 특성 추출 알고리즘은 Python의 Scikit Learn 라이브러리를 사용하였다.

2. Experimental Dataset

실험은 Benign과 Malware를 구분하는 이진 분류, 12개의 Malware 카테고리를 분류하는 다중분류로 구분하여 진행하였다.

이진 분류를 위한 서브 데이터세트는 다음과 같이 구성하였다. 먼저 Benign은 5개의 클래스에서 각각 1,400개씩 총 7,000개를 추출하였다. Malware는 가장 작은 수를 가지는 카테고리인 'FileInfector'(669개)를 기준으로 삼아 No_Category와 Zero_day 클래스를 제외한 12개의 클래스에서 각각 600개씩 총 7,200개를 추출하였다. 그리고 이러한 서브 데이터세트 추출 과정은 총 20회를 반복하여 수행하였다. 즉 원본 데이터세트에 포함된 많은 데이터가 실험에 사용될 수 있도록 함으로써 특정 데이터세트에 국한된 성능분석이 아니라 대부분 데이터에 적용 가능한 성능분석이 될 수 있도록 노력하였다. 추출된 서브 데이터세트는 다시 Train 서브 데이터세트와 Test 서브 데이터세트로 구분하였으며, 그 비율은 선행 연구[4]를 참고하여 8:2의 비율로 구성하였다.

한편 특성 선택을 위해 사용할 서브 데이터세트는 학습 및 성능 검증을 위해 사용하는 서브 데이터세트를 추출한 이후에 별도의 임의 추출 과정을 거쳐 동일한 수의 데이터를 추출하였다. 이러한 과정은 특성 선택 과정에서 사용한 데이터세트와 학습 및 성능 검증 과정에서 사용한 데이터세트가 동일할 경우 발생할 수 있는 과적합(overfitting)을 방지하기 위한 조치이다. 다만 Table 2에서 확인할 수 있는 바와 같이 'FileInfector'와 'Trojan_Banker' 2개의 클래스는 그 수가 다른 클래스에 비해 작기 때문에, 특성 선택과 학습 및 성능 검증에 동일한 서브 데이터세트를 사용하였다.

3. Binary Classification

Filter Method 및 Embedded Method를 적용하여 서브 데이터세트를 대상으로 특성 선택을 실시하고, 새롭게 구성된 학습된 서브 데이터세트를 대상으로 선택된 특성 집합만을 학습시켜 Benign과 Malware로 분류하는 이진 분류를 실시하였다. 실험 결과는 Table 5에서 보는 바와 같으며, 특성 선택 및 학습에 소요된 총 시간은 Table 6에서 보는 바와 같다.

Table 5. Results of Binary Classification

Method		Selected feature	Accuracy
Filter Method	Chi -Square	1,496	96.97
	ANOVA	1,496	97.09
	Mutual	101	96.74
Embedded Method	Lasso	188	97.07
	Ridge	763	96.61
	Elastic	221	96.92
	Extra -Tree	1,256	96.46

Table 6. Spending Time of Binary Classification

Method		Selection	Train
Filter Method	Chi -Square	1:02	20:05
	ANOVA	1:02	32:32
	Mutual	9:42	47:58
Embedded Method	Lasso	0:03	26:15
	Ridge	0:18	20:21
	Elastic	2:08	30:05
	Extra -Tree	1:42	30:07

특성 선택 과정에서는 9,503개의 특성 중에서 최대 15% 규모의 특성만이 선택되었지만, 97% 수준의 정확도로 이진 분류가 수행됨을 확인하였다. 그리고 Filter Method에 비해 Embedded Method는 작은 수의 특성만 사용하더라도 우수한 분류성능을 달성할 수 있었다. 이진분류에 대한 Confusion Matrix는 Fig. 2에서 확인할 수 있다.

