Journal of Convergence for Information Technology
Vol. 12. No. 3, pp. 17-24, 2022
e-ISSN 2586-4440
DOI : https://doi.org/10.22156/CS4SMB.2022.12.03.017

선별된 특성 정보를 이용한 안드로이드 악성 앱 탐지 연구

명상준[1], 김강석[2*]
[1]아주대학교 정보통신대학원 사이버보안전공 학생, [2]아주대학교 사이버보안학과 교수

A Study on Android Malware Detection using Selected Features

Sangjoon Myeong[1], Kangseok Kim[2*]
[1]Student, Graduate School of Information and Communication Technology, Ajou University
[2]Professor, Department of Cyber Security, Ajou University

요 약 모바일 악성 앱이 급증하고 있으며, 전 세계 모바일 OS 시장의 대부분을 차지하고 있는 안드로이드가 모바일 사이버 보안 위협의 주요 대상이 되고 있다. 따라서 빠르게 진화하는 악성 앱에 대응하기 위해 인공지능 구현기술 중 하나인 기계학습을 활용한 악성 앱 탐지 기법의 필요성이 대두되고 있다. 본 논문은 악성 앱의 탐지 성능을 향상할 수 있는 특성 선택 및 특성 추출을 이용한 특성 선별 방법을 제안하였다. 특성 선별 과정에서 특성 개수에 따라 탐지 성능이 향상되었으며, 권한보다 API가 상대적으로 좋은 탐지 성능을 보였고, 두 특성을 조합하면 평균 93% 이상의 높은 탐지 정밀도를 보여 적절한 특성의 조합이 탐지 성능을 높일 수 있음을 확인하였다.

주제어 : 안드로이드 악성 앱, 기계학습, 특성 선택, 특성 추출, 정보 보안

Abstract Mobile malicious apps are increasing rapidly, and Android, which accounts for most of the global mobile OS market, is becoming a major target of mobile cyber security threats. Therefore, in order to cope with rapidly evolving malicious apps, there is a need for detection techniques of malicious apps using machine learning, one of artificial intelligence implementation technologies. In this paper, we propose a selected feature method using feature selection and feature extraction that can improve the detection performance of malicious apps. In the feature selection process, the detection performance improved according to the number of features, and the API showed relatively better detection performance than the permission. Also combining the two characteristics showed high precision of over 93% on average, confirming that the appropriate combination of characteristics could improve the detection performance.

Key Words : Android Malware, Machine Learning, Feature Selection, Feature Extraction, Information Security

1. 서론

최근 소프트웨어, 하드웨어 및 5G를 넘어 6G [1] 와 같은 네트워크 등의 진화와 함께 글로벌 스마트폰 시장이 지속적으로 성장하고 있으며, 이에 따른 모바일 앱 시장의 성장 또한 가속화되고 있다. 이러한 성장 속에 수많은 앱이 사용됨에 따라 스마트폰의 사용자를 노리는 악의적 기능을 포함한 악성 앱 또한 증가하고 있다. 2020년 4분기 모바일 악성 앱 탐지 건수는 4천만여 건이고 이 중 3백만여 건이 신규 악성 앱이었다[2].

스탯카운터(Statcounter)에 따르면 전 세계 모바일 운영 체제(Operation System)의 시장 점유율은 안드

*This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT: Ministry of Science and ICT) (No. NRF-2019R1F1A1059036).
*This article is extended and excerpted from Master Thesis.
*Corresponding Author : Kangseok Kim(kangskim@ajou.ac.kr)
Received January 10, 2022 Revised February 28, 2022
Accepted March 20, 2022 Published March 28, 2022

로이드(Android)가 71.09%, iOS가 28.21%를 차지하고 있어 모바일 분야 사이버 위협의 많은 부분이 안드로이드 OS를 대상으로 하고 있다[3].

이렇듯 안드로이드 악성 앱은 꾸준히 증가하고 있으며 대응책으로 주로 패턴 기반으로 악성 앱을 탐지하고 차단하고 있다. 패턴 기반은 탐지 정밀도가 높고 탐지 속도가 빠른 장점이 있지만, 새로운 악성 앱에 대한 대응 시간이 필요하다는 점과 악성 앱으로 정확한 분류를 위해 악성코드 분석가의 지식이 필요하다는 단점이 있다.

이러한 문제점을 극복하기 위해 다양한 연구가 진행되고 있으며 그 중 정적·동적 분석을 이용해 인공지능 구현 기술 중 하나인 기계학습을 활용한 방법의 연구도 있다[4-6]. 일반적으로 기계학습 기반 탐지 모델의 성능은 학습을 위해 어떤 데이터 표현(특성)을 모델에 주입하는가에 따라 달라질 수 있다. 따라서 기계학습을 활용한 악성 앱 탐지 모델의 성능을 높일 수 있는 적절한 특성 선별이 필요하다.

본 논문에서는 악성 앱의 탐지 성능을 향상할 수 있는 특성 선택 및 특성 추출을 이용한 특성 선별 방법을 제안하고 선별된 특성으로 악성 앱의 탐지 성능 평가를 위해 기계학습 모델을 생성하여 학습·검증하였다.

본 논문의 구성은 2장에서는 안드로이드 권한 및 API와 특성 공학에 대해 살펴보고 3장에서는 제안하는 연구 방법을 설명하며 4장에서는 제안된 특성 정보로 안드로이드 악성 앱 탐지 모델의 실험 결과를 분석하고, 5장에서 결론 및 향후 연구에 대하여 논한다.

2. 관련 연구

기존의 연구들은 기계학습 기반의 지능형 악성코드 판별을 실시간에 가능하게 하는 특성 조합을 선택하는 알고리즘을 제안하거나[7] 안드로이드 권한(Permission) 특성을 이용, 특성 선택 알고리즘을 통해 선택된 특성과 이 특성들에 가중치를 부여한 악성 앱 탐지의 방법론을 제시하였다[8]. 본 연구에서는 안드로이드 특성 중 권한뿐 아니라 API를 추가하여 하나의 특성 단위 또는 특성들을 조합할 때 악성 앱 탐지의 효율성을 비교·분석하였다. 또한 특성을 선별할 때 특성을 선택 후 추출하여 악성 앱의 탐지 성능 저하를 최소화하면서 학습 소요 시간을 감소할 수 있는지에 대해 다양한 기계학습 모델들로 비교·분석하였다.

2.1 안드로이드 앱의 구조

구글은 새로운 스토어 정책으로 2021년 8월부터 신규 앱에 대하여 Google Play Store에 APK 형식이 아닌 ABB(Android App Bundle)를 사용하여 게시하도록 하였다[9]. 기존 앱은 APK 형식으로 배포되더라도 새롭게 게시되는 앱들 ABB 형식으로 되어야 한다. ABB는 앱의 모든 컴파일된 코드 및 리소스를 포함하며 APK 생성 및 서명을 Google Play에 맡기는 게시 형식으로 앱을 다운하는 기기에 최적화된 APK를 동적으로 생성한다. 기존 APK 형식과 ABB 형식의 차이는 Fig. 1과 같다.

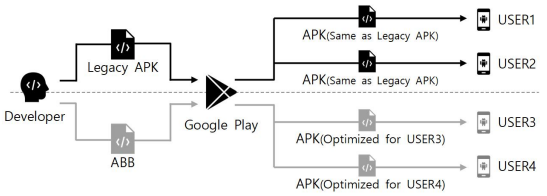

Fig. 1. The difference between APK and ABB

2.1.1 권한(Permission)

사용자 개인정보 보호를 위해 제한된 데이터, 제한된 작업에 대한 앱 권한이 필요하다. 제한된 데이터란 시스템의 상태 및 사용자의 연락처 정보 등이며 제한된 작업이란 페어링 된 기기에 연결 및 오디오 녹음 등을 말한다. 권한의 기능에 따라 사용자의 승인 없이 시스템에서 자동으로 권한이 부여되거나 사용자에게 권한 승인을 요청하게 된다.

안드로이드에서는 설치시간 권한, 런타임 권한, 특별 권한을 비롯한 권한 유형이 있으며 각 권한은 시스템에서 앱에 권한을 부여할 때 액세스할 수 있는 제한된 데이터의 범위와 실행할 수 있는 제한된 작업 범위를 표시한다[10]. 안드로이드의 권한 보호 레벨(Protection Level)은 Normal, dangerous, Signature, SignatureOrSystem 으로 나뉜다[11].