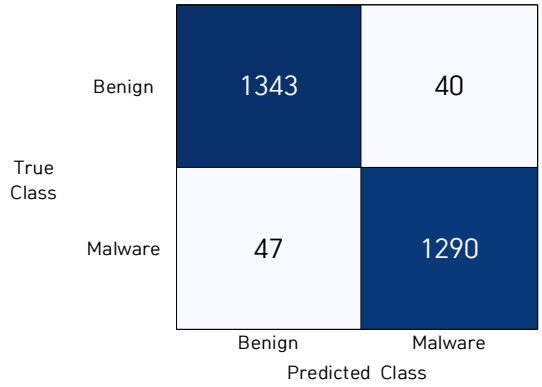

Fig. 2. Confusion matrix of Binary Classification (Feature Selection : Chi-Square Test)

4. Category Classification

앞서 사용한 데이터 중 Benign을 제거한 Malware를 각각의 카테고리로 분류하는 다중분류 실험도 함께 진행하였다. 실험 결과는 Table 7에서 보는 바와 같으며, 특성 선택 및 학습에 소요된 시간은 Table 8에서 보는 바와 같이 나타났다.

Table 7. Result of Category Classification

Feature Selection		Selected features	Accuracy
Filter Method	Chi -Square	781	83.07
	ANOVA	781	82.61
	Mutual	593	83.40
Embedded Method [4]	Extra -Tree	2,237	82.22

Table 8. Spending Time of Category Classification

Feature Selection		Selection	Train
Filter	Chi -Square	0:09	15:37
	ANOVA	0:08	13:24
	Mutual	5:24	13:19
Embedded Method [4]	Extra -Tree	Unknown	

Table 7과 Table 8에 명시된 Extra-Tree Classifier는 기존 연구[4]에서 확인된 사항을 참고하여 작성하였다.

실험 결과 9,503개의 특성 중 최대 8% 규모의 특성만이 특성 집합으로 선택되었다. 그러나 23.5%의 특성을 사용하였음에도 다중분류의 Accuracy가 82.22%에 그쳤던 기존 연구보다 조금 향상된 83%의 Accuracy를 달성하였다. Chi-Square Test 알고리즘을 사용하여 특성을 선택한 후 카테고리 다중분류를 실시한 결과에 대한 Confusion Matrix는 Fig. 3에서 확인할 수 있다.

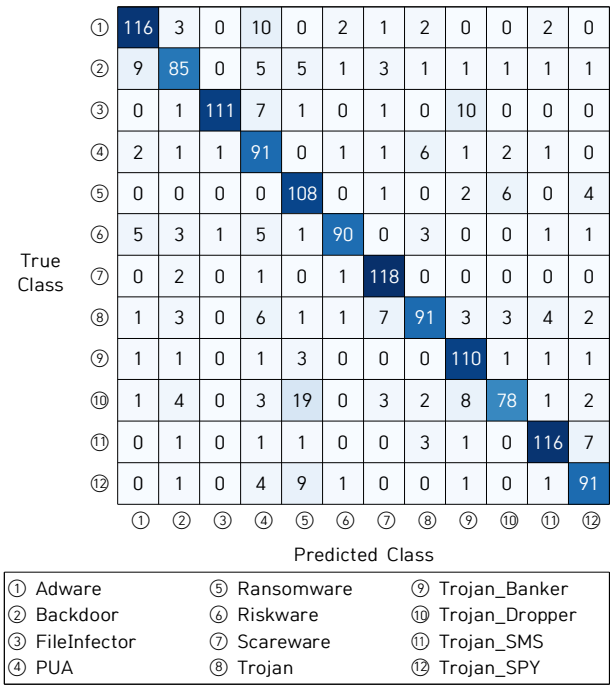

Fig. 3. A Confusion matrix of Category Classification (Feature Selection : Chi-Square Test)

5. Analysis of Selected Feature

일반적으로 활용되는 특성 선택 알고리즘을 적용하여 CICAndMal2020 데이터세트가 포함하고 있는 API Call 특성 정보에 대한 차원 축소를 시도한 결과, 이진분류의 경우 기존 9,503개의 특성 중 최대 1,508개의 특성이 선택되었고, 다중분류의 경우 최대 781개의 특성만 선택되었다. 이렇게 선택된 특성을 이용하여 CNN 모델을 학습시킨 후 분류성능을 검증한 결과 기존 연구보다 더 나은 정확도로 분류가 가능함을 확인하였다.