2.1.2 API(Application Programming Interface)

안드로이드 앱은 Java 또는 Kotlin으로 개발되어 달빅 가상머신(Dalvik virtual machine) 또는 ART (Android Runtime)에 최적화된 바이트 코드로 만들어진다. 이 바이트 코드는 APK 내 classes.dex 파일로 저장되며 앱에서 사용하는 API와 데이터에 대한 정보

가 포함되어 있다[12]. 악성 앱에서는 악성 행위에 필요한 모바일 단말기 상태 정보 호출, SMS 내용 접근 및 전송 호출, 연락처 내용 호출, 위치 정보 호출 등 특정 API가 주로 포함된 경우가 많다.

2.2 특성 공학(Feature engineering)

특성 공학이란 기계학습 모델에 사용할 유용한 특성을 추출하는 작업으로 특성 선택(feature selection), 특성 추출(feature extraction)의 2가지 기법이 있다.

2.2.1 특성 선택

기계학습 모델로 데이터를 학습하고 평가하는 과정에서 가장 유용한 특성의 부분집합을 선택하는 과정이 특성 선택이다. 특성이 너무 많으면 모델이 복잡해지고 과대 적합 가능성이 커지기 때문에 관련이 높은 특성 부분집합(Feature Subset)만 선택하여 특성의 수를 줄이는 게 필요하다.

특성 선택 방법으로는 특성 간 관련성으로 측정하는 Filter 방법, 특성 부분집합의 유용성을 측정하는 Wrapper 방법, 특성 부분집합의 유용성 측정에 과적합을 줄이기 위해 내부적으로 페널티(Penalty)를 사용하는 Embedded 방법이 있다[13].

2.2.2 특성 추출

특성 추출은 원본 특성들의 조합으로 새로운 특징을 생성한다. 고차원의 원본 특성 공간을 저차원의 새로운 특성 공간으로 투영시키는 것이다[14]. 가장 대표적인 알고리즘으로 PCA(Principle Component Analysis)가 있다.

2.3 기계학습 알고리즘

기계학습은 학습 문제의 형태에 따라 나뉜다. 학습 데이터의 속성이 무엇을 분석할지 정의하는 데이터인 레이블(label)의 유무에 따라 지도학습(Supervised Learning)과 비지도학습(Unsupervised Learning)으로 나뉘며, 주어진 상태(state)에 최적의 행동(action)의 선택을 학습하는 강화학습(Reinforcement Learning) 등이 있다.

본 논문에서는 Logistic Regression, Multi-layer Perceptron(Artificial Neural Network), Support Vector Machine, K-Nearest Neighbors, Random Forest의 기계학습 알고리즘을 사용하였다.

3. 연구 방법론

본 연구에서는 특성 종류를 다양화(권한과 API)하여 특성 선택 시 개수 및 선택 방법의 변화가 악성 앱 탐지 성능에 미치는 영향을 비교·분석하고 차원 축소를 추가 시도하여 학습 소요 시간의 단축 등 탐지 효율성의 이점을 비교·분석하였다. Fig. 2는 본 연구에서 제안하는 연구 방법의 흐름도이다.

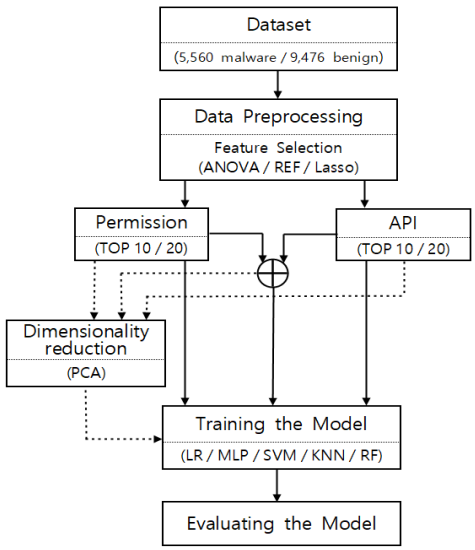

Fig. 2. A simplified workflow of proposed method

3.1 데이터세트 구성 및 전처리

악성 앱 탐지 연구를 위해 사용한 데이터세트는 안드로이드 악성 앱 탐지의 다단계 분류기 융합 접근법인 DroidFusion[15] 연구에 사용한 데이터로 인터넷에서 제공하는 데이터세트를 활용하였다. 9,476개 정상 앱과 5,560개 악성 앱으로 총 15,036개의 앱 표본으로 구성되어 있으며, 공개적으로 널리 사용되는 Drebin[16] 및 Android Malgenome[17] 프로젝트에서 사용한 샘플로 부터 추출한 데이터세트이다.

데이터세트의 특성을 권한(Permission)과 API로 구분하였고 정상 앱과 악성 앱의 구별을 위해 레이블(label)을 부여하였다.

3.2 특성정보의 선별

특성 정보의 선별은 권한과 API 별 특성 선택 방법(Feature Selection)인 Filter 방법 알고리즘의 분산분

석(ANOVA) F검점 통계값(F-value), Wrapper 방법 알고리즘의 RFE(Recursive Feature Elimination), Embedded Method 알고리즘의 Lasso(Least Absolute Shrinkage and Selection Operator)로 상위 10개, 20개를 각각 선별하였다. 그리고 각 알고리즘으로 선별한 특성 중에서 중복되는 특성으로만 최종 특성으로 선별하였으며 Table 1, 2, 3, 4와 같은 결과를 얻었다.

Table 1. Selected feature among the top 10 permissions where D stands for duplicate

	Permission	D
1	SEND_SMS	3
2	WRITE_HISTORY_BOOKMARKS	3
3	ACCESS_LOCATION_EXTRA_COMMANDS	2
4	DELETE_CACHE_FILES	2
5	READ_PHONE_STATE	2
6	READ_SMS	2

Table 2. Selected feature among the top 10 API where D stands for duplicate

	API	D
1	android.telephony.gsm.SmsManager	2
2	android.telephony.SmsManager	2
3	createSubprocess	2
4	TelephonyManager.getDeviceId	2
5	transact	2

Table 3. Selected feature among the top 20 permissions where D stands for duplicate

	Permission	D
1	ACCESS_LOCATION_EXTRA_COMMANDS	3
2	READ_PHONE_STATE	3
3	READ_SMS	3
4	SEND_SMS	3
5	WRITE_HISTORY_BOOKMARKS	3
6	ADD_VOICEMAIL	2
7	CONTROL_LOCATION_UPDATES	2
8	DELETE_CACHE_FILES	2
9	GET_ACCOUNTS	2
10	HARDWARE_TEST	2
11	INTERNET	2
12	MODIFY_AUDIO_SETTINGS	2
13	READ_HISTORY_BOOKMARKS	2
14	WRITE_APN_SETTINGS	2

Table 4. Selected feature among the top 20 API where D stands for duplicate

	API	D
1	android.telephony.SmsManager	3
2	android.os.IBinder	2
3	android.telephony.gsm.SmsManager	2
4	createSubprocess	2
5	HttpUriRequest	2
6	Ljava.lang.Class.getCanonicalName	2
7	Ljava.lang.Class.getDeclaredField	2
8	Ljava.lang.Class.getResource	2
9	Ljava.net.URLDecoder	2
10	onServiceConnected	2
11	ServiceConnection	2
12	TelephonyManager.getDeviceId	2
13	TelephonyManager.getLine1Number	2
14	transact	2

최종적으로 선별된 특성들과 이 선별된 특성에서 주성분 분석(PCA)으로 추출한 특성(feature extraction)들은 Table 5와 같으며 이를 기계학습에 학습하여 악성 앱 탐지 성능 및 소요 시간을 비교 분석하였다.

Table 5. Result of selected feature

feature	method
f(1)	Selected feature among the top 10 permissions
f(2)	Selected feature among the top 20 permissions
f(3)	Selected feature among the top 10 API
f(4)	Selected feature among the top 20 API
f(5)	Combination of f(1) and f(3)
f(6)	Combination of f(3) and f(4)
f(7)	feature extracted from f(1)
f(8)	feature extracted from f(2)
f(9)	feature extracted from f(3)
f(10)	feature extracted from f(4)
f(11)	feature extracted from f(5)
f(12)	feature extracted from f(6)

4. 실험 및 결과

4.1 실험 환경

실험에 활용한 컴퓨터 환경은 AMD Ryzen 5600X CPU와 32GB 메모리를 기반으로 가상환경(VMware Workstation)의 6 core CPU와 4GB 메모리, 200GB

하드디스크의 환경에서 실험하였고 기계학습을 위해 파이썬(Python), 사이킷런(Scikit-learn) 등의 개발 언어 및 라이브러리를 사용하였으며 정리하면 Table 6과 같다.