이러한 결과 중 Filter Method의 Chi-Square와 ANOVA를 통해 선택된 특성의 수가 같지만 실제로 선택된 특성은 확연하게 구분된다. Table 9는 이진분류 및 다중분류를 위해 선택된 특성 집합의 예를 보여주고 있다.

Table 9. Features selected from Chi-Square and ANOVA

#	Binary		Category	
	Chi -Square	ANOVA	Chi -Square	ANOVA
1	0	49	0	3394
2	7	3334	3	4827
3	1	50	4	323
4	3	36	7	3458
5	8	65	5	137
		⋮		
100	8386	8386	129	262
		⋮		
500	4093	9188	2938	183
		⋮		
730	4032	8418	3465	3465
		⋮		
1000	4408	1541	-	-
		⋮		

가장 왼쪽의 숫자(#)는 특성 선택 알고리즘으로 선정된 특성들의 가중치 순위를 의미하며, 내부의 숫자는 특성의 번호를 의미한다. CICAndMal2020 데이터세트는 모든 특성을 번호로 표시하고 있으며, 별도의 특성 명칭은 식별이 불가능하여 본 연구에서도 번호로만 표시하였다.

표에 나타난 바와 같이 Chi-Square Test 알고리즘을 통해 이진분류를 위해 선정된 특성 중 가장 가중치가 높은 특성은 0번 특성이며, ANOVA 알고리즘은 49번 특성을 가장 가중치가 높은 특성으로 선택하였다. 이진분류를 위해 선택된 1,496개의 특성 중 가중치 순위가 일치하는 특성은 존재하지 않았다.

다중분류에서는 좀 더 유의미한 결과가 확인되었다. Chi-Square Test 알고리즘이 다중분류를 위해 선택한 특성 중 가중치 순위가 높은 특성은 대부분 악성코드의 네트워크 속성과 관련된 특성이다. 반면 ANOVA 알고리즘은 API Call과 관련된 특성을 가중치가 높은 특성으로 선택하였다. 즉, 다중분류를 위해 Chi-Square Test 알고리즘은 네트워크 측면의 특성을 우선적으로 고려하지만, ANOVA 알고리즘은 API Call 정보를 먼저 고려한다는 점을 확인할 수 있다.

이상과 같이 동일한 수의 특성 집합이 선택되었더라도 그 세부 내용이 달라지는 이유는 Chi-Square Test 알고리즘과 ANOVA 알고리즘이 특성을 선택하는 기준이 다르기 때문이다. Chi-Square Test 알고리즘은 특성 선택을 위해 연속된 두 데이터 간의 비교를 수행하지만, ANOVA 알고리즘은 범주형 데이터에서 각 데이터가 얼마나 영향력을 미치는지를 비교한다.

V. Conclusions

CICAndMal2020 데이터세트를 비롯하여 API Call과 관련된 정보를 포함하고 있는 정적 분석 데이터는 대부분 특성 정보의 차원이 매우 크다. 따라서 특성 정보를 일반적인 시스템 환경에서 분석할 경우 과도한 시간과 자원을 필요로 하기 때문에 차원 축소가 반드시 선행되어야 한다. 이러한 문제를 해결하기 위해 본 논문에서는 API Call 정보가 가지는 차원의 크기를 일반적인 특성 선택 알고리즘을 통해 줄인 후 안드로이드 악성코드 분류를 시도하였다.

특성 선택 알고리즘을 사용하여 선택된 적은 수의 특성 집합을 CNN 기반의 모델에 학습시킨 후 악성코드 분류성능을 측정한 결과, 적의 수의 특성 집합만으로도 일정한 수준의 악성코드 분류성능을 달성할 수 있었다. 이진분류의 경우 특성 집합을 전체 크기의 15% 크기로, 카테고리 다중분류의 경우 최대 8% 이하의 크기로 줄여도 동일한 데이터세트를 활용한 기존 연구[4]보다 더욱더 효율적이며 높은 성능을 확인할 수 있었다. 또한, 특성 선택 알고리즘에 따라 선택되는 특성 집합이나 가중치 우선순위도 달라짐을 확인하였다.

그러나 특성 선택 방법 중 하나인 Wrapper Method의 경우 데이터의 특성 정보로 구성된 집합들 중 가장 적절한 부분집합을 찾는 방법이기에 시간이 과도하게 소모되는 문제가 발생하여 특성 선택 및 악성코드 분류성능을 확인하지 못하였다. 향후에는 일반적인 컴퓨팅 환경에서 Wrapper Method를 통한 특성 선택을 진행하여 3가지 방법에 대한 비교를 진행하고, Wrapper Method를 다른 특성 선택 방법과 결합하여 특성 집합의 차원을 줄이며 시간을 단축시키는 연구를 진행할 예정이며, 나아가 선택된 특성들의 정보를 분석하여 실제 악성코드 동작에 영향을 미치는 특성, 즉 어떤 API Call이 악성코드 분류에 큰 영향을 미치는지를 보다 정확하게 분석하고자 한다.

REFERENCES

[1] Statcounter, Mobile Operating System Market Share Worldwide, https://gs.statcounter.com/os-market-share/mobile/ worldwide/#monthly-202009-202109

[2] Statcounter, Mobile Operating System Market Share Worldwide, https://gs.statcounter.com/os-market-share/mobile/ south-korea/#monthly-202009-202109

[3] GDATA, Some 343 new Android malware samples every hour in 2017, https://www.gdatasoftware.com/blog/2018/02/30491 -so

me-343-new-android-malware-samples-every-hour-in-2017

[4] Abir Rahali, Arash Habibi Lashkari, Gurdip Kaur, Laya Taheri, Francois Gagnon, and Frédéric Massicotte, "DIDroid: Android Malware Classification and Characterization Using Deep Image Learning", 10th International Conference on Communication and Network Security (ICCNS2020), Pages 70-82, Tokyo, Japan, November 2020, DOI: 10.1145/3442520.3442522

[5] Naser Peiravian and Xingquan Zhu, "Machine Learning for Android Malware Detection Using Permission and API Calls", 2013 IEEE 25th International Conference on Tools with Artificial Intelligence, Pages 300-305, Herndon, VA, USA, February 2014, DOI: 10.1109/ICTAI.2013.53

[6] Lucky Onwuzurike, Enrico Mariconti, Panagiotis Andriotis, Emiliano De Cristofaro, Gordon Ross and Gianluca Stringhini, "MaMaDroid: Detecting Android Malware by Building Markov Chains of Behavioral Models (Extended Version)", ACM Transactions on Privacy and Security, Volume 22, Issue 2, Article No.: 14, pp 1-34, April 2019, DOI: 10.1145/3313391

[7] AndreaDe Lorenzo, FabioMartinelli, Eric Medvet, Francesco Mercaldo and Antonella Santone, "Visualizing the outcome of dynamic analysis of Android malware with VizMal", Journal of Information Security and Applications, Volume 50, February 2020, DOI: 10.1016/j.jisa.2019.102423

[8] David Sean Keyes, Beiqi Li, Gurdip Kaur; Arash Habibi Lashkari, Francois Gagnon and Frédéric Massicotte, "EntropLyzer: Android Malware Classification and Characterization Using Entropy Analysis of Dynamic Characteristics", 2021 Reconciling Data Analytics, Automation, Privacy, and Security: A Big Data Challenge (RDAAPS), pp 1-8, Hamilton, ON, Canada, June 2021, DOI: 10.1109/RDAAPS48126.2021.9452002

[9] Suleiman Y. Yerima and Mohammed K. Alzaylaee, "Machine learning-based dynamic analysis of Android apps with improved code coverage", EURASIP Journal on Information Security 2019, Article No.: 4, April 2019, DOI: 10.1186/s13635-019-0087-1

[10] Laya Taheri, Andi Fitriah Abdul kadir and Arash Habibi Lashkari, "Extensible Android Malware Detection and Family Classification Using Network-Flows and API-Calls", 2019 International Carnahan Conference on Security Technology (ICCST)), pp 1-8, Chennai, India, October 2019, DOI: 10.1109/CCST.2019.8888430