Table 6. Experimental environment

	Specification
OS	Windows 10
CPU	AMD Ryzen 5600X
RAM / SSD	32GB / 1TB
VMware Workstation	VMware Workstation 16 OS: Windows 10 CPU: 6 core RAM / HDD: 4GB / 200GB
Language	Python 3.9.8
Tool	Jupyter Notebook
Library	Scikit-learn, pandas, numpy, matplot, etc

4.2 모델 구성

정상 앱 9,476개와 악성 앱 5,560개로 구성된 총 15,036개의 데이터(raw data)를 8:2의 비율로 훈련 데이터(train set)와 테스트 데이터(test set)로 구분하였다. 또한 클래스가 불균형한 데이터로는 좋은 모델을 만들 수 없으므로 원데이터(raw data)의 정상 앱과 악성 앱의 비율(63% : 37%)을 훈련 데이터와 테스트 데이터에도 동일하게 정상 앱과 악성 앱의 비율을 유지하여 실험하였다.

LR(Logistic Regression), MLP(Multi-Layer Perceptron), SVM(Support Vector Machine), KNN(K-Nearest Neighbors), RF(Random Forest)의 기계학습 알고리즘에 12가지 선별된 특성과 조합하여 모델을 구성하였다.

학습 시 파라미터(Parameter)값은 SVM의 경우 C 파라미터값을 [0.001, 0.01, 0.1, 1, 10, 25, 50, 100]으로 평가 시 평균적으로 10일 때 가장 좋은 성능을 보여 10을 선택하였으며 KNN 경우 이웃의 개수(K 개수) 파라미터를 [3,5,7,9]로 평가 시 평균적으로 7로 설정했을 때 가장 좋은 성능을 보여 7로 설정하였다. 나머지 기계학습 알고리즘의 파라미터는 사이킷런(scikit-learn ver.1.0.1)의 기본 파라미터값을 사용하였으며 Table 7과 같다.

Table 7. Parameter values of machine learning

ML	Parameter values
LR	Regularization strength(C) : 10 solver : lbfgs, penalty : l2
MLP	Hidden_layer : 100, activation : relu Weight optimization : adam, learning rate : 0.001
SVM	Regularization parameter(C) : 10 Kernel : rbf, probability : true
KNN	Number of neighbors : 7 Weights : uniform
RF	Number of trees : 100

4.3 악성 앱 탐지 성능 비교

선별된 특성으로 각 기계학습 모델에 악성 앱 탐지 평가를 5회씩 수행하여 평가 방법인 정확도, 정밀도, 재현율, F1 스코어의 평균값과 소요 시간 및 ROC AUC 값을 비교하였다.

그중 정밀도는 악성 앱 이라고 판단한 앱 중 실제 악성 앱에 해당하는 비율을 뜻하며 이진 분류의 성능 평가 시 중요한 지표기도 한다. 정밀도 결과에서 특성 선택 시 더 많은 수의 특성 개수에서 중복된 특성을 선택한 것과 권한과 API를 조합할 때 즉, 특성의 여러 종류를 조합할 때 탐지 성능이 평균적으로 향상되었다.

특성을 선별하여 중복된 특성을 선택 시 Fig. 3과 같이 더 많은 개수의 특성에서 선택하였을 때 좋은 성능을 나타냈다. 특성의 조합 및 기계학습 별 탐지 성능 결과는 종합적으로 Table 8과 같이 볼 수 있다.

Table 8. Precision (mean of 5 times, %)

LR	f(1)	f(2)	f(3)	f(4)	f(5)	f(6)
	83.50	92.28	83.81	90.03	86.38	93.26
	f(7)	f(8)	f(9)	f(10)	f(11)	f(12)
	79.75	91.14	79.78	87.88	82.49	89.33
MLP	f(1)	f(2)	f(3)	f(4)	f(5)	f(6)
	81.04	93.04	84.67	95.03	88.18	96.54
	f(7)	f(8)	f(9)	f(10)	f(11)	f(12)
	80.96	93.27	84.25	93.38	87.85	95.56
SVM	f(1)	f(2)	f(3)	f(4)	f(5)	f(6)
	81.00	93.50	84.65	94.19	88.26	96.05
	f(7)	f(8)	f(9)	f(10)	f(11)	f(12)
	80.49	92.91	84.60	89.93	85.60	92.44
KNN	f(1)	f(2)	f(3)	f(4)	f(5)	f(6)
	84.09	81.78	84.66	90.69	88.11	94.98
	f(7)	f(8)	f(9)	f(10)	f(11)	f(12)
	76.28	88.80	84.60	91.18	87.84	94.07
RF	f(1)	f(2)	f(3)	f(4)	f(5)	f(6)
	81.10	93.34	84.65	94.65	88.14	96.41
	f(7)	f(8)	f(9)	f(10)	f(11)	f(12)
	81.02	93.31	84.65	94.37	88.07	95.83

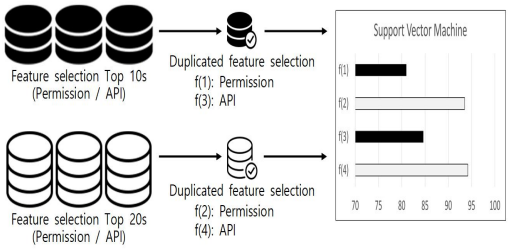

Fig. 3. Comparison(SVM) of precision of feature selection

즉, f(1)보다 f(2), f(3)보다 f(4)가 정밀도가 높은 것으로 보아 권한 특성보다 API 특성에서, 그리고 더 많은 개수에서 중복된 특성을 선택한 특성에서 좋은 결과를 보였고 권한과 API를 조합한 f(5), f(6)에서 탐지 성능이 더 향상된 것을 나타냈으며. 차원 축소한 f(7)~f(12)에서도 동일했다. 그리고 차원 축소 전에는 MLP(Multi-Layer Perceptron), 차원 축소 후에는 RF(Random Forest)에서 가장 좋은 성능을 보였다.

4.4 모델 학습 시간 비교

선별된 특성별로 모델 학습 시간을 비교한 결과, 차원 축소한 f(7)~(12)와 차원 축소 전 f(1)~(6)를 비교하면 평균적으로 탐지 성능 및 학습 시간이 감소한 것을 알 수 있었다. Fig. 4, Fig. 5와 같이 LR(Logistic Regression)과 KNN(K-Nearest Neighbors)에서 학습 시간의 감소폭이 컸고 SVM(Support Vector Machine)에서는 시간이 늘어나 학습 시간 단축 모델로는 적합하지 않았으며, Fig. 6과 같이 K-Nearest Neighbors에서는 탐지 성능이 비슷하여 학습 시간 단축의 이점을 얻기에 유용한 모델이었다.

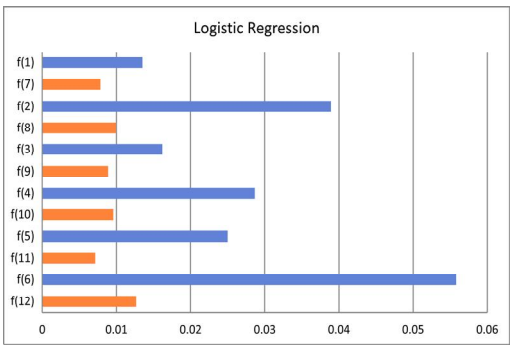

Fig. 4. Comparison of learning time by LR (sec)

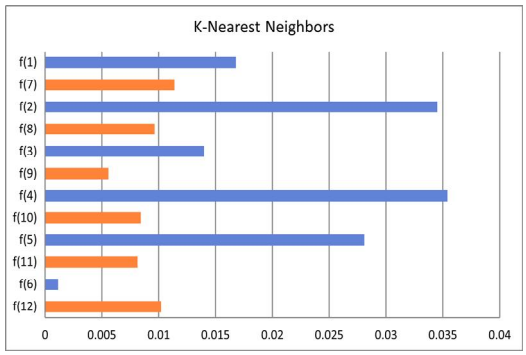

Fig. 5. Comparison of learning time by KNN (sec)

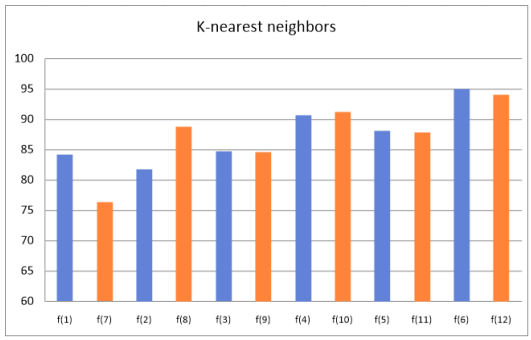

Fig. 6. Comparison of precision by KNN

4.5 ROC AUC 비교

하나의 특성이 아닌 권한과 API를 조합한 f(6) 과 f(12)에서 모델별 모두 97 이상의 수치를 보였으며 Fig. 7의 MLP(Multi-layer Perceptron)와 Fig. 8의 RF (Random Forest)는 99 이상의 높은 수치를 나타냈다.