[11] Zebari, R., Abdulazeez, A., Zeebaree, D., Zebari, D. and Saeed, J. "A comprehensive review of dimensionality reduction techniques for feature selection and feature extraction.", Journal of Applied Science and Technology Trends Vol 1, No. 2, pp.56-70, May 2020. DOI: 10.38094/ jastt1224

[12] I. Sumaiya Thaseen, Ch. Aswani Kumar and Amir Ahmad, "Integrated Intrusion Detection Model Using Chi-Square Feature Selection and Ensemble of Classifiers", JArabian Journal for Science and Engineering, Volume 44, Issue 4, Pages 3357-3368, August 2018, DOI: 10.1007/s13369-018-3507-5

[13] B Jyothi Kumar, H Naveen, B Praveen Kumar, Sai Shyam Sharma and Jaime Villegas, "Logistic regression for polymorphic malware detection using ANOVA F-test", 2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), pp 1-34, Coimbatore, India, April 2019, DOI: 10.1109/ICIIECS.2017.8275880

[14] N.Hoquea, D.K.Bhattacharyyaa and J.K.Kalitab, "MIFS-ND: A mutual information-based feature selection method", Expert Systems with Applications, Volume 41, Issue 14, Pages 6371-6385, October 2014, DOI: 10.1016/j.eswa.2014.04.019

[15] MatthiasReifa and FaisalShafaitb, "Efficient feature size reduction via predictive forward selection", Pattern Recognition, Volume 47, Issue 4, Pages 1664-1673, April 2014, DOI: 10.1016/j.patcog.2013.10.009

[16] Hoai Bach Nguyen, Bing Xue, Ivy Liu and Mengjie Zhang, "Filter based backward elimination in wrapper based PSO for feature selection in classification", 2014 IEEE Congress on Evolutionary Computation (CEC), Page(s):3111 - 3118, Beijing, China, September 2014, DOI: 10.1109/CEC.2014.6900657

[17] R Muthukrishnan and R Rohini, "LASSO: A feature selection technique in predictive modeling for machine learning", 2016 IEEE International Conference on Advances in Computer Applications (ICACA), Page(s):18 - 20, Coimbatore, India, March 2017, DOI: 10.1109/ICACA.2016.7887916

[18] Saurabh Paul and Petros Drineas, "Feature Selection for Ridge Regression with Provable Guarantees", Neural Computation, Volume: 28, Issue: 4, Page(s): 716 - 742, April 2016, DOI: 10.1162/NECO_a_00816

[19] Hui Zou and Trevor Hastie, "Regularization and variable selection via the elastic net", Journal of the Royal Statistical Society Series B (Statistical Methodology), Volume: 67, Issue: 2, Pages: 301-320, March 2005, DOI: 10.1111/j.1467-9868.2005.00503.x

[20] Howida Abubaker, Aida Ali, Siti Mariyam Shamsuddin and Shafaatunnur Hassan, "Exploring permissions in android applications using ensemble-based extra tree feature selection" Indonesian Journal of Electrical Engineering and Computer Science, Vol 19, No 1, Pages: 543-552, July 2020, DOI: 10.11591/ijeecs.v19.i1.pp543-552

Authors

Hee-Jin Hwang received the B.S., degree in Mathematics from Keimyung University, Deagu Korea, in 2013. He is currently a M.S. Student in Department of Computer Science and Engineering from Korea National Defense University, Nonsan, Korea. He is interested in Cyber warfare and Cyber Security, Artificial Intelligence, Machine Learning.

Soojin Lee received the B.S., degree from Korea Military Academy in 1992, M.S. degree in Computer Science from Yonsei University in 1996, and Ph.D. degree in Computer Science from Korea Advanced Institute of Science and Technology in 2006. Dr. Lee is currently a Professor in the Departmant of Computer Science and Engineering at Korea National Defense University, Nonsan, Korea, from 2006. He is interested in National Cyber Security and Cyber warfare, Intrusion Detection System, Mobile Network Security, Encryption theory and applications.