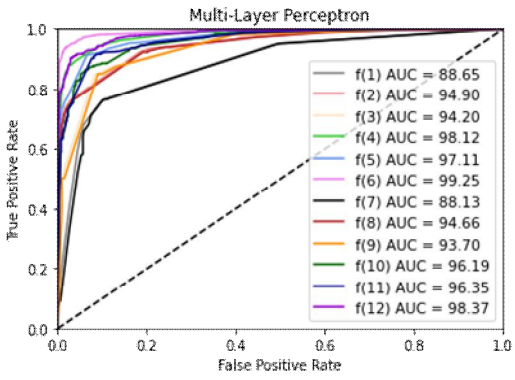

Fig. 7. ROC-AUC by Multi-Layer Perceptron

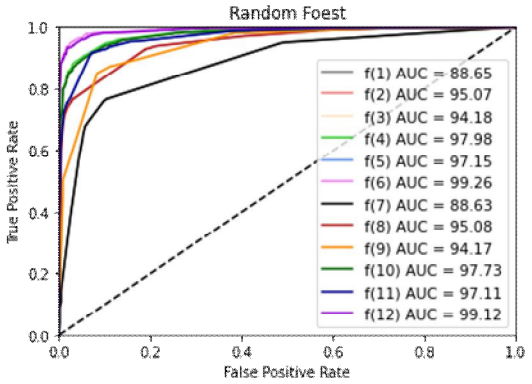

Fig. 8. ROC-AUC by Random Forest

5. 결론

본 연구로 기계학습을 이용한 악성 앱 탐지 시 특성의 선별 방법에 따라 탐지 성능을 향상시킬 수 있음을 확인하였다. 다수의 특성 선택(feature selection) 알고리즘에서 선택한 특성 중 중복되는 특성을 선별하여 기계학습 모델에 적용하였고, 특성 선택 과정에서 선별하는 특성 개수에 따라 탐지 정확도를 높일 수 있는 것을 알 수 있었다.

안드로이드 앱의 권한과 API를 이용하여 기계학습 모델에 적용 시 권한보다 API를 활용할 때 상대적으로 좋은 정밀도를 나타냈으며, 권한과 API를 조합할 때 더욱 높은 정밀도를 보여 적절한 특성의 조합이 악성 앱의 탐지 정밀도를 높일 수 있는 조건임을 알 수 있었다.

특성 선택 후 특성 추출(feature extraction)로 차원을 축소하면 평균적으로 다소 정밀도는 낮아졌지만, 학습 소요 시간이 감소하는 이점이 있었다. KNN 모델의 경우, 정밀도는 거의 같지만 학습 소요 시간이 감소한 결과를 볼 수 있었다. 기계학습을 이용하여 악성 앱을 탐지할 때 다양한 특성 선택 알고리즘을 혼합하여 적절한 수의 특성을 선별하면 악성 앱의 탐지 정밀도를 높일 수 있으며, 안드로이드 앱에서 추출할 수 있는 다른 특성들을 조합한다면 악성 앱 탐지 성능을 더욱 향상시킬 수 있을 것으로 기대한다. 또한, 디바이스의 성능이 낮거나 빠른 탐지 결과물 도출이 필요할 경우 특성 선택 후 차원을 축소하는 방법이 유용하게 활용될 수 있을 것이다.

향후 최신 신·변종 안드로이드 악성 앱의 데이터를 이용한 탐지 성능 평가로 연구의 신뢰성을 높이고, 입력 데이터로서 제안한 선별된 특성의 변화뿐만 아니라 기계학습 모델의 설정값 변화를 통해 최적의 악성 앱 탐지 정밀도를 가질 수 있는 기법을 고안하는 연구를 진행할 것이다.

REFERENCES

[1] R. Shafin et al. (2020). Artificial intelligence-enabled cellular networks: A critical path to beyond-5g and 6g, *IEEE Wireless Communications, 27(2)*, 212-217.

[2] McAfee. (2021). *McAfee Mobile Threat Report 2021.*

[3] Statcounter Global Stats. (2021. Dec.). *Mobile Operating System Market Share Worldwide*, (Online). https://gs.statcounter.com/os-market-share/mobile/worldwide

[4] F. A. Narudin, A. Feizollah, N. B. Anuar & A. Gani. (2016. Jan.). Evaluation of machine learning classifiers for mobile malware detection, *Soft Computing, 20.* 343-357.
DOI : 10.1007/s00500-014-1511-6

[5] S. E. Kang, N. V. Long & S. H. Jung. (2018. June). Android malware detection using permission-based machine learning approach, *Journal of The Korea Institute of Information Security & Cryptology, 28(3).* 617-623.
DOI : 10.13089/JKIISC.2018.28.3.617

[6] H. Cho. (2019). *A study on Android malware event trigger based on reinforcement learning*, Master Thesis, Graduate School of Soongsil University, Seoul.

[7] J. G. Joo, I. S. Jeong & S. H. Kang. (2019). An optimal feature selection method to detect malwares in real time using machine learning, *Journal of Korea Multimedia Society, 22(2),* 203-209.
DOI : 10.9717/kmms.2019.22.2.203

[8] J. H. Bo & K. H. Lee. (2020. June). Advanced feature selection method on Android malware detection by machine learning, *Journal of the Korea Institute of Information Security & Cryptology, 30(3),* 357-367.
DOI : 10.13089/JKIISC.2020.30.3.357

[9] Android Developer. (n. d.). Android App Bundle Information (Online). https://developer.android.com/guide/app-bundle

[10] Android Developer. (n. d.). Authority on Android. (Online). https://developer.android.com/guide/topics

/permissions/overview

[11] Android Developer. (n. d.). Android developer 〉 Document 〉 Guide (Online). https://developer.android.com/guide/topics/manifest/permission-element

[12] J. H. Yu, I. H. Seo & S. J. Kim. (2017). Study on DNN based Android malware detection method for mobile environment, *KIPS Transactions on Computer and Communication Systems, 6(3),* 159-168. DOI : 10.3745/KTCCS.2017.6.3.159

[13] L. Li et al. (2017. Aug.). Static analysis of android apps: A systematic literature review, *Information and Software Technology, 88,* 67-95. DOI : 10.1016/j.infsof.2017.04.001

[14] S. Sarangi, M. Sahidullah & G. Saha. (2020. Sept.). Optimization of data-driven filterbank for automatic speaker verification. *Digital Signal Processing, 104.* DOI : 10.1016/j.dsp.2020.102795

[15] S. Y. Yerima & S. Sezer. (2019. Feb.). DroidFusion: A novel multilevel classifier fusion approach for android malware detection, I*EEE Transactions on Cybernetics, 49(2),* 453-466. DOI : 10.1109/TCYB.2017.2777960

[16] D. Arp, M. Spreitzenbarth, M. Hubner, H. Gascon & K. Rieck. (2014. Feb.). Drebin: efficient and explainable detection of android malware in your pocket, *Network and Distributed System Security (NDSS) Symposium*, San Diego, CA, USA. DOI : 10.14722/ndss.2014.23247

[17] Y. Zhou & X. Jiang. (2012. May). Dissecting android malware: characterization and evolution, IEEE Symposium on Security and Privacy, San Fransisco, CA, USA. DOI : 10.1109/SP.2012.16

명 상 준(SangJoon Myeong) [정회원]

· 2007년 2월 : 한신대학교 정보통신학과(학사)
· 2020년 2월 ~ 현재 : 아주대학교 정보통신대학원 사이버보안전공 석사과정
· 관심분야 : 사이버 보안, 기계학습 및 딥러닝
· E-Mail : mjmmm5@ajou.ac.kr

김 강 석(Kangseok Kim) [정회원]

· 2007년 11월 : Indiana University at Bloomington 컴퓨터 공학(박사)
· 2010년 9월 ~ 2016년 2월 : 아주대학교 대학원 지식정보공학과 연구교수
· 2016년 3월 ~ 현재 : 아주대학교 사이버보안학과 부교수
· 관심분야 : 빅데이터 응용보안, 기계학습 및 딥러닝
· E-Mail : kangskim@ajou.ac.kr